Neural Information Extraction from Natural Language Text

Dissertation
an der Fakultät für Mathematik, Informatik und Statistik
der Ludwig–Maximilians–Universität
München

Pankaj Gupta

München 2019

Erstgutachter: Prof. Dr. Hinrich Schütze
Zweitgutachter: Associate Prof. Dr. Ivan Titov, PhD
Drittgutachter: Assistant Prof. Dr. William Wang, PhD

Tag der Einreichung: 30. April 2019
Tag der mündlichen Prüfung: 26. September 2019

Eidesstattliche Versicherung
(Siehe Promotionsordnung vom 12.07.11, § 8, Abs. 2 Pkt. .5.)

Hiermit erkläre ich an Eides statt, dass die Dissertation von mir selbstständig ohne unerlaubte Beihilfe angefertigt ist.

München, 26. September 2019

Pankaj Gupta

Abstract

Natural language processing (NLP) deals with building computational techniques that allow computers to automatically analyze and meaningfully represent human language. With an exponential growth of data in this digital era, the advent of NLP-based systems has enabled us to easily access relevant information via a wide range of applications, such as web search engines, voice assistants, etc. To achieve it, a long-standing research for decades has been focusing on techniques at the intersection of NLP and machine learning.

In recent years, deep learning techniques have exploited the expressive power of Artificial Neural Networks (ANNs) and achieved state-of-the-art performance in a wide range of NLP tasks. Being one of the vital properties, Deep Neural Networks (DNNs) can automatically extract complex features from the input data and thus, provide an alternative to the manual process of handcrafted feature engineering. Besides ANNs, Probabilistic Graphical Models (PGMs), a coupling of graph theory and probabilistic methods have the ability to describe causal structure between random variables of the system and capture a principled notion of uncertainty. Given the characteristics of DNNs and PGMs, they are advantageously combined to build powerful neural models in order to understand the underlying complexity of data.

Traditional machine learning based NLP systems employed shallow computational methods (e.g., SVM or logistic regression) and relied on handcrafting features which is time-consuming, complex and often incomplete. However, deep learning and neural network based methods have recently shown superior results on various NLP tasks, such as machine translation, text classification, named-entity recognition, relation extraction, textual similarity, etc. These neural models can automatically extract an effective feature representation from training data.

This dissertation focuses on two NLP tasks: *relation extraction* and *topic modeling*. The former aims at identifying semantic relationships between entities or nominals within a sentence or document. Successfully extracting the semantic relationships greatly contributes in building structured knowledge bases, useful in downstream NLP application areas of web search, question-answering, recommendation engines, etc. On other hand, the task of topic modeling aims at under-

standing the thematic structures underlying in a collection of documents. Topic modeling is a popular text-mining tool to automatically analyze a large collection of documents and understand topical semantics without actually reading them. In doing so, it generates word clusters (i.e., topics) and document representations useful in document understanding and information retrieval, respectively.

Essentially, the tasks of relation extraction and topic modeling are built upon the quality of representations learned from text. In this dissertation, we have developed task-specific neural models for learning representations, coupled with relation extraction and topic modeling tasks in the realms of supervised and unsupervised machine learning paradigms, respectively. More specifically, we make the following contributions in developing neural models for NLP tasks:

1. *Neural Relation Extraction*: Firstly, we have proposed a novel recurrent neural network based architecture for table-filling in order to jointly perform entity and relation extraction within sentences. Then, we have further extended our scope of extracting relationships between entities across sentence boundaries, and presented a novel dependency-based neural network architecture. The two contributions lie in the supervised paradigm of machine learning. Moreover, we have contributed in building a robust relation extractor constrained by the lack of labeled data, where we have proposed a novel weakly-supervised bootstrapping technique. Given the contributions, we have further explored interpretability of the recurrent neural networks to explain their predictions for the relation extraction task.

2. *Neural Topic Modeling*: Besides the supervised neural architectures, we have also developed unsupervised neural models to learn meaningful document representations within topic modeling frameworks. Firstly, we have proposed a novel dynamic topic model that captures topics over time. Next, we have contributed in building static topic models without considering temporal dependencies, where we have presented neural topic modeling architectures that also exploit external knowledge, i.e., word embeddings to address data sparsity. Moreover, we have developed neural topic models that incorporate knowledge transfers using both the word embeddings and latent topics from many sources. Finally, we have shown improving neural topic modeling by introducing language structures (e.g., word ordering, local syntactic and semantic information, etc.) that deals with bag-of-words issues in traditional topic models.

   The class of proposed neural NLP models in this section are based on techniques at the intersection of PGMs, deep learning and ANNs.

Here, the task of neural relation extraction employs neural networks to learn representations typically at the sentence level, without access to the broader docu-

ment context. However, topic models have access to statistical information across documents. Therefore, we advantageously combine the two complementary learning paradigms in a *neural composite model*, consisting of a neural topic and a neural language model that enables us to jointly learn thematic structures in a document collection via the topic model, and word relations within a sentence via the language model.

Overall, our research contributions in this dissertation extend NLP-based systems for relation extraction and topic modeling tasks with state-of-the-art performances.

Zusammenfassung

Natural language processing (NLP) umfasst die Technologien, die es Computern erlauben, menschliche Sprache (Natural Language) zu analysieren und zu interpretieren (Processing). Mit dem exponentiellen Wachstum an Daten im Digitalisierungszeitalter, werden NLP-basierte System benötigt, um einen einfachen Zugang zu den relevanten Informationen in Texten zu erhalten. Bekannte Applikationen in diesem Bereich sind Suchmaschinen im Internet, Sprachassistenten, etc. Um das zu erreichen, war Jahrzehnte-lange Forschung notwendig, die sich auf Techniken an der Schnittstelle von NLP und maschinellem Lernen fokussierte. In den letzten Jahren wurden Deep-Learning-Technologien für NLP-Aufgabenstellungen angewendet, die die Mächtigkeit neuronaler Netze nutzten und damit state-of-the-art Ergebnisse erzielen konnten.

Es ist eine wesentliche Fähigkeit tiefer Neuronaler Netze (DNN), automatisch komplexe Merkmale aus Daten zu extrahiren und so eine Alternative zu dem manuellen Explorieren von Merkmalen zu bieten. Neben den künstlichen neuronalen Netzen (ANN) haben die probabilistischen graphischen Modelle (PGM), die eine Verbindung von Graphen-Theorie und probabilistischen Methoden darstellen, eine ähnliche Fähigkeit, kausale Strukturen zwischen Zufallsvariablen eines Systems zu beschreiben und dabei mit der vorhandenen Unsicherheit prinzipiell umzugehen. Die unterschiedlichen, komplementären Charakteristiken von ANNs und PGMs werden in dieser Arbeit zu einem mächtigen neuronalen Modell kombiniert, um die in den natürlichsprachlichen Daten vorhandene Komplexität noch besser zu verstehen.

Bisherige Ansätze für NLP-Systeme, die auf maschinellem Lernen basieren, haben vergleichsweise einfache, nicht sehr rechenintensive Methoden angewandt, wie z.B. Support Vector Machines (SVM) oder logistic Regression und sind abhängig von aussagekräftigen Merkmalen, die durch Fachexperten erzeugt werden müssen. Das Erzeugen von Merkmalen durch Fachexperten ist jedoch zeitintensiv, komplex und kann oft nur unvollständig sein. Dahingegen haben das tiefe Lernen sowie auf neuronalen Netzen basierte Methoden in jüngerer Zeit klar überlegene Ergebnisse bei NLP-Aufgaben wie maschinellem Übersetzen, Textklassifikation, Named-Entity-Erkennung, Relationsextraktion, Erkennung textueller Ähnlichkeit,

etc. gezeigt. Diese neuronalen Modelle können automatisch eine effektive Merkmalsrepräsentation aus Trainingsdaten lernen.

Vor diesem Hintergrund konzentriert sich die Dissertation im wesentlichen auf zwei NLP-Aufgabenstellungen: die *Relationsextrakion* und das sogenannte *Topic Modeling*. Die Relationsextraktion hat zum Ziel, semantische Beziehungen zwischen Entitäten innerhalb eines Satzes oder eines Dokumentes zu erkennen. Semantische Beziehungen zwischen Entitäten zu erkennen trägt stark zu dem Aufbau strukturierter Wissensbasen bei, die in NLP-Applikationen wie der Internetsuche, Frage-Antwort-Systemen, Recommender-Systemen usw. eingesetzt werden. Auf der anderen Seite hat das Topic Modelling zum Ziel, Themen in Dokumenten zu analysieren. Topic Modeling ist ein beliebtes Text-Mining Verfahren, um große Dokumentenmengen auf darin vorkommende Themen zu untersuchen, ohne sie lesen zu müssen. Dabei generiert das Verfahren Wort-Cluster, die als Themen gesehen werden können, und Dokumentenrepräsentationen, die für das Verstehen von Dokumenten sowie für das Information Retrieval verwendet werden können.

Im wesentlichen bauen sowohl die Relationsextraktion als auch das Topic Modeling auf Textrepräsentationen auf, die von den Texten gelernt werden müssen. In der vorliegenden Arbeit haben wir aufgabenstellungsspezifische neuronale Modelle für das Erlernen von Textrepräsentationen entwickelt, wobei wir das Lernen jeweils im Zusammenspiel mit der Relationsextraktion und dem Topic Modeling gestalten. Wir verfolgen dabei jeweils die beiden Paradigmen des überwachten und des unüberwachten maschinellen Lernens. Genauer gesagt besteht unser Beitrag zu der Entwicklung neuronaler Modelle im Bereich des NLP in folgendem.

1. *Neuronale Relationsextraktion*: wir schlagen eine neuartige Architektur für rekurrente neuronale Netze vor, die Entitäten- und Relationsextraktion in einem Schritt macht. Darüber hinaus haben wir die Reichweite der Entitäten- und Relationsexktraktion über Satzgrenzen hinaus erweitert und dafür eine neuartige Dependency-basierte neuronale Netzwerkarchitektur entwickelt. Die beiden Hauptbeiträge liegen dabei im Bereich des überwachten maschinellen Lernens. Wir haben weiterhin eine schwach überwachte Bootstrapping Methode eingeführt, um Relationsextraktion ohne annotierte Trainingsdaten durchführen zu können. Wir haben auch die Interpretierbarkeit der rekurrenten neuronalen Netze untersucht, um ihre Funktionsweise bei der Relationsextraktion zu erklären.

2. *Neuronales Topic Modeling*: neben den überwachten neuronalen Architekturen haben wir auch unüberwachte neuronale Modelle entwickelt, um relevante Dokumentenrepräsentationen innerhalb von Topic Modeling Frameworks zu lernen. Zunächst haben wir ein neuartiges dynamisches Topic

Model entwickelt, das Topics in Dokumenten über der Zeit erfasst. Dann haben wir einen Ansatz im Bereich des Topic Modelings entwickelt, der dem prinzipiellen Problem der Knappheit an Trainingsdaten dadurch begegnet, dass er vortrainierte Word-Embeddings nutzt. Darüberhinaus haben wir neuronale Topic Modelle entwickelt, die einen Wissenstransfer aus verschiedensten Quellen leisten können, indem sie auf den jeweiligen Quellen trainierte Word-Embeddings und latente Topics vereinen. Schließlich haben wir gezeigt, wie wir Neuronales Topic Modeling durch die Einführung von Sprachstrukturen (z.B. die Reihenfolge der Wörter, lokale syntaktische wie semantische Informationen, etc.) verbessern, was dabei hilft, Limitierungen der sogenannten Bag-of-Word-Ansätze im traditionellen Topic Modeling zu überwinden.

Dabei basiert die Klasse der vorgeschlagenen neuronalen NLP-Modelle auf Technologien an der Schnittstelle von PGM, tiefem Lernen und künstlichen neuronalen Netzen.

Die Relationsextraktion setzt neuronale Netzwerke ein, um Repräsentationen typischerweise auf Satzebene zu lernen, ohne den erweiterten Dokumentenkontext zu betrachten. Topic Modelle haben demgegenüber Zugang zu statistischer Information über alle Dokumente hinweg. Wir verbinden die beiden komplementären Lernparadigmen in einem Modell, bestehend aus einem neuronalen Topic Modell und einem neuronalen Sprachmodell, und ermöglichen so, gemeinschaftlich thematische Strukturen in einem Dokumentenset sowie Wortrelationen in Sätzen zu lernen.

Alles in allem konnten wir zeigen, dass die in der vorliegenden Arbeit entwickelten Ansätze den Stand der Technik bei Relationsextraktion und Topic Modeling erweitern.

Acknowledgements

I am grateful to many amazing people I met during the course of my PhD. First and foremost, I would like to thank my PhD supervisor Hinrich Schütze. You gave me a lot of freedom to explore and pursue my research interests. I am grateful for your valuable guidance, perception and constructive feedback that made a big difference in the success of my PhD thesis. Thank you also for reading my drafts of papers and providing helpful comments.

I am also grateful to my colleagues at Siemens, particularly Bernt Andrassy, Florian Buettner and Ulli Waltinger. Bernt, I thank you for providing me several opportunities to apply my research findings into industrial applications at Siemens. Your advice kept me motivated, especially in my difficult times. Ulli, I am thankful for your support, confidence and belief in me. I admire you for providing me an effective research environment at Siemens. Florian, I have learnt a lot from you about probabilistic graphical models and their formulations. I thank you for reading my drafts and collaborating with me on few papers. I also thank Mark Buckley and Stefan Langer for proof-reading some of my drafts of papers. I am fortunate having such opportunities to experience challenges of both the worlds: academia and industry at the same time.

I also want to thank amazing students (master thesis/intern) and co-authors for being part of my research and industrial tasks. I really enjoyed my collaborations with talented master intern: Usama Yaseen and master students (in chronological order): Subburam Rajaram, Yatin Chaudhary and Khushbu Saxena. I sincerely thank you all for being part of my journey as co-authors and friends. I also appreciate and thank Thomas Runkler (at Siemens, Munich) for your support in bridging the collaboration between me and students. Without you, it would not have been possible to setup such collaborations between Technical University of Munich and Siemens. Thank you Benjamin Roth (at LMU, Munich) for your insightful suggestions. I also want to thank Angela Fahrni and Abdel Labbi for your valuable guidance during my research internship at IBM Research, Zurich.

Most importantly, this dissertation would not have been possible without the constant support, motivation and love from my parents: Braj Mohan Gupta and Vidhya Gupta; wife: Soumya Dubey; sisters: Hemlata Dhanotiya and Preeti

Gupta; and brother-in-laws: Vinod Dhanotiya and Nitin Gupta. Soumya, thank you very much for being patient during this long journey, especially in the busiest times for paper submissions. Literally, you kept me alive and fueled positive energy in difficult times. I will never forget how we also celebrated my success in different conferences.

Lastly, I express my gratitude to Siemens AG, Corporate Technology, Machine Intelligence & Siemens AI Lab (Munich, Germany), and the Bundeswirtschaftsministerium (BMWi) via the Grant 01MD15010A (Smart Data Web) for financial support of my PhD.

Contents

Publications and Declaration of Co-Authorship

Chapter 2

Chapter 2 corresponds to the following publication:

> Pankaj Gupta, Hinrich Schütze, Bernt Andrassy; **Table Filling Multi-Task Recurrent Neural Network for Joint Entity and Relation Extraction**; Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers (Osaka, Japan, December 11-16, 2016), pages 2537–2547

I regularly discussed this work with my advisor, but I conceived of the original research contributions and performed implementation and evaluation. I wrote the initial draft of the article and did most of the subsequent corrections. My advisor and Bernt Andrassy assisted me in improving the draft.

Chapter 3

Chapter 3 corresponds to the following publication:

> Pankaj Gupta, Benjamin Roth and Hinrich Schütze; **Joint Bootstrapping Machines for High Confidence Relation Extraction**; Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (New Orleans, Louisiana, USA, June 1-6, 2018), pages 26–36

I regularly discussed this work with my advisor, but I conceived of the original research contributions and performed implementation and evaluation. I wrote the initial draft of the article and did most of the subsequent corrections. My advisor and Benjamin Roth assisted me in improving the draft.

Chapter 4

Chapter 4 corresponds to the following publication:

> Pankaj Gupta, Subburam Rajaram, Thomas Runkler and Hinrich Schütze; **Neural Relation Extraction Within and Across Sentence Boundaries**; Proceedings of the Thirty-third AAAI Conference on Artificial Intelligence (AAAI-19) (Honolulu, Hawaii, USA, January 27 - February 1, 2019)

Subburam Rajaram contributed in dataset preparation and collaborated with me on designing the implementation and experimental evaluation. I also regularly discussed this work with my coauthors. Apart from these explicitly declared exceptions, I conceived of the original research contributions and performed implementation and evaluation. I wrote the initial draft of the article and did most of the subsequent corrections. My coauthors assisted me in improving the draft.

Chapter 5

Chapter 5 corresponds to the following publication:

> Pankaj Gupta, Subburam Rajaram, Hinrich Schütze, Bernt Andrassy; **Deep Temporal-Recurrent-Replicated-Softmax for Topical Trends over Time**; Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (New Orleans, Louisiana, USA, June 1-6, 2018), pages 1079–1089

Subburam Rajaram collaborated with me on designing the implementation and experimental evaluation. I also regularly discussed this work with my coauthors. Apart from these explicitly declared exceptions, I conceived of the original research contributions and performed implementation and evaluation. I wrote the initial draft of the article and did most of the subsequent corrections. My coauthors assisted me in improving the draft.

Chapter 6

Chapter 6 corresponds to the following publication:

> Pankaj Gupta, Yatin Chaudhary, Florian Buettner, Hinrich Schütze; **Document Informed Neural Autoregressive Topic Models with Distributional Prior**; Proceedings of the Thirty-third AAAI Conference on Artificial Intelligence (AAAI-19) (Honolulu, Hawaii, USA, January 27 - February 1, 2019)

Yatin Chaudhary collaborated with me on designing the experimental evaluation. I also regularly discussed this work with my coauthors. Apart from these explicitly declared exceptions, I conceived of the original research contributions and performed implementation and evaluation. I wrote the initial draft of the article and did most of the subsequent corrections. My coauthors assisted me in improving the draft.

Chapter 7

Chapter 7 corresponds to the following publication:

> Pankaj Gupta, Yatin Chaudhary, Hinrich Schütze; **Multi-view and Multi-source Transfers in Neural Topic Modeling**; Under Review

Yatin Chaudhary collaborated with me on designing the implementation and experimental evaluation. I also regularly discussed this work with my coauthors. Apart from these explicitly declared exceptions, I conceived of the original research contributions and performed implementation and evaluation. I wrote the initial draft of the article and did most of the subsequent corrections. My coauthors assisted me in improving the draft.

Chapter 8

Chapter 8 corresponds to the following publication:

> Pankaj Gupta, Yatin Chaudhary, Florian Buettner, Hinrich Schütze; **textTOvec: Deep Contextualized Neural Autoregressive Topic Models of Language with Distributed Compositional Prior**; Proceedings of the Seventh International Conference on Learning Representations (ICLR-19) (New Orleans, Louisiana, USA, May 6-9, 2019)

Yatin Chaudhary collaborated with me on designing the implementation and experimental evaluation. Florian Buettner contributed in setting up some of the baselines. I also regularly discussed this work with my coauthors. Apart from these explicitly declared exceptions, I conceived of the original research contributions and performed implementation and evaluation. I wrote the initial draft of the article and did most of the subsequent corrections. My coauthors assisted me in improving the draft.

Chapter 9

Chapter 9 corresponds to the following publication:

Pankaj Gupta, Bernt Andrassy, Hinrich Schütze; **Replicated Siamese LSTM in Ticketing System for Similarity Learning and Retrieval in Asymmetric Texts**; Proceedings of the Third COLING Workshop on Semantic Deep Learning (Santa Fe, New Mexico, USA, August 20-26, 2018), pages 1–11

I regularly discussed this work with my advisor, but I conceived of the original research contributions and performed implementation and evaluation. I wrote the initial draft of the article and did most of the subsequent corrections. My advisor and Bernt Andrassy assisted me in improving the draft.

Chapter 10

Chapter 10 corresponds to the following publication:

Pankaj Gupta, Hinrich Schütze; **LISA: Explaining Recurrent Neural Network Judgments via Layer-wIse Semantic Accumulation and Example to Pattern Transformation**; Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP (Brussels, Belgium, October 31 - November 4, 2018), pages 154–164

I regularly discussed this work with my advisor, but I conceived of the original research contributions and performed implementation and evaluation. I wrote the initial draft of the article and did most of the subsequent corrections. My advisor assisted me in improving the draft.

München, 26. September 2019

Pankaj Gupta

Chapter 1

Introduction

1.1 Outline, Contributions and Overall Summary

In this *introductory* chapter, we first introduce some of the supervised neural networks, including Recurrent (RNNs), Recursive (RecvNNs) and Siamese (SNNs) Neural Networks. Then, we discuss in detail about the unsupervised paradigm of learning representations via neural density estimators, especially Restricted Boltzmann Machine (RBM), Neural Autoregressive Distributional Estimation (NADE), Replicated Softmax (RSM) and Neural Autoregressive Topic Model (DocNADE). This class of stochastic graphical models forms the basis for our neural topic learning in text documents. Following the two paradigms in neural network learning, the next section highlights the foundation of distributed representation learning at word and document levels. Moreover, it underlines a need for joint leaning in a composite model, consisting of a topic and a neural language model. Then, we provide an outline for the task of semantic relation extraction (RE) within (intra-) and across (inter-) sentence boundary. Additionally, we also feature major related works in the realms of relation extraction as well as joint entity and relation extraction. Finally, we review recent questions in explainability of neural models. While we describe the basic fundamentals, we briefly mention our contribution(s) in the corresponding sections.

In this dissertation, we organize our **contributions** in form of the following Chapters 2-10, each describing a publication. Moreover, the publications are categorized in the realms of the following research directions, highlighting our contributions:

1. **Relation Extraction (RE)**

    - Chapter 2 $\rightarrow$ Joint entity and relation extraction via a supervised neural table-filling approach

- Chapter 3 → Relation Extraction via a novel weakly-supervised bootstrapping technique
- Chapter 4 → Relation Extraction within and across sentence boundary via a novel dependency-based (supervised) neural architecture
- Chapter 10 → Interpretability of Recurrent neural networks via Layer-wise Semantic Accumulation (LISA) approach in supervised RE

2. **Topic Modeling (TM)**

   - Chapter 5 → Neural dynamic topic model to capture topics over time
   - Chapter 6 → Improve neural topic modeling in the sparse-data settings via knowledge transfer using word embeddings
   - Chapter 7 → Improve neural topic modeling in the sparse-data settings via knowledge transfer using both the word embeddings and latent topics from many sources

3. **Composite neural architecture of a Topic and Language model**

   - Chapter 8 → Composite modeling to jointly learn representations from both the global and local semantics, captured respectively by a neural topic and a neural language model; introduce language structures (e.g., word ordering, local syntactic and semantic information, etc.) to deal with bag-of-words issues in topic modeling
   - Chapter 9 → Combine a neural topic and a neural language model for semantic textual similarity within a Siamese network, applied to a real-world industrial application of a ticketing system

To **summarize**, the task of supervised relation extraction (Chapters 2, 4 and 10) applied RNN-based neural models typically at the sentence level (i.e., local view), without access to the broader document context. However, the topic models (Chapters 5, 6, and 7) take a global view in the sense that topics have access to statistical information across documents. Therefore, we naturally extend these neural NLP models to leverage the merits of the two complementary learning paradigms. In doing so, we present a composite model (Chapters 8 and 9), consisting of a neural topic (DocNADE) and a neural language (LSTM) model, that enables us to jointly learn thematic structures in a document collection via the topic model and word relations within a sentence via the language model.

Additionally, Tables 1.1 and 1.2 summarize our contributions in relation extraction, topic modeling and composite modeling, respectively. They further link the concepts (or features) and their related sections (of the introductory chapter) to the relevant publications (Chapter 2-10). Here, we also point out the conference proceedings of each of the publications.

Features	Proposed Relation Extraction Systems				Related Section
	TF-MTRNN (chapter 2) (COLING-16)	*JBM* (chapter 3) (NAACL-18)	*iDepNN* (chapter 4) (AAAI-19)	*LISA* (chapter 10) (EMNLP-18)	
Intra-sentential RE	✓	✓		✓	1.5.1
Inter-sentential RE			✓		1.5.1
Supervised RE	✓		✓	✓	1.2, 1.5.2
Weakly Supervised RE		✓			1.5.4
Joint NER+RE	✓				1.2.1, 1.5.6
Interpretable RE		✓		✓	1.2.1, 1.6

Table 1.1 – *Our multi-fold contributions in Relation Extraction (RE). The symbol ✓ signifies if a feature (or related section) is related to a chapter. Thus, we illustrate how the introduction section (i.e., chapter 1) is related to the rest of the chapters (i.e., our publications).*

Features	Probabilistic Graphical and Neural NLP models of TM							Related Section
	Related Works			**Our Contributions**				
	NADE (chapter 1)	*RSM* (chapter 1)	*DocNADE* (chapter 1)	*RNN-RSM* (chapter 5) (NAACL-18)	*iDocNADEe* (chapter 6) (AAAI-19)	*MVT* (chapter 7)	*textTOvec* (chapter 8) (ICLR-19)	
Tractability	✓		✓		✓	✓	✓	1.3.1, 1.3.2
Static TM	✓	✓	✓		✓	✓	✓	1.3.3, 1.3.4
Dynamic TM				✓				1.3.3, 1.4.3
Autoregressive	✓		✓		✓	✓	✓	1.3.2, 1.3.4
Informed					✓			1.3.4
Word Ordering							✓	1.2.1, 1.4.1
Language Structures							✓	1.2.1, 1.4.1
Composite Model							✓	1.4.4, 1.4.5
Transfer Learning^{+}					✓	✓	✓	1.4.2, 1.4.4
Transfer Learning^{++}						✓		1.4.3, 1.4.4

Table 1.2 – *Our multi-fold contributions in Neural Topic Modeling for learning representations of text documents. The symbol ✓ signifies if a feature (or related section) is related to a chapter. Thus, we illustrate that how the introduction section (i.e., chapter 1) is related to the rest of the chapters (i.e., our publications). Here, TM: topic modeling; Transfer Learning^{+}: Using word embeddings only; Transfer Learning^{++}: Using both the word embeddings and latent topics from many sources.*

1.2 Supervised Neural Networks

Machine Learning is a branch of artificial intelligence (AI) that aims at building intelligent systems with an ability to automatically learn from data, identify patterns and make decisions with minimal human involvement.

In context of machine learning, there are two main paradigms of learning: *supervised* and *unsupervised*. The difference lies in how they use prior knowledge, i.e., ground truth signals. Supervised learning makes use of the ground truth values for samples and aims to learn a mapping function that best approximates the relationship between the inputs and outputs in the data. In contrast, unsupervised learning focuses on learning the inherent structure of data without having the explicitly-provided output labels. Supervised learning is often applied to classification or regression tasks, whereas unsupervised learning to clustering, representation learning, dimensionality reduction, density estimation, etc. Common algorithms in supervised learning are: logistic regression, naive bayes, support vector machines, artificial neural networks, random forests, etc. On other hand, unsupervised learning algorithms include k-means clustering, principal component analysis, autoencoders, restricted Boltzmann machines, etc. *In context of this dissertation, we mainly focus on supervised and unsupervised learning in neural architectures, applied to text data.*

Though explaining how the human brain learns is quite difficult, the artificial neural networks (ANNs) attempt to simulate the human brain's ability to learn. ANNs are composed of neurons and connections (weights) between them, where they adjust the weights based on an error signal (or feedback) during the learning process to find a desired output for a given input. The learning algorithm of a neural network can either be supervised or unsupervised.

In 1943, Warren MuCulloch (neuroscientist) and Walter Pitts (logician) proposed a highly simplified and the first computational model of a neuron, where they made attempts to understand how the brain produces highly complex patterns by using many basic cells (or neurons) that are connected together. Neural network theory is founded on their McCulloch-Pitts model (McCulloch and Pitts, 1943; Piccinini, 2004). The next major advancement was the perceptron (Rosenblatt, 1958), introduced by Frank Rosenblatt in 1958.

Feed-forward Neural Networks

Feed-forward Neural Networks or *Multi-layer Perceptrons* (MLPs) consist of several perceptrons arranged in layers, with the first layer taking in inputs and the last layer producing outputs. The middle layers capture relationship between the input and output with no exposure to external world, therefore they are called hidden layers. In such network structures, information flows from one layer to

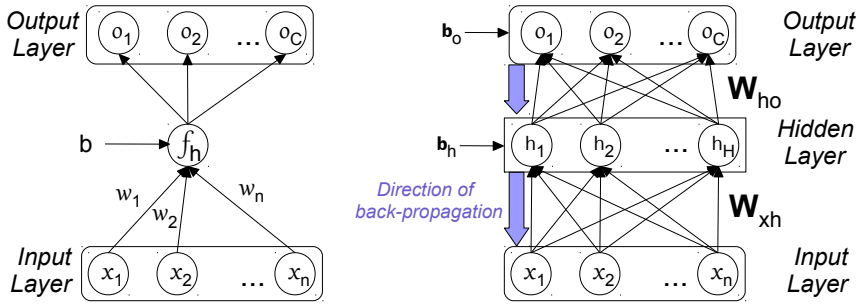


Figure 1.1 – *(left) An example of a neuron showing the input* $(x_1, ..., x_n)$*, their corresponding weights* $(w_1, ..., w_n)$*, a bias* b *and activation function* f_h *applied to weighted sum of the inputs. (right) A feed-forward Neural Network or Multi-layer Perceptron (MLP) with three layers, where the input and hidden layers are connected by weight matrix* $\mathbf{W}_{xh}$*, and the hidden and output layers by weight matrix* $\mathbf{W}_{ho}$*. Here, the* n*,* H *and* C *are the number of input, hidden and output dimensions (or units). The vectors* $\mathbf{b}_h \in \mathbb{R}^H$ *and* $\mathbf{b}_o \in \mathbb{R}^C$ *are biases of hidden and output layer, respectively.*

the next (e.g., input layer $\rightarrow$ hidden layer(s) $\rightarrow$ output layer), hence the name *feed-forward*.

Figure 1.1 (left) provides an illustration of a single perceptron that can classify points into two regions that are linearly separable. However, Figure 1.1 (right) shows an MLP that can be applied to model relationship between an input and output, where the input data points are not linearly separable. Observe that there are no feedback connections among perceptron units in the same layer.

Training Feed-forward Neural Networks: In the realm of supervised learning, the training comprises three steps: (1) `Forward-propagation` of information from an input through an output layer via a hidden layer(s) to compute the value of loss (or error minimization) function, (2) `Backward-propagation` (Kelley, 1960; Rumelhart et al., 1988) of errors from an output to an input layer via a hidden layer(s) in direction opposite of the forward-propagation, and (3) `Parameter updates` based on the feedback computed via training errors during back-propagation. The weights (e.g., $\mathbf{W}_{xh}, \mathbf{W}_{ho}$) and biases ($\mathbf{b}_h$ and $\mathbf{b}_o$) are adjusted so as to minimize the training errors.

Gradient-based methods (Rumelhart et al., 1985; Sutton, 1986; Amari et al., 2000) are the key tools in minimization of the error function. See Ruder (2016) for an overview about the gradient-based optimization.

Limitations of Feed-forward Networks: Feed-forward Neural Networks form the basis of many important neural network architectures of the recent times, such as Convolutional Neural Networks (CNNs) (LeCun et al., 2004; Krizhevsky et al., 2012), Recurrent Neural Networks (RNNs) (Rumelhart et al., 1988; Elman,

1990), Long-short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997), Recursive Neural Networks (RecvNN) (Goller and Küchler, 1996; Socher et al., 2011b), Siamese Neural Networks (SNN) (Bromley et al., 1993; Chopra et al., 2005; Gupta et al., 2018a), etc. Though Feed-forward Neural Networks can represent complex function, they are not designed for sequence data because they do not account for memory and thus, do not remember historic input data. Sequential data needs a feedback mechanism to model relationships in data inputs over time. In order to better model sequential data, RNNs have been popular and shown success in many sequential tasks.

Distributed Representations in Neural Networks: Each neuron represents something that can be seen as an explanatory factor about the data. In isolation, a single neuron is a local representation. On other hand, many neurons come together to form a concept and each neuron participates in the representation of many concepts. It leads to *distributed representations* (Plate, 1995; Hinton, 1986) in the sense that the informational content is distributed among multiple units, and at the same time, each unit can contribute to the representation of multiple elements. For instance, the distributed representations (Le and Mikolov, 2014; Mikolov et al., 2013c) are vectors of real numbers representing the meaning of words, phrases, sentences and documents.

1.2.1 Recurrent Neural Network (RNN)

An RNN (Rumelhart et al., 1988; Elman, 1990), a class of supervised neural networks, remembers its past input every time a new input is fed using an internal memory and thus, models sequential information with feedback loops over time. It is 'recurrent' in the sense that it performs (or loops over) the same task for every element of a sequence, with the output dependent on previous computations.

Unlike Feed-forward neural network, an RNN models arbitrary length sequences in input and/or output, shares features learned across different time steps and captures relationships in the sequential input to account for the direction of information flow. These capabilities have made the RNN a popular neural architecture in solving sequential tasks, such as:
(1) speech recognition (Graves et al., 2013),
(2) video activity analysis (Donahue et al., 2017),
(3) caption generation (Yang et al., 2016),
(4) machine translation (Cho et al., 2014),
(5) language modeling[1] (Mikolov et al., 2010; Peters et al., 2018),
(6) named entity recognition (Ma and Hovy, 2016; Gupta et al., 2016),
(7) relation extraction (Vu et al., 2016a; Gupta et al., 2019c),

[1] the task of predicting the next word given the previous ones in a text

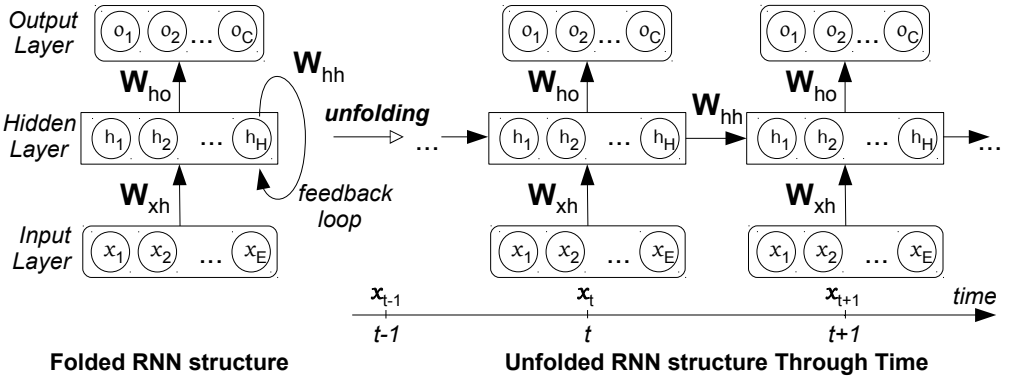


Figure 1.2 – *(left) Folded RNN with feedback loop to persist information. (right) Unfolded RNN structure through time.*

(8) textual similarity (Mueller and Thyagarajan, 2016; Gupta et al., 2018a)
(9) sentiment analysis (Tang et al., 2015b),
(10) text generation (Sutskever et al., 2011; Zhang and Lapata, 2014),
(11) music generation (Boulanger-Lewandowski et al., 2012),
(12) dynamic topic modeling (Gupta et al., 2018b), etc.

A Simple RNN Formulation: Essentially, an RNN persists information via a loop that takes information from previous time step and passes it to the input of current time step. When unfolded in time, an RNN can be seen as multiple copies of the same network, each passing a message to a successor.

Figure 1.2 shows a folded and an unfolded structure of an RNN through time t. Notice that a simple RNN consists of three layers: (1) input, (2) hidden and (3) output. The input layer takes a sequence $\mathbf{S}$ of vectors through time t, such that $\mathbf{S} = \{\mathbf{x}_1, ..., \mathbf{x}_t, ..., \mathbf{x}_T\}$ where the vector $\mathbf{x}_t = (x_1, ..., x_E)$; T and E are sequence length and input dimensions, respectively. Assume a sequence of words (e.g., a sentence) where a word is represented via an embedding vector (Mikolov et al., 2013b; Pennington et al., 2014), then $\mathbf{x}_t \in \mathbb{R}^E$ with E as the dimension of word embedding vector. The input units are connected to hidden units in the hidden layer by a input-to-hidden weight matrix $\mathbf{W}_{xh} \in \mathbb{R}^{H \times E}$. The hidden layers (e.g., $\mathbf{h_t} \in \mathbb{R}^H$) consist of H units, and are connected to each other through time by recurrent connections $\mathbf{W}_{hh} \in \mathbb{R}^{H \times H}$. Further, the output $\mathbf{o}_t = \{o_1, ..., o_C\}$ is attached to each of the hidden layers through time, and has C units, each defining the number of classes. Each of the hidden layers is connected to its output by weighted connections, $\mathbf{W}_{ho} \in \mathbb{R}^{C \times H}$.

Therefore, an RNN processes a sequence $\mathbf{S}$ of vectors by applying a recurrence at every time step t, as:

$$\mathbf{h}_t = g_h(\mathbf{W}_{xh} \cdot \mathbf{x}_t + \mathbf{W}_{hh} \cdot \mathbf{h}_{t-1} + \mathbf{b}_h) \tag{1.1}$$

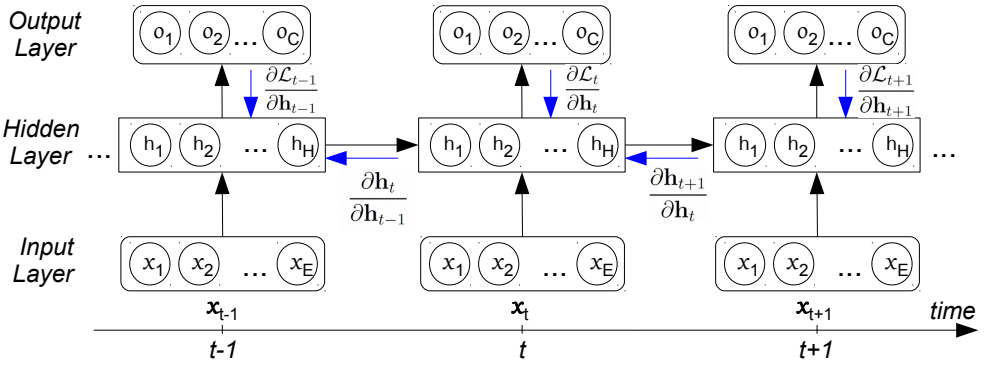


Figure 1.3 – *An illustration of Back-propagation Through Time (BPTT) in an unfolded RNN structure. The blue arrows indicate the gradient flow.*

where g_h is the hidden layer activation function (e.g., sigmoid, tanh, rectified linear unit, etc.), and $\mathbf{b}_h \in \mathbb{R}^H$ is the bias vector of the hidden units. Now, the output units are computed as:

$$\mathbf{o}_t = g_o(\mathbf{W}_{ho} \cdot \mathbf{h}_t + \mathbf{b}_o) \tag{1.2}$$

where g_o is the activation function (e.g., sigmoid, softmax, etc.), and $\mathbf{b}_o \in \mathbb{R}^C$ is the bias vector of the output layer.

Given that the input-output pairs are sequential in time, the above steps execute repeated over time $t \in \{1, ..., T\}$. Due to the feedback looping and internal memory, the hidden vector $\mathbf{h}_t$ can be seen as encoding the selective summarization of necessary information about the past states ($\mathbf{h}_{<t}$) of the network over many time-steps. In our work (Gupta and Schütze, 2018), we have shown how RNNs leverage inherent sequential patterns in relation classification and build semantics over a sequence of words.

RNNs (Figure 1.2) evaluate their performance by using a loss function $\mathcal{L}$ that compares the predicted output $\mathbf{o}_t$ with ground truth $\mathbf{z}_t$, as:

$$\mathcal{L}(\mathbf{o}, \mathbf{z}) = \sum_{t=1}^{T} \mathcal{L}_t(\mathbf{o}_t, \mathbf{z}_t) \tag{1.3}$$

where the summation is applied to overall loss at each of the time steps T. There are several choices for loss function such as cross-entropy over probability distribution of outputs for classification, Euclidean distance for real-values, Mean Squared Error (MSE), etc. Given the loss function, RNN parameters are optimized popularly using gradient descent in order to minimize prediction errors.

Training Recurrent Neural Network: Gradient descent (GD) is one of the popular optimization strategies in training neural network that computes derivatives of the loss function with respect to model parameters. Since an RNN is a

structure through time, therefore it is trained using back-propagation through time (BPTT) (Werbos, 1990). It is a generalization of GD, applied to feed-forward networks. Essentially, the BPTT propagates error signal backwards though time. See an illustration of BPTT in Figure 1.3.

Given RNN's parameters $\theta = \{\mathbf{W}_{xh}, \mathbf{W}_{hh}, \mathbf{W}_{ho}, \mathbf{b}_h, \mathbf{b}_o\}$ and hidden vector $\mathbf{h}_t$ at time step t, we can write the gradients of the loss as:

$$\frac{\partial}{\partial \theta}\mathcal{L}(\theta) = \sum_{t=1}^{T} \frac{\partial}{\partial \theta}\mathcal{L}_t(\theta) \tag{1.4}$$

where the loss $\mathcal{L}_t$ at time step t is further expanded using chain-rule:

$$\frac{\partial \mathcal{L}_t}{\partial \theta} = \sum_{1 \leq k \leq t} \Big(\frac{\partial \mathcal{L}_t}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} \frac{\partial \mathbf{h}_k}{\partial \theta} \Big) \tag{1.5}$$

The Jacobian $\frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k}$ propagates error backwards through time from time step t to k; therefore, $\mathbf{h}_t$ is dependent on the preceding hidden vectors $\mathbf{h}_{<t}$. To generalize, we can write the dependence for $\mathbf{h}_t$ as:

$$\frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} = \prod_{t \geq i > k} \frac{\partial \mathbf{h}_i}{\partial \mathbf{h}_{i-1}} = \prod_{t \geq i > k} \mathbf{W}_{hh}^T \text{diag}|g_h'(\mathbf{h}_{i-1})| \tag{1.6}$$

where g_h' is the derivative of the activation function g_h and diag is the diagonal matrix. Here, we define the *long term* and *short term* dependencies of hidden states over time, where the *long-term* refers to the contribution of the inputs and hidden states at time $k << t$, and *short-term* refers to the contribution of the inputs and hidden states at the other times, except $k << t$.

The above-mentioned formal treatment is partially based on the lecture material by Gupta (2019) and Pascanu et al. (2013). See Gupta (2019) for further details on Backpropagation through time (BPTT) in RNN.

Difficulty of Training RNNs: For a large sequence length, the repeated multiplications of the recurrent matrix $\mathbf{W}_{hh}$ is responsible for *exponential* decay or *explosion* of gradients. It is due to the fact that a product of $t - k$ real numbers can shrink to zero (i.e., *vanishing gradient*) or explode (i.e., *exploding gradient*) to infinity, as can the product of $t - k$ Jacobian matrices (equation 1.6).

The *vanishing* gradient causes internal memory of the network to ignore long-term dependencies and hardly learn the correlation between temporally distant events. This means that the RNN will tend to focus on short term dependencies which is often not desired. On other hand, the *exploding* gradient results in a large increase in the norm of the gradient during training. Therefore, it is difficulty to train RNNs to capture long-range temporal dependencies for longer sequences.

See Pascanu et al. (2013) and Gupta (2019)[2] for further details about the difficulty of training RNNs.

To deal with the exploding gradients, Pascanu et al. (2013) proposed a simple and popular approach of gradient norm-clipping coupled with a clipping threshold in training RNNs. The clipping scales down the gradients when their norm is greater than the clipping threshold. Additionally, using an L1 or L2 penalty on the recurrent weights can help.

On other hand, existing works such as Hochreiter and Schmidhuber (1997), Graves et al. (2009), Chung et al. (2014) and Chung et al. (2015) have proposed extensions of a simple RNN model using a gating mechanism to deal with the vanishing gradient problem. Additionally, several strategies (Martens and Sutskever, 2012; Mikolov et al., 2014; Le et al., 2015) of initializing recurrent matrix such as with identity and using rectifier linear units have shown that initialization plays an important role in training RNNs.

Popular RNN extensions: Beyond traditional uni-direction RNNs, Schuster and Paliwal (1997) have proposed bi-directional RNNs that consider all available input sequences in both the past and future for estimation of the output vector, instead of only using the previous context. They model the sequential data using two networks: a forward and a backward pass in time. A Recurrent Convolutional Neural Network (RCNN) (Liang and Hu, 2015) combines the complementary learning in RNN and Convolutional neural network (CNN) (LeCun et al., 2004; Krizhevsky et al., 2012). Hochreiter and Schmidhuber (1997) proposed one of the popular RNN models, named as Long-short Term Memory (LSTM) that reduces the effects of vanishing and exploding gradients. Recently, Lample et al. (2016) applied a bidirectional LSTM with a Conditional Random Field (CRF) (Lafferty et al., 2001) layer (LSTM-CRF) to sequence tagging tasks, such as named-entity recognition, port-of-speech tagging, etc. Moreover, Sutskever et al. (2014) presented a general approach of modeling sequences that can be applied to sequences having variable length in input and output.

Our Contribution: In our efforts to extend RNNs for multi-tasking (Caruana, 1997), we present a novel neural network architecture, which we named as *Table-fiiling Multi-task RNN* (TF-MTRNN) (Gupta et al., 2016) that jointly learns entities and relations within a sentence boundary.

Limitations of Vanilla RNN: Theoretically, RNNs can make use of information in arbitrarily long sequences, but empirically, they are limited to looking back only a few steps. Therefore, they can not model long-term dependencies. Essentially[3] to prevent gradients from vanishing or exploding (Pascanu et al., 2013),

[2] www.researchgate.net/publication/329029430_Lecture-05_Recurrent_Neural_Networks_Deep_Learning_AI

[3] www.dbs.ifi.lmu.de/Lehre/DLAI/WS18-19/script/05_rnns.pdf

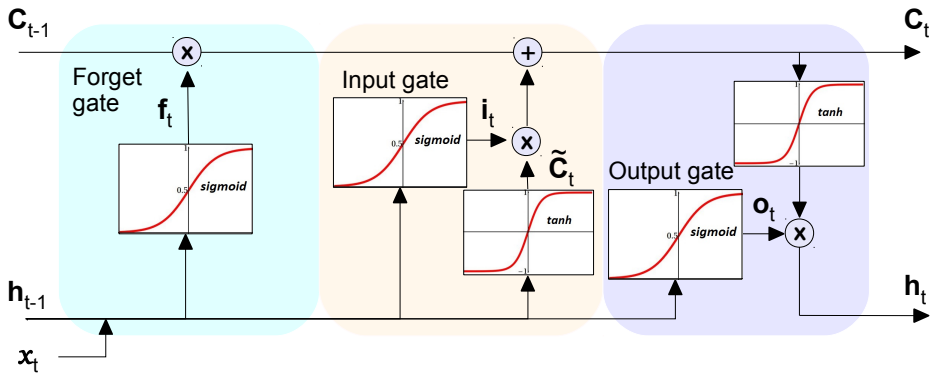


Figure 1.4 – *A Long Short-Term Memory (LSTM) Cell at time step t*

there is a need for tight conditions on eigenvalues of the recurrent matrix during training.

Long Short-Term Memory (LSTM)

LSTM (Hochreiter and Schmidhuber, 1997) networks are extensions of RNNs that reduce the effects of vanishing and exploding gradients. They used gating mechanism to control memory cells, instead of hidden units from *sigmoid* or *tanh*, where the gating is a way to optionally let information through and composed of a *sigmoid* layer and point-wise multiplication operation. The cells transport information through units and gates allow the flow of information to hidden neurons as well as remember information from previous step (i.e., remove or add information to the cell state). As a result, LSTMs propagate errors for much longer than ordinary RNNs and therefore, can exploit long range dependencies in the data.

In principle, an LSTM[4] (Hochreiter and Schmidhuber, 1997) creates a self loop path from where gradient can flow and the self loop corresponds to an eigenvalue of Jacobian to be slightly less[5] than 1, such that $\partial newState / \partial oldState \approx Identity$.

LSTM Formulation: Figure 1.4 illustrates an LSTM cell at time step t that consists of three gates: Forget, Input and Output. We discus the mechanics below:

1. The *forget gate* $\mathbf{f}_t$ using *sigmoid* layer decides what information to keep or throw away from the previous cell state $\mathbf{C}_{t-1}$. It takes in the input $\mathbf{x}_t$ of current time step t and hidden state $\mathbf{h}_{t-1}$ of previous step, and outputs

[4] `www.iro.umontreal.ca/~bengioy/talks/DL-Tutorial-NIPS2015.pdf`

[5] If an eigenvalue of Jacobian is $\gg 1$, the gradients explode. If an eigenvalue of Jacobian is $\ll 1$, the gradients vanish. See Gupta (2019) or `www.dbs.ifi.lmu.de/Lehre/DLAI/WS18-19/script/05_rnns.pdf` for the necessary condition for exploding gradient and the sufficient condition for vanishing gradients on eigenvalues of the Jacobian.

a number between 0 and 1 corresponding to each number in the cell state $\mathbf{C}_{t-1}$. Therefore, it learns weights to control information decay.

$$\mathbf{f}_t = \text{sigmoid}(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f) \tag{1.7}$$

2. On other hand, the *input gate* $\mathbf{i}_t$ selectively updates the cell state $\mathbf{C}_t$ based on the new input, where the *sigmoid* layer decides which values to update. Additionally, A *tanh* layer creates candidate values $\tilde{\mathbf{C}}_t$ to be selected to include to the cell state. Next, $\mathbf{i}_t$ and $\tilde{\mathbf{C}}_t$ are combined to update $\mathbf{C}_t$.

$$\begin{aligned} \mathbf{i}_t &= \text{sigmoid}(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i) \\ \tilde{\mathbf{C}}_t &= \tanh(\mathbf{W}_{xg}\mathbf{x}_t + \mathbf{W}_{hg}\mathbf{h}_{t-1} + \mathbf{b}_g) \end{aligned} \tag{1.8}$$

3. The old cell state $\mathbf{C}_{t-1}$ is now updated into a new one $\mathbf{C}_t$ as:

$$\mathbf{C}_t = \mathbf{f}_t * \mathbf{C}_{t-1} + \mathbf{i}_t * \tilde{\mathbf{C}}_t \tag{1.9}$$

4. Finally, the *output gate* $\mathbf{o}_t$ takes in the current input $\mathbf{x}_t$, previous hidden state $\mathbf{h}_{t-1}$ and current cell state $\mathbf{C}_t$, and regulates the amount of information from the cell state $\mathbf{C}_t$ that goes into hidden state $\mathbf{h}_t$.

$$\begin{aligned} \mathbf{o}_t &= \text{sigmoid}(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o) \\ \mathbf{h}_t &= \mathbf{o}_t * \tanh(\mathbf{C}_t) \end{aligned} \tag{1.10}$$

Limitations of LSTMs: Though LSTMs have been successful in modeling sequential data, they suffer from higher memory requirements due to an increase (4 times) in the number of parameters compared to a simple RNN. There is even a higher computational complexity due to multiple memory cells introduced in LSTMs.

Gated Recurrent Unit (GRU)

Like the LSTM, a GRU (Chung et al., 2014) deals with the vanishing gradient problem in RNNs via gating mechanism. It combines the forget and input gates of LSTM into a single *update gate*. Moreover, the cell state and hidden states are merged into a single memory content. In a way, the GRU does not have a controlled exposure of the memory content to other units in the network. It is different to LSTM due to an output gate with a controlled exposure. Though, the resulting GRU model is simpler than a standard LSTM, it has shown competitive performance and gained popularity.

See Chung et al. (2014) that outlined several similarities and differences between GRU and LSTM networks.

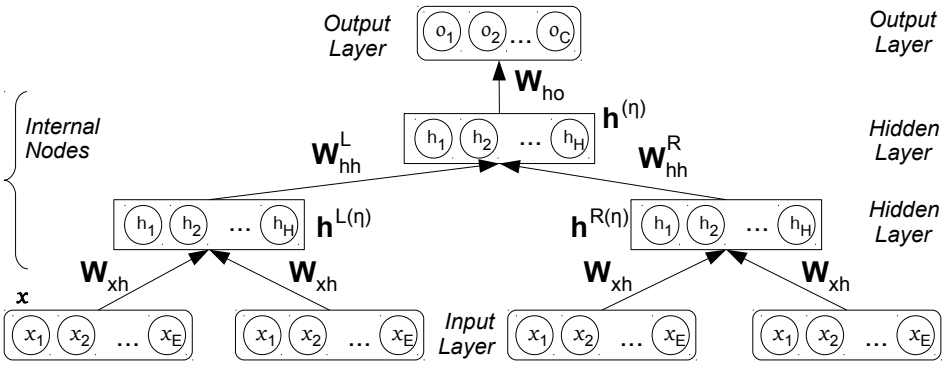


Figure 1.5 – *A Recursive Neural Network (RecvNN) architecture. Here, η indicates a current node, $L(\eta)$ and $R(\eta)$ are the left and right children of η in the tree structure. The leaf nodes consist of word vectors $\mathbf{x} \in \mathbb{R}^E$.*

1.2.2 Recursive Neural Network (RecvNN)

A Recursive Neural Network (Goller and Küchler, 1996; Socher et al., 2011b, 2013), a generalization of Recurrent Neural Network operates on a structured input, i.e., tree structure instead of a sequence. RecvNN expects a fixed number of branches within a tree structure and recursively computes parent representations in a bottom-up fashion, by combining child nodes. The computation is sequentially calculated from the leaf nodes toward the root node.

Recursive Neural Network Formulation: Figure 1.5 demonstrates an architecture of TreeRNN that operates on binary tree structure with word vector ($\mathbf{x} \in \mathbb{R}^E$) representations at the leaves and hidden vectors $\mathbf{h}$ as internal nodes. For a current node η, the hidden (internal) vector $\mathbf{h}^{(\eta)}$ of a current node η is computed from the hidden vectors ($\mathbf{h}^{L(\eta)}$ and $\mathbf{h}^{R(\eta)}$) of its left $L(\eta)$ and right $R(\eta)$ child nodes, as:

$$\mathbf{h}^{(\eta)} = f_H(\mathbf{W}^{L(\eta)}\mathbf{h}^{L(\eta)} + \mathbf{W}^{R(\eta)}\mathbf{h}^{R(\eta)} + \mathbf{b}) \tag{1.11}$$

where f_H is an activation function (e.g., sigmoid, tanh, etc.) for hidden layer. Notice that when η is a leaf node then, $\mathbf{h}^{L(\eta)} = \mathbf{x}^{L(\eta)}$, $\mathbf{h}^{R(\eta)} = \mathbf{x}^{R(\eta)}$, $\mathbf{W}^{L(\eta)} = \mathbf{W}^L_{xh}$, $\mathbf{W}^{R(\eta)} = \mathbf{W}^R_{xh}$ and $\mathbf{b} = \mathbf{b}_l \in \mathbb{R}^E$. Here, $\mathbf{x} \in \mathbb{R}^E$ is an input word vector. On other hand, when η is not a leaf node then, $\mathbf{W}^{L(\eta)} = \mathbf{W}^L_{hh}$, $\mathbf{W}^{R(\eta)} = \mathbf{W}^R_{hh}$ and $\mathbf{b} = \mathbf{b}_h \in \mathbb{R}^H$. Here, the weights $\mathbf{W}^L_{xh}$ and $\mathbf{W}^R_{xh}$ connect a leaf (e.g., word embedding) to its hidden vectors, whereas $\mathbf{W}^L_{hh}$ and $\mathbf{W}^R_{xh}$ are the weight matrices that connect hidden vectors of left and right children respectively to the parent vector representation. The vectors $\mathbf{b}_h$ and $\mathbf{b}_l$ are biases for hidden and input layers, respectively. A recursive computation is performed from each of the leaves up to the root node.

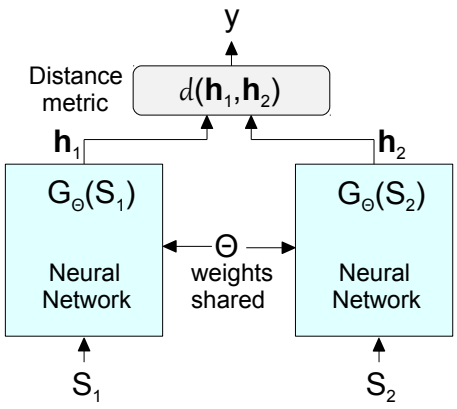


Figure 1.6 – *A Siamese Neural Network architecture.*

Based on the task, an output layer can be attached to each of the hidden vectors in the tree structure to perform a classification for the node η.

$$\mathbf{o}^{(\eta)} = f_o(\mathbf{W}_{oh}\mathbf{h}^{(\eta)} + \mathbf{b}_o) \tag{1.12}$$

Here, f_o is an activation function (e.g., sigmoid or softmax) for output layer in case of classification task. In Figure 1.5, a output layer is attached to the root only, suggesting a sentence-level classification task.

One of the key **benefits** of such topological composition order is that RecvNNs can express relationships between long-distance elements compared to RNNs, because the depth is logarithmic in N if the element count is N. Thus, they have been applied to scene parsing (Socher et al., 2011b), sentiment analysis (Socher et al., 2013), paraphrase detection (Socher et al., 2011a), dependency parsing (Kiperwasser and Goldberg, 2016), relation extraction (Liu et al., 2015b; Miwa and Bansal, 2016; Zhang et al., 2018), etc.

Our Contribution: In the realm of relation extraction within and across sentence boundary, we present a novel *inter-sentential Dependency-based Neural Network* (iDepNN) (Gupta et al., 2019c) that essentially combines the recurrent and recursive networks over dependency parse features to extract long-distant relationships between entities, spanning sentence boundary.

1.2.3 Siamese Neural Network (SNN)

Siamese neural network (Bromley et al., 1993; Chopra et al., 2005) is a class of supervised neural networks that employs a unique structure to learn similarity in a pair of inputs. It is a dual-branch network with tied (or shared) weight parameters and an objective function, i.e., a distance metric to learn similarity/dissimilarity between feature representations of the distinct input pairs on each side. Also,

the twin network is symmetric in the sense that whenever we present two distinct inputs (e.g., image or text pairs), the top conjoining layer will compute the same metric as if we were to present the same two inputs but to the opposite twins.

When applied to textual pairs, the aim of training is to learn text pair representations to form a highly structured space, where they reflect complex semantic relationships.

Siamese Neural Network Formulation: Figure 1.6 illustrates a Siamese architecture, where a neural network, e.g., RNN, LSTM, CNN, etc. can be applied to each of the two branches. The aim of the neural networks is to find a function G_Θ that can generate feature representations and map input into a target space, such that a distance metric d in the target space approximates the "semantic" distance in the input space. In doing so, the learning is performed by finding the θ (being shared) that minimizes a loss function, evaluated over a training set.

Assume S_1 and S_2 are two input sentence and G_Θ is a feature generator (e.g., RNN, LSTM, CNN, etc.) within a neural network framework. The twin network shares Θ and generates $\mathbf{h}_1$ and $\mathbf{h}_2$ mapped in a features space from each of the branches, where the distance function d is small if S_1 and S_2 belong to the same category or similar, and large otherwise. y measures the compatibility between S_1 and S_2, i.e., a category in case of classification or a similarity/dissimilarity score.

Several mapping function G_Θ have been investigated. Turk and Pentland (1991) applied a PCA-based method and Yang et al. (2000) outlined non-linear extensions using Kernel-PCA and Kernel-LDA. Further, Chopra et al. (2005) employed CNN-based feature generators in an application to face verification. To assess semantic similarity between sentences, Mueller and Thyagarajan (2016) and Yin et al. (2016) applied LSTM and CNN networks to encode the underlying meaning expressed in sentence pairs.

Our Contribution: We present a Siamese-based neural architecture, which we named as *Replicated Siamese* (Gupta et al., 2018a), applied to an industrial ticketing system to learn similarity in textual pairs beyond sentences.

1.3 Unsupervised Neural Density Estimators

Given a set of data points, probability density estimation is the task of reconstructing the probability density function that explains underlying structure in the input multivariate data distribution. From the probabilistic modeling perspective, it is a process to recover a set of model parameters for a generative neural network (i.e., density estimator) such that it can describe the distribution underlying the observed high-dimensional data. Essentially in doing so, the density estimator can learn interesting explanatory factors in the underlying data by projecting it in a latent space while retaining maximal variations in the data. The quality

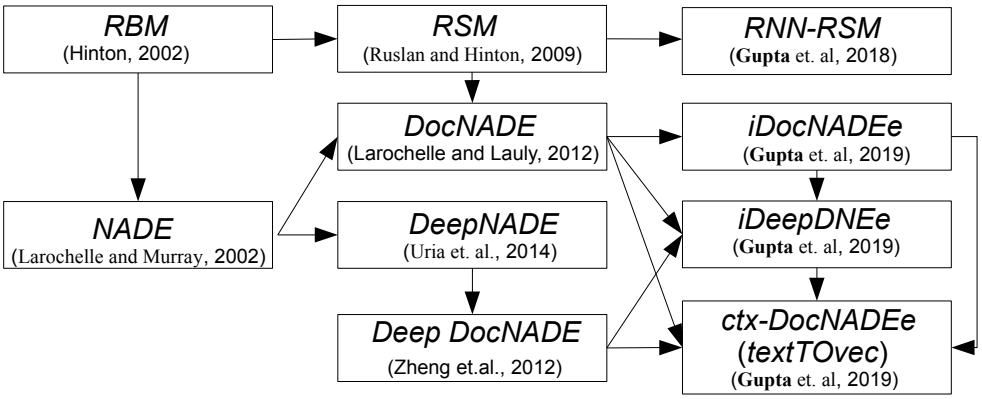


Figure 1.7 – *RBM and NADE evolution with our contributions*

of explanatory factors captured in such generative models is measured by their predictive data-likelihood in the sense that the probability density function of the model is as close as possible to the data distribution.

In the following section, we discuss several probability density estimators, especially Restricted Boltzmann Machine (RBM) (Smolensky, 1986) and its extensions in an unsupervised neural network setting. Table 1.3 defines the notations used in describing the different density estimators.

Our Contribution(s): Figure 1.7 summarizes the evolution of neural density estimators, as discussed in the context of this dissertation. It includes the density estimators of binary as well as count (i.e., text) data, where the four rightmost rectangular boxes (i.e., *RNN-RSM, iDocNADEe, iDeepDNEe* and *ctx-DocNADEe*) signify our research contributions in the realm of document topic modeling.

1.3.1 Restricted Boltzmann Machine (RBM)

Restricted Boltzmann Machine (Smolensky, 1986; Freund and Haussler, 1992; Hinton, 2002) is a type of two-layer neural network, consisting of stochastic units with undirected interactions between pairs of observed visible (input) $\mathbf{V}$ and unobserved hidden $\mathbf{H}$ units. The visible and hidden units (or neurons) are binary, which can be seen as being arranged in two layers. The visible units form the first layer and represent the observable data (e.g., one visible unit for each pixel of a digital input image). The hidden units model dependencies between the components of observations (e.g., dependencies between pixels in images) and are inferred from the visible units.

An RBM is an undirected graphical and generative model representing a probability distribution underlying the training data. Given the training data, an RBM learns to adjust its parameters in a stochastic manner via iterative forward and

Notation	Data Type	Description
$\mathbf{v}$	$\{0,1\}^D$	Binary visible units
$\mathbf{h}$	$\mathbb{R}^H$	Hidden units
$\mathbf{V}$	$\{0,1\}^{K\times D}$	An observed binary matrix in *RSM*
K	$\mathbb{I}$	Vocabulary size
N	$\mathbb{I}$	Number of documents in a corpus
D	$\mathbb{I}$	Number of visible units; document size in *DocNADE*
H	$\mathbb{I}$	Number of units (i.e., dimension) in a hidden layer
Z	$\mathbb{R}$	Partition function
E	$\mathbb{R}$	Energy of the model
$\mathbb{E}$	-	Expectation
$\mathbf{b}$	$\mathbb{R}^D$	Visible bias
$\mathbf{c}$	$\mathbb{R}^H$	Hidden bias
$\mathbf{W}$	$\mathbb{R}^{D\times H}$	Weights connecting visible-to-hidden layers in *RBM* and *NADE*
$\mathbf{W}'$	$\mathbb{R}^{H\times D}$	Weights connecting hidden-to-visible layer in *NADE*
$\mathbf{W}$	$\mathbb{R}^{H\times K}$	Weights connecting visible-to-hidden layers in *RSM* and *DocNADE*
$\mathbf{U}$	$\mathbb{R}^{K\times H}$	Weights connecting hidden-to-visible layers in *DocNADE*
$\mathbf{\Theta}$	$\{\}$	A set of parameters
$bold{+}lowercase$	-	A vector
$bold{+}uppercase$	-	A matrix

Table 1.3 – *Notations used in the unsupervised density estimators*

backward passes between hidden and visible layers, such that the probability distribution represented by the RBM fits the training data.

The RBM is a special type of Boltzmann Machine (BM) (Ackley et al., 1985) without lateral connections where the pair interactions $\mathbf{I}$ are *restricted* to be between visible and hidden units, i.e., $\mathbf{I} = \{\{i,j\} : i \in \mathbf{v}, j \in \mathbf{h}\}$. Each unit (or neuron) can take one of the two states, i.e., either 0 or 1, where the joint state of all the units is defined by an associated energy. Figure 1.8 provides graphical illustrations of a BM and an RBM.

RBMs belong to Energy Based Models (EBMs) (Hopfield, 1982) that capture dependencies between observed and unobserved units through modifying an energy function determined by the pair interactions of visible and hidden states, weights and biases. A high energy configuration/state implies a bad compatibility in configuration/pairwise interactions of the units. Therefore, an RBM learning corresponds to minimizing a predefined energy function, and the states of the units are updated in a stochastic manner.

The following formulation of an RBM is partially based on Bengio (2009).

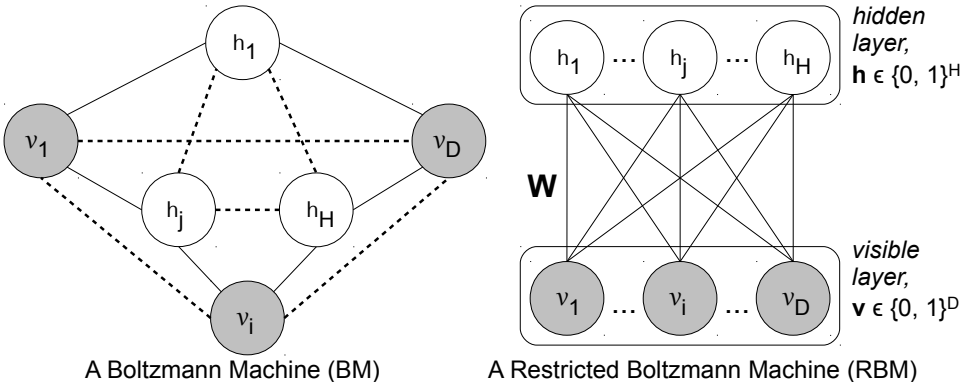


Figure 1.8 – *(left) A Boltzmann Machine (BM), where the dashed lines indicate visible-visible or hidden-hidden connections. (right) A Restricted Boltzmann Machine (RBM), an undirected graphical model based on a bipartite graph with the pair connections between visible and hidden units, while visible-visible and hidden-hidden connections (dashed lines in BM) are not allowed (i.e., restricted). Each visible (or hidden) unit is arranged into a visible (or hidden) layer and therefore, an RBM is seen as a 2-layered neural network with symmetric connections via* $\mathbf{W}$ *in visible-visible and hidden-hidden units.*

Formulation of an RBM

In a binary RBM of D visible units $\mathbf{v} = (v_1, ..., v_D)$ and n hidden units $\mathbf{h} = (h_1, ..., h_H)$, the random variables $(\mathbf{v}, \mathbf{h})$ take values $(\mathbf{v}, \mathbf{h}) \in \{0, 1\}^{D+H}$ and the joint probability distribution $p(\mathbf{v}, \mathbf{h})$ under the model is related to an energy function $E(\mathbf{v}, \mathbf{h})$ and is given by:

$$p(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp(-E(\mathbf{v}, \mathbf{h})) \tag{1.13}$$

where Z is the partition function (normalization constant):

$$Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} \exp(-E(\mathbf{v}, \mathbf{h})) \tag{1.14}$$

and the energy function $E(\mathbf{v}, \mathbf{h})$ takes the following form in order to model the relationship between the visible and hidden variables:

$$E(\mathbf{v}, \mathbf{h}) = -\mathbf{b}^T\mathbf{v} - \mathbf{c}^T\mathbf{h} - \mathbf{v}^T\mathbf{W}\mathbf{h} \tag{1.15}$$

where $\mathbf{W} \in \mathbb{R}^{D \times H}$ is the weight matrix connecting hidden and visible units, and $\mathbf{b}$ and $\mathbf{c}$ are biases of the visible and hidden layers, respectively.

Intractability: We can observe that the partition function Z is computed by exhaustively summing over all states and therefore, it would be computationally intractable implying that the joint probability distribution $p(\mathbf{v})$ is also intractable.

Factorial Distributions via Conditionals: Due to the bipartite graph[6] structure of RBMs, the visible and hidden units are conditionally independent given one-another, i.e., its conditional distributions $p(\mathbf{v}|\mathbf{h})$ and $p(\mathbf{h}|\mathbf{v})$ are factorial and tractable, i.e, easy to compute.

Given the joint distribution (equation 1.13) and associated energy function (equation 1.15), the conditional distribution on the visible units $\mathbf{v}$ is computed as:

$$\begin{aligned} p(\mathbf{h}|\mathbf{v}) &= \frac{p(\mathbf{v},\mathbf{h})}{p(\mathbf{v})} \\ &= \frac{1}{p(\mathbf{v})}\frac{1}{Z}\exp(\mathbf{b}^T\mathbf{v} + \mathbf{c}^T\mathbf{h} + \mathbf{v}^T\mathbf{W}\mathbf{h}) \\ &= \frac{1}{Z'}\exp(\mathbf{c}^T\mathbf{h} + \mathbf{v}^T\mathbf{W}\mathbf{h}) \\ &= \frac{1}{Z'}\exp(\sum_{j=1}^{H} c_j h_j + \sum_{j=1}^{H} \mathbf{v}^T\mathbf{W}_{:,j}h_j) \\ &= \frac{1}{Z'}\prod_{j=1}^{H}\exp(c_j h_j + \mathbf{v}^T\mathbf{W}_{:,j}h_j) \end{aligned} \tag{1.16}$$

Due to the factorial nature of the conditionals, we can write the joint probability over the vector h as the product of (unnormalized) distributions over the individual elements, h_j:

$$p(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^{H} p(h_j|\mathbf{v}) \tag{1.17}$$

$$p(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{D} p(v_i|\mathbf{h}) \tag{1.18}$$

Here, each unit i or j is turned ON with probability $p(v_i = 1|\mathbf{h})$ or $p(h_j = 1|\mathbf{v})$ and turned OFF with probability $1 - p(v_i = 1|\mathbf{h})$ or $1 - p(h_j = 1|\mathbf{v})$. In other words, an RBM tries to find an optimal configuration in the network via paired visible-hidden connections such as positively paired connections seek to share the same state (i.e., be both ON or OFF), while the negatively paired connections prefer to be in different states.

[6] a graph whose vertices can be divided into two disjoint and independent sets such that no two graph vertices within the same set are adjacent, i.e., every edge connects a vertex in one set to a vertex in another set.

Now, normalizing the distributions over individual binary h_j as:

$$\begin{aligned} p(h_j = 1|\mathbf{v}) &= \frac{p(h_j = 1|\mathbf{v})}{p(h_j = 0|\mathbf{v}) + p(h_j = 1|\mathbf{v})} \\ &= \frac{\exp(c_j + \mathbf{v}^T \mathbf{W}_{:,j})}{\exp(0) + \exp(c_j + \mathbf{v}^T \mathbf{W}_{:,j})} \\ &= \text{sigmoid}(c_j + \mathbf{v}^T \mathbf{W}_{:,j}) \end{aligned} \tag{1.19}$$

Therefore, the full conditional over the hidden layer is expressed as the factorial distribution:

$$p(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^{H} \text{sigmoid}(c_j + \mathbf{v}^T \mathbf{W}_{:,j}) \tag{1.20}$$

Similarly, the conditional distribution on the hidden units $\mathbf{h}$ is given by:

$$p(\mathbf{v}|\mathbf{h}) = \frac{1}{Z''} \prod_{i=1}^{D} \exp(b_i v_i + v_i \mathbf{W}_{i,:} \mathbf{h}) \tag{1.21}$$

and the full conditional over the visible layer is expressed as the factorial distribution:

$$p(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{D} \text{sigmoid}(b_i + \mathbf{W}_{i,:} \mathbf{h}) \tag{1.22}$$

Due to the factorial nature of the conditionals, one can efficiently draw samples from the joint distribution via a block Gibbs sampling[7] strategy (Casella and George, 1992). For instance, each iteration of block Gibbs sampling (Markov chain) consists of two steps: (1): Sample $\mathbf{h}^{(l)} \sim p(\mathbf{h}|\mathbf{v}^{(l)})$ and (2): Sample $\mathbf{v}^{(l+1)} \sim p(\mathbf{v}|\mathbf{h}^{(l)})$. We can simultaneously and independently sample from all the elements of $\mathbf{h}^{(l)}$ given $\mathbf{v}^{(l)}$ and $\mathbf{v}^{(l+1)}$ given $\mathbf{h}^{(l)}$, respectively.

Training Restricted Boltzmann Machines

During training, an RBM learns to reconstruct the data in an unsupervised fashion and therefore, iteratively makes several forward and backward passes between the visible and hidden layer to adjust its parameters. In the reconstruction phase, the distance (measured by Kullback-Leibler Divergence) between its estimated probability distribution and the ground-truth distribution of the input is minimized,

[7] The idea is to generate posterior samples by sweeping through each variable (or block of variables) to sample from its conditional distribution with the remaining variables fixed to their current values. Further details: `https://ermongroup.github.io/cs323-notes/probabilistic/gibbs/`

i.e., the log-likelihood of the data (loss or cost) under the RBM with parameters $\Theta = \{\mathbf{b}, \mathbf{c}, \mathbf{W}\}$ is maximized and is given by:

$$\begin{aligned}
\mathcal{L}(\mathbf{W}, \mathbf{b}, \mathbf{c}) &= \sum_{k=1}^{N} \log(p(\mathbf{v}^k)) \\
&= \sum_{k=1}^{N} \log \sum_{\mathbf{h}} p(\mathbf{v}^k_{k,:}, \mathbf{h}) \\
&= \Big(\sum_{k=1}^{N} \log \sum_{\mathbf{h}} \exp\big(- E(\mathbf{v}^k, \mathbf{h})\big)\Big) - N \log Z \\
&= \Big(\sum_{k=1}^{N} \log \sum_{\mathbf{h}} \exp\big(- E(\mathbf{v}^k, \mathbf{h})\big)\Big) - N \log \sum_{\mathbf{v},\mathbf{h}} \exp\big(- E(\mathbf{v}, \mathbf{h})\big)
\end{aligned} \tag{1.23}$$

where N is the number of training examples. Here, the gradient of the log-likelihood with respect to the model parameters Θ is given as:

$$\begin{aligned}
\frac{\partial}{\partial \Theta}\mathcal{L}(\Theta) &= \frac{\partial}{\partial \Theta}\Big(\sum_{k=1}^{N} \log \sum_{\mathbf{h}} \exp\big(- E(\mathbf{v}^k, \mathbf{h})\big)\Big) - N \frac{\partial}{\partial \Theta} \log \sum_{\mathbf{v},\mathbf{h}} \exp\big(- E(\mathbf{v}, \mathbf{h})\big) \\
&= \sum_{k=1}^{N} \frac{\sum_{\mathbf{h}} \exp\big(- E(\mathbf{v}^k, \mathbf{h})\big) \frac{\partial}{\partial \Theta}\big(- E(\mathbf{v}^k, \mathbf{h})\big)}{\sum_{\mathbf{h}} \exp\big(- E(\mathbf{v}^k, \mathbf{h})\big)} \\
&\qquad - N \frac{\log \sum_{\mathbf{v},\mathbf{h}} \exp\big(- E(\mathbf{v}, \mathbf{h})\big) \frac{\partial}{\partial \Theta}\big(- E(\mathbf{v}, \mathbf{h})\big)}{\log \sum_{\mathbf{v},\mathbf{h}} \exp\big(- E(\mathbf{v}, \mathbf{h})\big)} \\
&= \underbrace{\sum_{k=1}^{N} \mathbb{E}_{p(\mathbf{h}|\mathbf{v}^k)}\Big[\frac{\partial}{\partial \Theta} - E(\mathbf{v}^k, \mathbf{h})\Big]}_{\text{the data-driven term}} - \underbrace{N\, \mathbb{E}_{p(\mathbf{v},\mathbf{h})}\Big[\frac{\partial}{\partial \Theta} - E(\mathbf{v}, \mathbf{h})\Big]}_{\text{the model-driven term}}
\end{aligned} \tag{1.24}$$

As we can see that the gradient of the log-likelihood $\frac{\partial}{\partial \Theta}\mathcal{L}(\Theta)$ is written as the difference between the two expectation terms of the gradient of the energy function: (1) the *data-driven term*, the expectation with respect to the product of the empirical distribution over the data, $p(\mathbf{v}) = \frac{1}{N}\sum_{k=1}^{N} \delta(x - \mathbf{v}^k)$ and the conditional distribution $p(\mathbf{h}|\mathbf{v}^k)$; (2) the *model-driven term*, the expectation with respect to the joint model distribution, p(**v**, **h**).

Using equation 1.15, we expand the energy term E and compute its gradient with respect to $\mathbf{W}$, $\mathbf{b}$ and $\mathbf{c}$ as:

$$\frac{\partial}{\partial \mathbf{W}}\big(-E(\mathbf{v},\mathbf{h})\big) = \frac{\partial}{\partial \mathbf{W}}\big(\mathbf{b}^T\mathbf{v} + \mathbf{c}^T\mathbf{h} + \mathbf{v}^T\mathbf{W}\mathbf{h}\big) = \mathbf{h}\mathbf{v}^T \tag{1.25}$$

$$\frac{\partial}{\partial \mathbf{b}}\big(-E(\mathbf{v},\mathbf{h})\big) = \mathbf{v} \tag{1.26}$$

$$\frac{\partial}{\partial \mathbf{c}}\big(-E(\mathbf{v},\mathbf{h})\big) = \mathbf{h} \tag{1.27}$$

Putting all together the equations 1.24, 1.25, 1.26 and 1.27, the gradients of the log-likelihood take the following forms:

$$\begin{aligned}\frac{\partial}{\partial \mathbf{W}}\mathcal{L}(\Theta) &= \underbrace{\sum_{k=1}^{N} \mathbb{E}_{p(\mathbf{h}|\mathbf{v}^k)}\big[\mathbf{h}\big]\mathbf{v}^{k\,T}}_{\text{the data-driven term}} - \underbrace{N\,\mathbb{E}_{p(\mathbf{v},\mathbf{h})}\big[\mathbf{h}\mathbf{v}^T\big]}_{\text{the model-driven term}} \\ &= \sum_{k=1}^{N} \text{sigmoid}(\mathbf{c} + \mathbf{v}^k\mathbf{W})\mathbf{v}^{k\,T} - N\,\mathbb{E}_{p(\mathbf{v},\mathbf{h})}\big[\mathbf{h}\mathbf{v}^T\big]\end{aligned} \tag{1.28}$$

$$\frac{\partial}{\partial \mathbf{b}}\mathcal{L}(\Theta) = \sum_{k=1}^{N} \mathbf{v}^k - N\,\mathbb{E}_{p(\mathbf{v},\mathbf{h})}\big[\mathbf{v}\big] \tag{1.29}$$

$$\frac{\partial}{\partial \mathbf{c}}\mathcal{L}(\Theta) = \sum_{k=1}^{N} \mathbb{E}_{p(\mathbf{h}|\mathbf{v}^k)}[\mathbf{h}] - N\,\mathbb{E}_{p(\mathbf{v},\mathbf{h})}\big[\mathbf{h}\big] \tag{1.30}$$

Intractability of Gradients: Though we can express the gradients of the log-likelihood, we are not able to calculate the gradients due to the expectations over the joint model distribution $p(\mathbf{v},\mathbf{h})$, i.e., the exponential number of sums due to $(\mathbf{v},\mathbf{h}) \in \{0,1\}^{m+n}$ configurations in computing the partition function (equation 1.14), and therefore, the computations of gradients are still *intractable* even we have conditional distributions $p(\mathbf{v}|\mathbf{h})$ and $p(\mathbf{h}|\mathbf{v})$ that are easy to compute.

Due to the *intractability*, it is impractical to compute the exact log-likelihood gradients leading to approximation strategies in order to train RBMs:

1. **Contrastive Divergence (CD)**: Carreira-Perpiñán and Hinton (2005) and Hinton (2002) proposed the approximation strategy to estimate the expectation term over the joint distribution $p(\mathbf{v},\mathbf{h})$ using the two factorial conditionals ($p(\mathbf{h}|\mathbf{v})$ and $p(\mathbf{v}|\mathbf{h})$) as basis of Gibbs sampling chains (Casella and George, 1992). It is aimed at drawing S Monte Carlo Markov Chain

(MCMC)[8] (Hastings, 1970; Metropolis and Ulam, 1949) samples from the joint distribution $p(\mathbf{v}, \mathbf{h})$ to form a Monte Carlo estimate of the expectations over $p(\mathbf{v}, \mathbf{h})$ as:

$$\mathbb{E}_{p(\mathbf{h},\mathbf{v})}\big[f(\mathbf{h}, \mathbf{v})\big] \approx \frac{1}{S}\sum_{s=1}^{S} f(\mathbf{h}^s, \mathbf{v}^s) \tag{1.31}$$

where the k-step MCMC chains are initialized with the current example from the training set. Additionally, CD requires an extended "burn-in" MCMC step to reach equilibrium/stationary distribution[9] at each iteration. These lead to a biased approximation of the log-likelihood gradient.

2. **Persistent Contrastive Divergence (PCD)** (Stochastic Maximum Likelihood): While CD is a popular method of training RBMs, it suffers from the problem of initialization of the MCMC chains and extended "burn-in" MCMC steps. PCD (Tieleman, 2008; Tieleman and Hinton, 2009) assumes that the model is significantly invariant due to the gradient updates by model parameters in two subsequent iterations and the MCMC state of the penultimate iteration $itr - 1$ should correspond to the equilibrium distribution of the last iteration itr. Thus, instead of initializing the k-step MCMC chain with the current example from the training set, PCD initializes the MCMC chain for training iteration itr with the last state of the MCMC chain from the last training iteration ($itr-1$). It further minimizes the number of "burn-in" MCMC steps required to reach equilibrium distribution at the current iteration itr.

In training an RBM with CD or PCD to estimate the gradients of the log-likelihood, the network "understands" the pair connections in visible and hidden units using the training data, adjusts its parameters so as the probability distribution of the model fits the training data. Thus, the training consists of two phases: (1) *Positive phase*, where the first term in equation 1.28 measures the association between the ith visible and jth hidden unit given the training examples, (2) *Negative phase* (reconstruction), where an RBM generates the states of the visible units from its hypothesis encoded in hidden units, i.e, generates samples that look like they come from the underlying distribution in the data. The second term in

[8]The idea of Monte Carlo simulation is to draw an i.i.d. set of samples from a target density defined on a high-dimensional space. These samples can be used to approximate the target density. Further details: `https://www.cs.ubc.ca/~arnaud/andrieu_defreitas_doucet_jordan_intromontecarlomachinelearning.pdf`

[9]In a stochastic process, a probability distribution that satisfies $\pi = \pi P$, i.e., if you choose the initial state of the Markov chain with distribution π, then the process is stationary and the stationary distribution is invariant under the Markov chain evolution.

Limitations of RBMs	Extensions of RBMs
Intractability due to partition function Z (equation 1.14) and can not efficiently compute the joint probability distribution $p(\mathbf{v}, \mathbf{h})$. Therefore, tricky to train and exact gradients can not be computed.	Neural Autoregressive Distribution Estimator (*NADE*) (Larochelle and Murray, 2011)
Restricted to only binary visible units and can not model real-valued data	Gaussian-Bernoulli RBM (GRBM) (Welling et al., 2004)
Do not model the count data, e.g., document	*Replicated Softmax* (RSM) (Salakhutdinov and Hinton, 2009), *DocNADE* (Larochelle and Lauly, 2012), *iDocNADEe* **(Gupta et al., 2019a)**, *ctx-DocNADEe* (*textTOvec*) **(Gupta et al., 2019b)**

Table 1.4 – *Limitations of an RBM and its (some) extensions, addressing the limitations. The bold indicates our contributions.*

equation 1.28 measures the association between the network generated states of the hidden and visible units. The pair connections are updated for optimal configuration/association(s).

Repeating the positive and negative phases over all training examples, the network parameters are updated due to each of the pair connections using equations 1.28, 1.29 and 1.30. See Hinton (2012) for further details in training RBMs with CD and PCD.

Applications of Restricted Boltzmann Machines

After training of an RBM, it can be used to generate different samples from the learnt distribution and the hidden layer $\mathbf{h}$ encodes the structure of the input data that is further used for dimensionality reduction or feature learning (Hinton and Salakhutdinov, 2006), classification (Larochelle and Bengio, 2008; Salama et al., 2010; Gupta et al., 2015c), regression (Hinton and Salakhutdinov, 2006), collaborative filtering (Salakhutdinov et al., 2007), etc. Additionally to support better generalization in deep architectures from the training data, the stacked RBMs such as Deep Belief Net (DBN) (Bengio et al., 2006; Erhan et al., 2010; Gupta et al., 2015b) can be used to initialize a multi-layer neural network followed by supervised fine-tuning for classification.

Limitations and Variants of RBMs

While RBMs are expressive enough to (1) encode high-order correlations, (2) form a distributed representation of the data, and (3) model any distribution robustly in presence of noise in the training data, they cannot efficiently compute

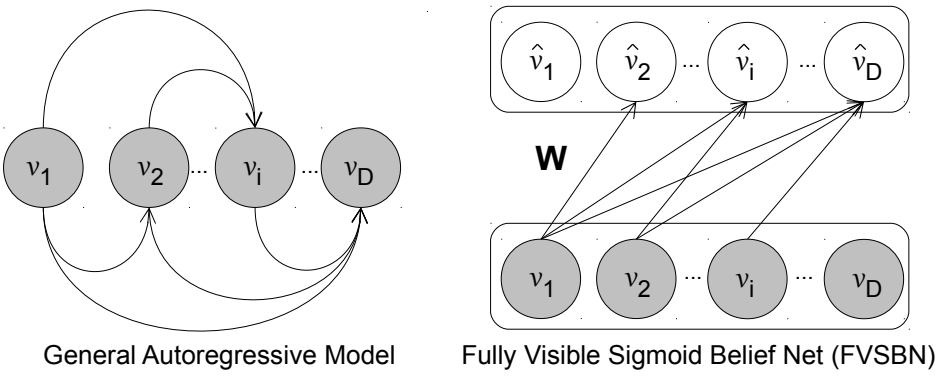


Figure 1.9 – *(left) A general autoregressive graphical model. (right) Fully Visible Sigmoid Belief Net (FVSBN), where $\hat{v}_i = p(v_i = 1|\mathbf{v}_{<i})$ and each autoregressive conditional $\hat{v}_i$ is modeled as logistic regression. The joint probability distribution is decomposed as: $p(\mathbf{v}) = \prod_{i=1}^{D} p(v_i|\mathbf{v}_{<i})$.*

(i.e., intractable) the probability distribution $p(\mathbf{v})$ for a reasonable number of visible and hidden units, and therefore, several approximation strategies have been investigated to estimate it.

Additionally in Table 1.4, we mention some limitations and a few extensions of RBMs that addresses the issues.

1.3.2 Neural Autoregressive Distribution Estimation (NADE)

While the RBMs have the difficulties in computing the likelihood of the model, Larochelle and Murray (2011) proposed an autoencoder-like tractable distribution estimator named as *Neural Autoregressive Distribution Estimator* (NADE) that is inspired by the RBM and aimed at estimating the distribution of binary multivariate observations .

Specifically, NADE is a directed graphical model that factorizes the joint distribution $p(\mathbf{v})$ of a vector (or all variables) $\mathbf{v}$ using a chain rule and expresses it as an ordered product of the one-dimensional distributions, i.e., $p(\mathbf{v}) = \prod_{i=1}^{D} p(v_i|\mathbf{v}_{<i})$, where $\mathbf{v}_{<i} = \{v_1, v_2, ...v_{i-1}\}$ denotes a sub-vector consisting of all attributes preceding $v_i \in \{0, 1\}$ in a fixed arbitrary ordering of the attributes. Importantly, each distribution (called as *autoregressive* conditional) is conditioned on the values of previous dimensions in the (arbitrary) ordering and modeled via a feed-forward neural network. The main *advantage* of NADE model is that each autoregressive conditional $p(v_i|\mathbf{v}_{<i})$ is tractable, therefore the model distribution $p(\mathbf{v})$ is also **tractable**.

Figure 1.9 (left) illustrates the autoregressive property, while a Fully Visible Sigmoid Belief Network (FVSBN) or logistic autoregressive Bayesian networks

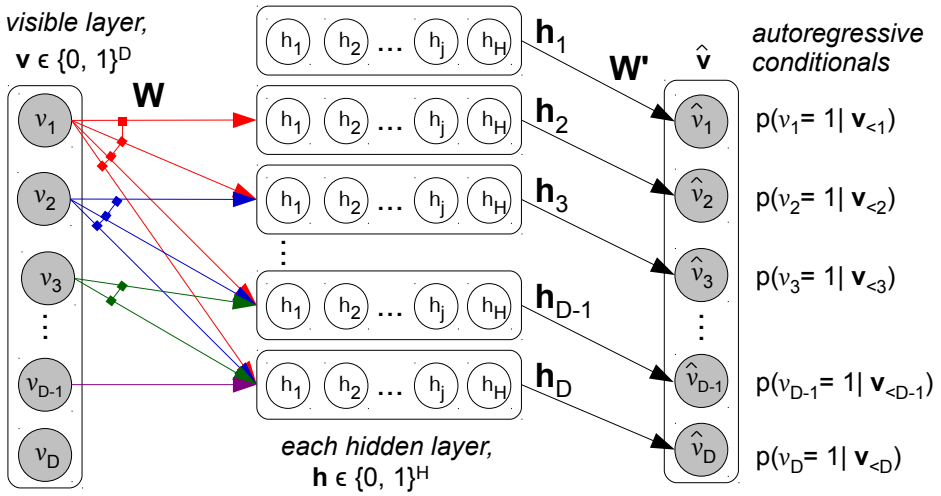


Figure 1.10 – *Illustration of NADE architecture (Uria et al., 2016). Arrows in color connected together correspond to connections with shared (tied) parameters across each of the feed-forward neural networks.*

(Figure 1.9, right) (Frey et al., 1995) shows a powerful framework for deriving a tractable distribution of binary data and models each conditional via a logistic regression. Following the family of fully visible Bayesian networks (Frey et al., 1998), the FVSBN converts an RBM into a Bayesian network (Larochelle and Murray, 2011) and factorizes the joint probability distribution of observations as:

$$p(\mathbf{v}) = \prod_{i=1}^{D} p(v_i|\mathbf{v}_{parents(i)}) \tag{1.32}$$

where all observations v_i are arranged in a directed acyclic graph[10] (DAG) and each $\mathbf{v}_{parents(i)}$ corresponds to variables that are parents of v_i in the DAG, for instance, $\mathbf{v}_{parents(i)} = \mathbf{v}_{<i}$ for an observation v_i. Due to the DAG formulation, each of the conditional distributions is tractable leading to *tractable* joint distribution $p(\mathbf{v})$.

NADE formulation

Instead of a logistic regressor, NADE model extends the FVSBN using a feed-forward neural network for each conditional with one hidden layer $\mathbf{h}_i \in \mathbb{R}^H$ and tied weighted connections going in and out of the hidden layer. Each conditional in the NADE architecture is given by:

[10] a directed graph where a sequence of the vertices is arranged such that every edge is directed from earlier to later in the sequence and every edge is uni-directional.

Algorithm 1 Computation of $p(\mathbf{v})$ in *NADE*
(inspired by Larochelle and Murray (2011))

Input: A training observation vector $\mathbf{v}$
Output: $p(\mathbf{v})$
Parameters: $\{\mathbf{W}, \mathbf{W}', \mathbf{b}, \mathbf{c}\}$

$\mathbf{a} \leftarrow \mathbf{c}$
$p(\mathbf{v}) = 0$
for i from 1 to D **do**
  $\mathbf{h}_i \leftarrow \text{sigmoid}(\mathbf{a})$
  $p(v_i = 1|\mathbf{v}_{<i}) \leftarrow \text{sigmoid}(b_i + \mathbf{W'}_{i,:}\mathbf{h}_i)$
  $p(\mathbf{v}) \leftarrow p(\mathbf{v})\Big(p(v_i = 1|\mathbf{v}_{<i})^{v_i} + \big(1 - p(v_i = 1|\mathbf{v}_{<i})\big)^{1-v_i}\Big)$
  $\mathbf{a} \leftarrow \mathbf{a} + \mathbf{W}_{:,i}v_i$

$$\hat{v}_i = p(v_i = 1|\mathbf{v}_{<i}) = \text{sigmoid}(b_i + \mathbf{W'}_i^T\mathbf{h}_i) \tag{1.33}$$

$$\text{where,} \quad \mathbf{h}_i = \text{sigmoid}(\mathbf{c} + \mathbf{W}_{:,<i}\mathbf{v}_{<i}) = \text{sigmoid}(\mathbf{c} + \sum_{k<i}\mathbf{W}_{:,<i}) \tag{1.34}$$

and $\mathbf{W}_{:,<i}$ is a matrix made of the $i-1$ first column of $\mathbf{W}$ and sigmoid(x) = $1/(1+\exp(-x))$. Equations 1.33 and 1.34 correspond to a feed-forward neural network for each autoregressive conditional $p(v_i = 1|\mathbf{v}_{<i})$, and connections are tied (or shared) *across* these neural networks, as marked by the *colored* lines in Figure 1.10. Unlike in the RBM, NADE architecture does not require a symmetric connection and therefore, uses a separate matrix $\mathbf{W}' \in \mathbb{R}^{H\times D}$ in the hidden-to-input connections during the reconstruction.

In order to model the conditionals $p(v_i = 1|\mathbf{v}_{<i})$, a NADE model is inspired by the mean-field procedure, where the forward pass in NADE corresponds to applying a single pass of mean-field inference. NADE is related to the RBM as these computations (equations 1.33 and 1.34) are inspired by the approximation inference in the RBM. See Raiko et al. (2014) and Larochelle and Murray (2011) for further details about how NADE is computationally related to a mean field inference and RBM, respectively.

The Shared Activations Trick: The parameter $\mathbf{W}$ sharing is advantageous in NADE architecture, since it speeds up the computation of conditionals from quadratic to linear time. Let's denote the pre-activations of the i^{th} and $(i+1)^{th}$ hidden layers by $\mathbf{a}_i$ and $\mathbf{a}_{(i+1)}$, respectively. The linear complexity can be achieved by the following recurrence:

$$\mathbf{a}_{i+1} = \mathbf{a}_i + \mathbf{W}_{:,i}\mathbf{v}_i = \mathbf{a}_i + \mathbf{W}_{:,i} \quad \text{and} \quad \mathbf{a}_1 = \mathbf{c} \tag{1.35}$$

for $i \in 2, ..., D$, since the pre-activations differ by:

$$\begin{aligned} \mathbf{a}_{i+1} - \mathbf{a}_i &= (\mathbf{c} + \mathbf{W}_{:,<i}\mathbf{v}_{<i+1}) - (\mathbf{c} + \mathbf{W}_{:,<i}\mathbf{v}_{<i}) \\ &= \mathbf{W}_{:,i+1}\mathbf{v}_{i+1} = \mathbf{W}_{:,i+1} \end{aligned} \tag{1.36}$$

where the equation 1.36 can be computed in $O(H)$. The NADE computes the probability distribution $p(\mathbf{v})$ by factorizing it into D conditionals and therefore, computing $p(\mathbf{v})$ costs in time linear in dimensionality $O(DH)$, instead of in time quadratic $O(D^2H)$. NADE achieves it by sharing the computations of the pre-activations across the conditionals in each of the feed-forward neural networks.

Loss function: Given the training set $\{\mathbf{v}\}_{k=1}^{N}$, NADE minimizes the average negative log-likelihood,

$$\mathcal{L}(\mathbf{W}, \mathbf{W}', \mathbf{b}, \mathbf{c}) = \frac{1}{N}\sum_{k=1}^{N} -\log p(\mathbf{v}^k) = \frac{1}{N}\sum_{k=1}^{N}\sum_{i=1}^{D} -\log p(v_i^k|\mathbf{v}_{<i}^k) \tag{1.37}$$

and the minimization can be performed using stochastic (mini-batch) gradient descent in order to learn the model parameters ($\mathbf{W}$, $\mathbf{W}'$, $\mathbf{b}$ and $\mathbf{c}$). Algorithm 1 illustrates the pseudocode of computing the probability distribution $p(\mathbf{v})$ in the NADE model.

NADE Extensions: While the NADE is restricted to binary observations, Uria et al. (2013) proposed its extension, named as Real-valued Neural Autoregressive Density Estimator (RNADE) that models real-valued observations using a mixture of Gaussians to represent the conditional distributions. Moreover, Uria et al. (2014) and Zheng et al. (2013) proposed deep and supervised variants of NADE architecture. On other hand, DocNADE (Larochelle and Lauly, 2012; Lauly et al., 2017) models text documents via multinomial observations, for instance, to perform topic modeling.

1.3.3 Replicated Softmax (RSM)

The Replicated Softmax (RSM) (Salakhutdinov and Hinton, 2009) is an undirected probabilistic topic model to learn representations of documents, and a generalization (Figure 1.7) of the RBM, since (1) words are multinomial observations, not binary and (2) documents are of varying lengths. It is difficult to model documents in the RBM (Smolensky, 1986; Freund and Haussler, 1992; Hinton, 2002) even when the word-count vectors are modeled as a Poisson distribution (Gehler et al., 2006).

As illustrated in Figure 1.11 (left), RSM can be interpreted as a collection of different-sized RBMs created for documents of different lengths, where each

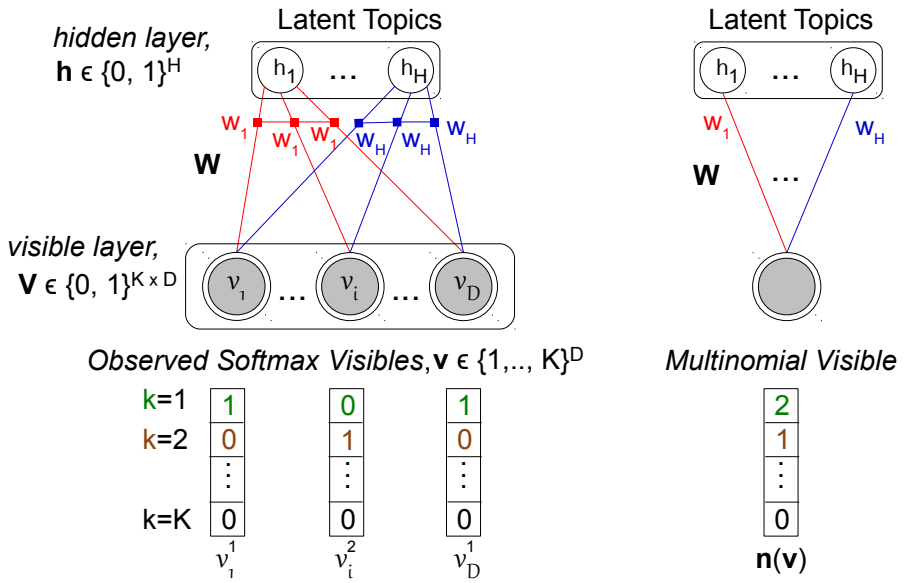


Figure 1.11 – *An ilustration of the Replicated Softmax (RSM). (left) RSM with D softmax units as there are D words in the document. The connections* $\mathbf{W}$ *between each observation v_i and hidden units are shared, where v_i is an index of a word w. (right): A different interpretation of RSM where the D softmax units with identical weights is equivalent to a single multinomial unit that is sampled D times. Notations:* $\mathbf{V} \in \mathbb{R}^{K \times D}$ *is an observed binary matrix such that a word index $v_i^k = 1$ if the visible unit i takes on k^{th} value in the dictionary of size K. E.g., for a document of size D, an observation vector $\mathbf{v} = [v_1, ..., v_i, ..., v_D]$ is a sequence of word indices v_i taking values in $\{1, ..., K\}$. In the illustration, a document of $D = 3$ words (w_1, w_2, w_1) respectively be indexed in vocabulary at $(1, 2, 1)$ is represented by $\mathbf{v} = [1, 2, 1]$.* $\mathbf{n(v)}$ *is a vector of size K with the word-counts of each word in the vocabulary.*

RBM has as many softmax (multinomial) observation units as there are words in the corresponding document with the weights between an observed and all latent units are shared (replicated) across all the observed units. Moreover, the weights are shared across the whole family of different-sized RBMs and therefore, the name *Replicated Softmax*. It enables RSM to model documents of different lengths.

Being a generative model of word counts, RSM has demonstrated an efficient training using Contrastive Divergence (CD) (Carreira-Perpiñán and Hinton, 2005; Hinton, 2002; Tieleman, 2008; Tieleman and Hinton, 2009), a better dealing with documents of different lengths and better generalization compared to Latent Dirichlet Allocation (LDA) (Blei et al., 2003) in terms of both the log-probability

on unseen documents and retrieval accuracy. Additionally, RSM has been used in extracting topics from a collection of documents, where its hidden vector encodes a document representation (Salakhutdinov and Hinton, 2009).

RSM formulation

As illustrated in Figure 1.11, the RSM is a 2-layered architecture with visible units $\mathbf{v} \in \{1, ..., K\}^D$ and binary hidden units (latent topic features) $\mathbf{h} \in \{0, 1\}^H$, where D, K and H are document, dictionary and hidden layer sizes, respectively. Assuming a document of size D and its words be indexed at $(v_1, v_2, ..., v_D)$ in the vocabulary, the observation vector $\mathbf{v}$ for the document is given by a sequence of word indices taking values in $\{1, ..., K\}$. Here, each observed softmax visible v_i is a word (marked by double circle in Figure 1.11, left), where the connections $\mathbf{W}$ between each softmax observation and hidden units are shared (marked by the colored lines and tied connections in Figure 1.11, left). Moreover, observe in Figure 1.11 (left and right) that the D softmax units with identical weights are equivalent to a single multinomial unit that is sampled D times.

RSM is a generalization of RBM with shared connections across different positions i in $\mathbf{v}$ and thus, defined by the energy of the state $\{\mathbf{v}, \mathbf{h}\}$ for a document of size D with word indices $(v_1, ..., v_i, ..., v_D)$ (each v_i modeled as a softmax unit),

$$\begin{aligned} E(\mathbf{v}, \mathbf{h}) &= -\sum_{i=1}^{D} b_{v_i} - \sum_{i=1}^{D} \mathbf{h}^T \mathbf{W}_{:,v_i} - \sum_{i=1}^{D} \mathbf{c}^T \mathbf{h} \\ &= -\mathbf{b}^T \mathbf{n}(\mathbf{v}) - \mathbf{h}^T \mathbf{W} \mathbf{n}(\mathbf{v}) - D\mathbf{c}^T \mathbf{h} \end{aligned} \tag{1.38}$$

with model parameters $\{\mathbf{W}, \mathbf{b}, \mathbf{c}\}$, where $\mathbf{W}_{:,v_i}$ is the v_i^{th} column vector extracted from the matrix $\mathbf{W} \in \mathbb{R}^{H \times K}$ and $\mathbf{n}(\mathbf{v}) \in \mathbb{R}^K$ is a vector obtained by summing the word count of each word v_i in the vocabulary, as shown in Figure 1.11 (right). Observe that the hidden-bias term $\mathbf{c}^T \mathbf{h}$ is multiplied (scaled up) by the document length D in order to maintain a balance between all the terms, especially when $\mathbf{v}$ is larger (i.e., a number of summations over i) and documents are of different lengths.

Similar to the RBMs, the probability distribution of the RSM model is related to its energy and is given by:

$$p(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} \exp\big(-E(\mathbf{v}, \mathbf{h})\big) \text{ and } Z = \sum_{\mathbf{v}, \mathbf{h}} \exp\big(-E(\mathbf{v}, \mathbf{h})\big) \tag{1.39}$$

where Z is the normalization constant and intractable. Following the RBMs, the conditional distributions across layers are factorized as:

$$p(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^{D} p(v_i|\mathbf{h}) \quad \text{and} \quad p(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^{H} p(h_j|\mathbf{v}) \tag{1.40}$$

and each conditional is computed as:

$$\begin{aligned} p(h_j = 1|\mathbf{v}) &= \text{sigmoid}(Dc_j + \sum_{i}^{D} W_{j,v_i}) \\ p(v_i = w|\mathbf{h}) &= \frac{\exp(b_w + \mathbf{h}^T \mathbf{W}_{:,w})}{\sum_{w'} \exp(b_{w'} + \mathbf{h}^T \mathbf{W}_{:,w'})} \end{aligned} \tag{1.41}$$

where $p(v_i = w|\mathbf{h})$ is the softmax visible unit for the word index v_i. Observe that the distribution of each word v_i in the document is obtained due to a contribution from each of the topic features in $\mathbf{h}$.

Similar to the RBMs (section 1.3.1), the computation of the gradients of the negative log-likelihood of training documents with respect to model parameters is expensive for sufficiently large $\mathbf{v}$ due to a number of sums in computing the partition function Z (equation 1.39). Hence, the gradients are approximated using the contrastive divergence (CD) (Carreira-Perpiñán and Hinton, 2005; Hinton, 2002) or its variant PCD (Tieleman, 2008; Tieleman and Hinton, 2009). See Salakhutdinov and Hinton (2009) for more details in training the Replicated Softmax.

Limitations of RSM

The RSM has difficulty in training due to intractability in its partition function (normalization constant). Even, computing the conditional $p(v_i|\mathbf{h})$ for each word v_i via Gibbs sampling is expensive when the dictionary size K tends to be quite large. Additionally, RSM is a bag-of-words topic model and therefore, does not account for the word ordering that might be helpful for textual representation in certain tasks. Moreover, RSM is a static topic model in the sense that it does not consider the temporal ordering of documents.

Our Contribution: As illustrated in Figure 1.7, we attempt to extend RSM for the dynamic[11] topic model setting and present a novel unsupervised neural architecture, which we named as *Recurrent Neural Network-Replicated Softmax Model* (*RNN-RSM*) (Gupta et al., 2018b) that facilitates the identification of topical and keyword trends over time in temporal collections of unstructured documents. Essentially, the RNN-RSM model can be seen as a temporal stack of RSM

[11]Generative models to analyze the evolution of topics of a collection of documents over time. The documents are grouped by time slice (e.g., years) and it is assumed that the documents of each group come from a set of topics that evolved from the set of the previous slice.

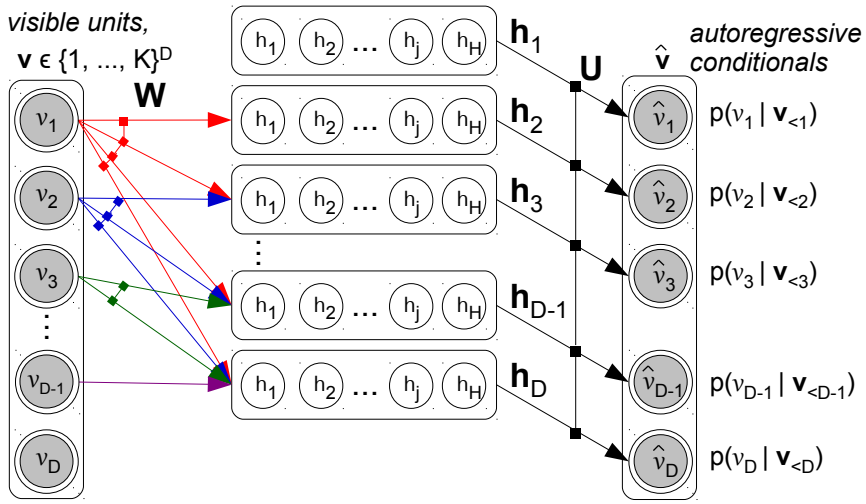


Figure 1.12 – *Illustration of DocNADE architecture (Larochelle and Lauly, 2012; Gupta et al., 2019a). Arrows in color connected together correspond to connections with shared (tied) parameters across each of the feed-forward neural networks. The double boundary circle represents a softmax (multinomial) observation for each word i indexed at v_i.*

models, conditioned by time-feedback connections using RNN. While RSM captures topical information at each time slice t, the time feed-back connections of RNN convey topical information through RSM biases across the time steps $<t$.

1.3.4 Neural Autoregressive Topic Model (DocNADE)

While NADE (Larochelle and Murray, 2011) (section 1.3.2) and RSM (Salakhutdinov and Hinton, 2009) (section 1.3.3) are good alternatives than RBM, NADE is limited to binary data and RSM has difficulties due to large vocabulary size and intractability leading to approximate gradients of the negative log-likelihood. Unlike in the RBM and RSM, NADE has an advantage that computing the gradients with respect to model parameters does not require approximation. On other hand, RSM is a generative model of word count to learn meaningful representations of documents.

Inspired by the benefits of NADE and RSM (Figure 1.7), Larochelle and Lauly (2012) proposed a neural network based generative topic model named as *Document Neural Autoregressive Distribution Estimator* (DocNADE) that learns topics over a sequence of words in a language modeling fashion (Bengio et al., 2003; Mikolov et al., 2010), where each word v_i is modeled by a feed-forward neural network accounting for preceding words $\mathbf{v}_{<i}$ in the sequence.

Figure 1.12 provides an illustration of the DocNADE model.

Specifically, DocNADE factorizes the joint distribution of words in a document as a product of conditional distributions and models each conditional via a feed-forward neural network to efficiently compute a document representation following the NADE architecture. Similar to the RSM, a DocNADE treats each word in a document as a multinomial observation and thus, computes a multinomial distribution given the hidden units at each of the autoregressive steps. More specifically in difference to the RSM, DocNADE organized words in a document in a hierarchy of binary logistic regressions, i.e., a binary tree where each leaf corresponds to a word of the vocabulary. It enables DocNADE with a competitive complexity of computing the probability of an observed word that scales sublinearly (i.e., logarithmic) with vocabulary size, as opposed to linearly in RSM.

In modeling documents, DocNADE has shown an improved performance over the other topic models such as LDA (Blei et al., 2003) and RSM (Salakhutdinov and Hinton, 2009) in terms of generalization over the unseen documents and information retrieval.

DocNADE Formulation

For a document $\mathbf{v} = (v_1, ...v_D)$ of size D, each word index v_i takes value in $\{1, ..., K\}$ from a dictionary of vocabulary size K. DocNADE models the joint probability distribution $p(\mathbf{v})$ of the document by decomposing it as the product of conditional distributions based on the probability chain rule, and is given by:

$$p(\mathbf{v}) = \prod_{i=1}^{D} p(v_i|\mathbf{v}_{<i}) \tag{1.42}$$

and each autoregressive conditional distribution $p(v_i|\mathbf{v}_{<i})$ is modeled by a feed-forward neural network,

$$\begin{aligned} \mathbf{h}_i(\mathbf{v}_{<i}) &= g(\mathbf{c} + \sum_{k<i} \mathbf{W}_{:,v_k}) \\ p(v_i = w|\mathbf{v}_{<i}) &= \frac{\exp(b_w + \mathbf{U}_{w,:}\mathbf{h}_i(\mathbf{v}_{<i}))}{\sum_{w'} \exp(b_{w'} + \mathbf{U}_{w',:}\mathbf{h}_i(\mathbf{v}_{<i}))} \end{aligned} \tag{1.43}$$

for $i \in \{1, ...D\}$, where $\mathbf{v}_{<i}$ is the subvector consisting of all v_j such that $j < i$ i.e., $\mathbf{v}_{<i} \in \{v_1, ..., v_{i-1}\}$, $g(\cdot)$ is a non-linear activation function, $\mathbf{W} \in \mathbb{R}^{H \times K}$ and $\mathbf{U} \in \mathbb{R}^{K \times H}$ are weight matrices, $\mathbf{c} \in \mathbb{R}^H$ and $\mathbf{b} \in \mathbb{R}^K$ are bias parameter vectors. H is the number of hidden units (topics). Unlike the RSM, DocNADE ignored the scaling factor D for the hidden bias $\mathbf{c}$ based on the performance of the model.

Algorithm 2 Computation of $\log p(\mathbf{v})$ in *DocNADE* using *tree-softmax* or *full-softmax* (inspired by Larochelle and Lauly (2012) and Gupta et al. (2019a))

Input: A training document vector $\mathbf{v}$
Parameters: $\{\mathbf{b}, \mathbf{c}, \mathbf{W}, \mathbf{U}\}$
Output: $\log p(\mathbf{v})$

$\mathbf{a} \leftarrow \mathbf{c}$
$p(\mathbf{v}) = 1$
for i from 1 to D **do**
  $\mathbf{h}_i \leftarrow g(\mathbf{a})$
  if `tree-softmax` **then**
    $p(v_i|\mathbf{v}_{<i}) = 1$
    for m from 1 to $|\pi(v_i)|$ **do**
      $p(v_i|\mathbf{v}_{<i}) \leftarrow p(v_i|\mathbf{v}_{<i})p(\pi(v_i)_m|\mathbf{v}_{<i})$
  if `full-softmax` **then**
    compute $p(v_i|\mathbf{v}_{<i})$ using equation 1.43
  $p(\mathbf{v}) \leftarrow p(\mathbf{v})p(v_i|\mathbf{v}_{<i})$
  $\mathbf{a} \leftarrow \mathbf{a} + \mathbf{W}_{:,v_i}$

As illustrated in Figure 1.12 and equation 1.43, notice that the conditional distribution $p(v_i = w|\mathbf{v}_{<i})$ of each word v_i is thus computed in a feed-forward fashion using a position-dependent hidden layer $\mathbf{h}_i(\mathbf{v}_{<i})$ that learns a representation based on all previous words $\mathbf{v}_{<i}$ in the sequence $(v_1, ..., v_i, ..., v_D)$. Moreover, computing the hidden representation at each of the autoregressive step is efficient due to the NADE architecture that leverages the pre-activation[12] $\mathbf{a}_{i-1}$ of $(i-1)^{th}$ step in computing the pre-activation $\mathbf{a}_i$ for the i^{th} step. The *shared activation trick* is further described in section 1.3.2.

Taken together, the negative log-likelihood of any document $\mathbf{v}$ of an arbitrary length can be computed as:

$$\mathcal{L}(\mathbf{v}) = \sum_{i=1}^{D} \log p(v_i|\mathbf{v}_{<i}) \tag{1.44}$$

The model parameters $\{\mathbf{b}, \mathbf{c}, \mathbf{W}, \mathbf{U}\}$ are learned by minimizing the average negative log-likelihood of the training documents using stochastic gradient descent.

Algorithm 2 gives the computation of $\log p(\mathbf{v})$ in *DocNADE* using a hierarchical (*tree-softmax*) or large softmax (*full-softmax*) over vocabulary to compute the autoregressive conditionals. Similar to probabilistic language models (Morin

[12]term before the application of non-linearity, for instance $a_i = \mathbf{c} + \sum_{k<i} \mathbf{W}_{:,v_k}$

and Bengio, 2005; Mnih and Hinton, 2008), DocNADE replaces a large softmax over words by a probabilistic tree model in which each path from the root to a leaf corresponds to a word. The probabilities π of each left/right transitions in the tree are modeled by a set of binary logistic regressors and the probability of a given word is then obtained by multiplying the probabilities of each left/right choice of the associated tree path.

See Larochelle and Lauly (2012) for details about the realization of a hierarchical binary logistic regression and computing the gradients with respect to the model parameters.

To compute the autoregressive conditional distributions $p(v_i|\mathbf{v}_{<i})$ for each word $i \in [1, 2, ..., D]$, the binary word tree instead of softmax over words reduces computational cost and achieves a complexity logarithmic in K. For a full binary tree of K words, it involves $O(\log(K))$ binary logistic regressions where each logistic regression requires $O(H)$ computations. Since there are D words, the complexity of computing all $p(v_i|\mathbf{v}_{<i})$ is in $O(\log(K)DH)$. Therefore, the total complexity of computing $p(\mathbf{v})$ in DocNADE with binary tree softmax is $O(\log(K)DH + DH)$, as opposed to $O(KDH + DH)$ of Replicated Softmax. Thus, DocNADE offers a complexity competitive to RSM for a large vocabulary size K.

Importantly, the mean field inference of $p(v_i|\mathbf{v})< i)$ in RSM corresponds to the mean field inference in RBM, given the multinomial observations in RSM. Following the derivation of NADE, DocNADE estimates $p(v_i|\mathbf{v})< i)$ with a single iteration of mean field procedure applied to RSM. See Larochelle and Lauly (2012) for further details.

Limitations of DocNADE and Our Contributions

While DocNADE has shown promising results in terms of generalization and IR tasks, it experiences the following limitations. Here, we first outline limitations of the DocNADE architecture, and then present our contributions to address them:

1. **Limitation**: In computing an autoregressive conditional, i.e., $p(v_i|\mathbf{v}_{<i})$ for a word v_i in a given sequence, DocNADE considers only the preceding context, i.e., $\mathbf{v}_{<i}$ while learning latent topics.

   Contribution: To extend, we present a novel architecture, which we named as *iDocNADE* (Gupta et al., 2019a) that exploits the full context information around words in the given document. Here, the prefix *i* stands for *informed*.

2. **Limitation**: DocNADE does not address the difficulties in learning representations especially in the limited context settings, e.g., short-text or a corpus of few documents.

Contribution: To address the data sparsity issues, we present a novel architecture, which we named as *DocNADEe* (Gupta et al., 2019a) that incorporates external knowledge, e.g., pre-trained word embeddings into the neural autoregressive topic model.

Moreover, we present a novel approach of *Multi-view Transfer* (*MVT*) (Chapter 7) in the DocNADE model with an aim to inject the two kinds of external knowledge: word embeddings (local semantics) and latent topics (global semantics) from many sources.

3. **Limitation**: DocNADE is a bag-of-words model and therefore, it does not account for language structures such as word ordering, local syntactic or semantic information, etc.

   Contribution: To this end, we present a novel **neural composite** model of topic learning that accounts for both the global and local contexts (section 1.4.4) while learning word and document representations. We call the proposed modeling approach as *contextualized-DocNADE* (*ctx-DocNADE*) (Gupta et al., 2019b) that generates *contextualized topic vectors* (*textTOvec*) in the sense that the local semantics is assimilated in the global (i.e., topics) semantic information.

   In doing so, an LSTM-LM (LSTM-based Language Model) captures language structures by accounting for the word ordering in local word co-occurrences at the sentence-level. However, a topic model (i.e., DocNADE) learns latent topics from the entire document and discovers the underlying thematic structures (i.e., global semantics) in the document collection. To benefit from the merits of the two complementary semantics, we unite the two paradigms of learning in a *composite* model (section 1.4.5) that jointly trains a neural topic and a neural language model.

   Also, we demonstrate an improved performance of the composite model in the sparse-data settings by introducing pre-trained word embeddings.

1.4 Distributional Semantics: Word and Document Representations

Recently, the success of deep learning based NLP systems is coupled with distributed representations of words (Mikolov et al., 2013b; Pennington et al., 2014), phrases (Socher et al., 2012, 2011b) or sentences (Le and Mikolov, 2014; Kiros et al., 2015) where the distributed representations are real-valued vectors to flexibly represent semantics of natural language.

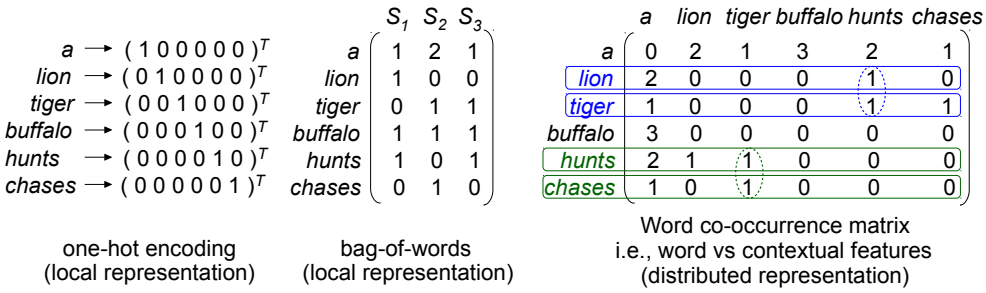


Figure 1.13 – *An illustration of Local vs. Distributed Representations using word co-occurrence matrix of one-window, and sentences S_1, S_2 and S_3*

1.4.1 Distributed Representations

Specifically, "word embeddings" are distributional vectors following the distributional hypothesis in the sense that the words appearing in similar context have similar meaning. In other words, they are learned representations of text where words that have the same meaning have a similar representation. Following the objective, neural networks (Bengio et al., 2003; Mikolov et al., 2010) have been successful in learning vector representation of a word based on its context. Consequently, each word is mapped to a real-valued vector in a predefined vector space.

To give an intuition for local and distributed representations (i.e., *word embeddings*) of words, consider the following three sentences in a small corpus:

S_1: `a lion hunts a buffalo`
S_2: `a tiger chases a buffalo`
S_3: `a tiger hunts a buffalo`

Local Representations

Figure 1.13 demonstrates local representations via a one-hot word vector representation (left) and a bag-of-words (right) representation for each of the three sentences. The one-hot representation denotes each single word as a binary vector of vocabulary size with one value 1 at the word-specific index and remaining values 0. On other hand, each dimension in the bag-of-words representation of a sentence indicates a word. These local representations get too sparse (many zeros) when the size of document and/or vocabulary grows. Additionally, the bag-of-words model does not account for the word ordering, although the words are presented in a sequence.

Limitations of local representations: (1) Extremely inefficient as the vocabulary size of the corpus increases, (2) Do not account for word ordering in the

sequence, (3) Inefficient in neural networks due to sparse vector representations, and (4) Can not capture word similarity because of a one-to-one mapping from a word to a vector. Therefore, words appearing differently in symbols are treated independently and mapped to different word indices in the vocabulary.

Distributed Representations

In contrast, distributed representations are powerful in the sense that each word is represented by a d-dimensional real-valued dense vector, and the whole vector represents a word or a sequence of words instead of only one dimension as in local representations. Essentially in a distributed representation (Plate, 1995; Hinton, 1986), the informational content is distributed among multiple units, and at the same time each unit can contribute to the representation of multiple elements. Specifically for words, the resulting distributional vectors represent words by describing information related to the contexts in which they appear.

To this end, Figure 1.13 (right) intuitively explains the concept of distributed representations, where we generate a distributed representation for each word by building a word vs. contextual feature (i.e., word co-occurrence) matrix using the corpus of three sentences and considering a 1-word window. Notice that a row vector is a distributed representation of a particular word, e.g., *lion* as highlighted, whereas all other words (corresponding columns) within the context of *lion* contribute in generating its distributed representation. Moreover, observe that the words (e.g. *lion* and *tiger*) sharing similar semantic attributes (e.g., *hunts*) are similar. Similarly, the word *hunts* is more similar to *chases* than *buffalo*, *lion* or *tiger*. Therefore, the words *lion* and *tiger* are represented by vectors that are similar in cosine similarity.

However, it is challenging to learn distributed representations explicitly using the original co-occurrence matrix that is very costly to obtain and store for large corpora. In order to deal with the large co-occurrence matrix and exploit the expressive power of neural networks, Bengio et al. (2003) and Mikolov et al. (2010) employed a neural network based approach using contextualized information with no explicitly computed word co-occurrence matrix. They generated distributed, compact and dense vector representations of words, capture word semantics and interesting relationship between words.

Neural Language Models

A language model (LM) computes the probability distribution of the next word in a sequence, given the sequence of previous words in a predefined vocabulary $\mathbb{V}$. Assuming a sequence of D words indices $\{v_1, v_2, ...v_D\}$ with each $v_d \in \mathbb{V}$, a language model computes the joint probability distribution of the sequence as:

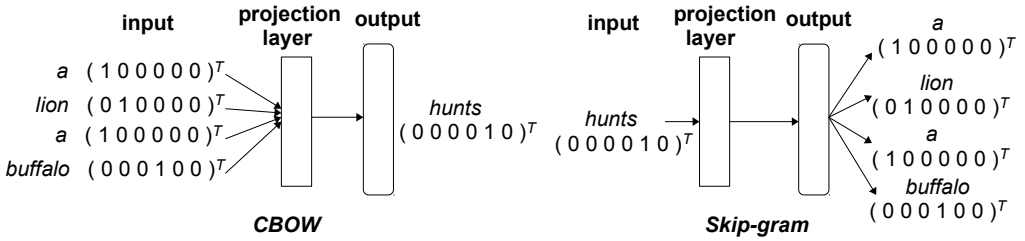


Figure 1.14 – *Word2vec: CBOW vs Skip-gram models*

$$p(v_1, v_2, ...v_D) = \prod_{d=1}^{D} p(v_d|\mathbf{v}_{<d}) \tag{1.45}$$

where $\mathbf{v}_{<d}$ is a sub-vector consisting of word indices preceding the word d in the given sequence. The conditional distribution of each word v_d is computed through the hidden state $\mathbf{h}_d$ of the RNN-based model,

$$\begin{aligned} p(v_d|\mathbf{v}_{<d}) &= p(v_d|\mathbf{h}_d) \\ \mathbf{h}_d &= f_H(\mathbf{h}_{d-1}, v_{d-1}) \end{aligned} \tag{1.46}$$

where $f_H(\cdot)$ can be realized by a basic RNN cell (Elman, 1990), LSTM (Hochreiter and Schmidhuber, 1997) cell, GRU (Chung et al., 2014) cell or an MLP (Bengio et al., 2003). Therefore, the name *neural language* model.

In the process of predicting distributions over words, the language models encode language structures, such as word ordering, semantic knowledge, grammatical structure, etc. in the text. RNN-based language models (Mikolov et al., 2010; Peters et al., 2018) have been successful in variety of NLP applications. However, they are typically applied at the sentence level without access to the broad document context, and consequently, it is difficult to capture long-term dependencies of a document (Dieng et al., 2017).

1.4.2 Learning Distributed Representations

In the following section, we briefly cover some of the major related works in learning distributed representations of words and sentences.

Distributed Word Representations

Word2vec (Mikolov et al., 2013b) is one of the popular approaches in learning word vectors given the local context, where the context is defined by a window of neighboring words. They proposed a simple single-layer architecture based on

the inner product between two word vectors and introduced two different learning techniques to learn word embeddings: (1) *Continuous Bag-of-Words model* (CBOW) that learns embeddings by predicting the current word based on its context, and (2) *Continuous Skip-Gram model* that learns by predicting a context (surrounding words) given an input word. Figure 1.14 depicts the two approaches of word2vec technique to model the sentence S_1. To train such models, many such pairs of word-context are provided during training and the prediction capability of the models is maximized.

Notice that the context is characterized by the window size that has a strong effect on the resulting vector similarities. The window-based methods also suffer from the disadvantage that they do not operate directly on the global co-occurrence statistics of the corpus. Instead, they scan context windows across the entire corpus, thus fail to leverage the vast amount of repetition in the data.

While matrix factorization methods such as as latent semantic analysis (LSA) (Deerwester et al., 1990) efficiently leverage global statistical information, they do relatively poorly on the word analogy task. In contrast, word2vec did better on the analogy task, but they poorly utilize the statistics of the corpus because they train on separate local context windows rather than on global co-occurrence counts. Therefore, **GloVe** (Global Vectors for Word Representation) (Pennington et al., 2014) algorithm extends word2vec method by combining global text statistics of matrix factorization techniques with the local context-based learning in word2vec.

As the two conventional word embeddings techniques (word2vec and GloVe) do not handle out-of-vocabulary (OOV) words, Bojanowski et al. (2017) proposed a technique called **fastText** and introduced the idea of subword-level embeddings that represents each word by a bag of character n-grams. Specifically, special boundary symbols $<$ and $>$ are added to mark the beginning and end of a word. For instance, a word 'halwa' with $n = 3$ is represented by character 3-grams ('$< ha$', 'hal', 'alw', 'lwa', '$wa >$') and a special sequence '$< halwa >$'. Since each character n-gram is associated to a vector representation, therefore a word is represented by the sum of these representations. As a result, this technique enables to compute representations for OOV words.

Buffalo survives a lion attack.
The State University of New York is based in Buffalo.
Buffalo is a brand of clothing and accessories.

Table 1.5 – *Different senses of the word 'buffalo' based on its context.*

Contextualized Word-Embeddings: Though word embeddings have shown to be powerful in capturing semantic properties of words, their inability to deal with different meanings (senses) of a word restricts their effectiveness in captur-

ing the semantics of ambiguous/polysemous words. Essentially, the conventional word embedding models generate the same embedding for the same word in different contexts. For instance in the sentences above (Table 1.5), the word 'buffalo' refers to an animal, location and brand, respectively.

Instead of learning a fixed number of senses per word, contextualized word embeddings learn "senses" dynamically, i.e., their representations dynamically change depending on the context in which a word appears. Recently, (Peters et al., 2018) have shown to capture context-dependent word semantics in order to address the issue of polysemous words. To achieve it, they first train a bi-directional LSTM-based language model (LSTM-LM) on large corpora, and then at test time, use the hidden states generated by the LSTM for each token to compute a vector representation of each word, i.e., *Embeddings from Language Models* (**ELMo**). The vector representation is a function of the task-specific sentence in which it appears, therefore the name contextualized embeddings. Moreover, they have shown that a deep contextualized LSTM-LM is able to capture different language concepts in a layer-wise fashion, e.g., the lowest layer captures language syntax and topmost layer captures semantic features useful in sense disambiguation.

More recently, Akbik et al. (2018) introduced a contextualized character-level word embedding (named as **FlairEmbeddings**) in sequence labeling that models words and context as sequences of characters. It offers several benefits such as generating different embeddings for polysemous words dependent on their context, handling rare and misspelled words, and accounting for subword-level structures, e.g., prefixes and suffixes.

Distributed Sentence Representations

One of the straightforward bottom-up baseline method in generating distributed sentence representation is to compose pre-trained word embeddings by element-wise addition (Mitchell and Lapata, 2010).

Some of the sophisticated methods include Paragraph Vector (**doc2vec**) (Le and Mikolov, 2014) and **SkipThought** Vectors (Kiros et al., 2015) that employ neural networks. The doc2vec is an extension of word2vec that introduced two models of sentence representations: (1) a Distributed Memory (DM) model that learns a paragraph in such a way that the paragraph vector is concatenated with several word vectors from a paragraph and predict the following word in the given context, and (2) a Distributed Bag-of-words (DBOW) model that ignores the context words in the input and forces the model to predict words randomly sampled from the paragraph in the output. However, SkipThought Vectors (Kiros et al., 2015) are trained to predict target sentences (preceding and following) given a source sentence. In doing so, they employed sequence-to-sequence RNN-based models (Sutskever et al., 2014).

1.4.3 Document Topic Models

A topic model (TM) is a type of statistical modeling that examines how words co-occur across a collection of documents, and automatically discovers coherent groups of words (i.e., themes or topics) that best explain the corpus. Essentially, they assume that (1) each document is composed of a mixture of topics, and (2) each topic is composed of a collection of words. A TM aims at uncovering the underlying semantic structure (i.e.., theme) of a document collection so as to organize, search or summarize according to these themes.

For instance, documents that contain frequent occurrences of words such as {*computer, information, data, software, network, device, keyboard*} are likely to share a topic on "computers".

Essentially, topics are inferred from the observed word distributions in the corpus and the semantics of topics are usually inferred by examining the top ranking words they contain. The number of topics are predefined in the unsupervised learning algorithm.

Unlike word-word co-occurrence matrices in learning word embeddings (section 1.4.1), topic models generate distributional semantic representations of words by *document-word* matrices with an intuition that words are similar if these words similarly appear in documents. In other words, words belonging to the same topic tend to be more similar and each topic is associated to some documents. Therefore, topic models have **global view** in the sense that each topic is learned by leveraging statistical information across documents (i.e., global context). To achieve it, the *word vs document* matrix is broken down into two matrices: document-topic and word-topic, where the former describes documents in terms of their topics and the later describes topics in terms of distributions over words in vocabulary.

Topic modeling can be used to classify or summarize documents based on the topics detected or to retrieve information based on topic similarities.

Related Studies in Topic Modeling

Latent Semantic Analysis (LSA) (Deerwester et al., 1990) is one of the earliest techniques in topic modeling, where the core idea is to take a document-term matrix and decompose it into a separate document-topic matrix and a topic-term matrix. LSA used tf-idf score to represent raw counts in the document-term matrix and then, singular value decomposition (SVD) to reduce dimensionality. Further, Probabilistic Latent Semantic Analysis (pLSA) (Hofmann, 1999) employed a stochastic method instead of SVD.

The Latent Dirichlet Allocation (LDA) (Blei et al., 2003) is one of the popular topic models that is a Bayesian version of pLSA. LDA models a document as a multinomial distribution over topics, where a topic is itself a multinomial

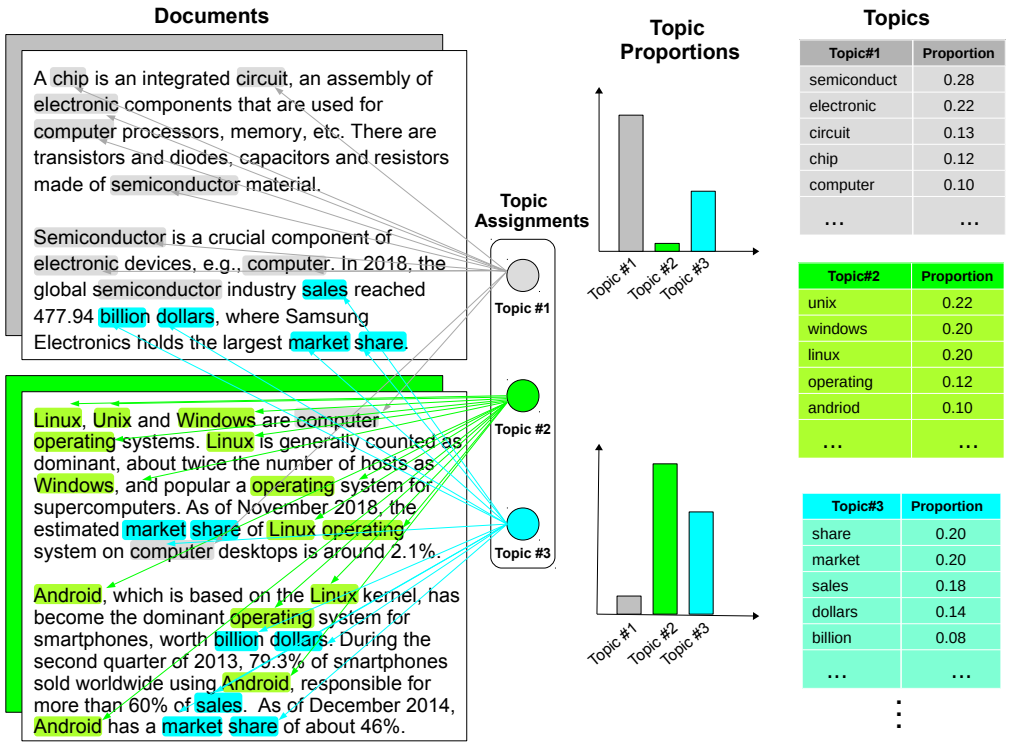


Figure 1.15 – *An illustration of the intuitions behind Latent Dirichlet Allocation (LDA) topic model, inspired by Blei (2012).*

distribution over words. While the distribution over topics is specific for each document, the topic-dependent distributions over words are shared across all documents. Thus, it can extract a semantic representation from a document by inferring its latent distribution over topics from the words it contains. For a new document, LDA provides its representation in terms of distribution over topics.

Figure 1.15 illustrates the intuition behind LDA where documents exhibit multiple topics (i.e., distributions over words) and the generative process assumes a fixed number of topics across the document collection. Each document is assumed to be generated as follows: (1) randomly choose a distribution over the topics (the histograms: topic proportions); (2) for each word, randomly choose a topic from the distribution over topics in step #1, i.e., topic assignments in the colored coins and then randomly choose the word from the corresponding topic. See Blei (2012) for a detailed discussion about the generative process of topic modeling.

Topic Modeling in Sparse-data Setting: Conventional topic modeling algorithms such as LDA infer document-topic and topic-word distributions from the co-occurrence of words within documents. However, learning representations remains challenging in the sparse-data settings with short texts and few documents,

since (1) limited word co-occurrences or little context, (2) significant word non-overlap in such short texts and (3) small training corpus of documents lead to little evidence for learning word co-occurrences. However, distributional word representations (i.e. word embeddings) (Pennington et al., 2014) have shown to capture both the semantic and syntactic relatedness in words and demonstrated impressive performance in NLP tasks.

For example, assume that the conventional topic model is run over the two short text fragments:

```
A lion catches a buffalo.
A tiger chases a cow.
```

The traditional topic models with bag-of-words assumption will not be able to infer relatedness due to the lack of word-overlap and/or small context in the two sentences. However, the pre-trained word embeddings (Pennington et al., 2014) as an external knowledge can help in expressing semantic relatedness in word pairs such as (`lion-tiger`, `catches-chases` and `buffalo-cow`) and therefore, improve topic models to generate more coherent topics.

Related work such as Sahami and Heilman (2006) employed web search results to improve the information in short texts and Petterson et al. (2010) introduced word similarity via thesauri and dictionaries into LDA. Das et al. (2015) and Nguyen et al. (2015a) integrated word embeddings into LDA and Dirichlet Multinomial Mixture (DMM) (Nigam et al., 2000) models.

Our Contribution: To alleviate the data sparsity issues, we extend the neural topic model, i.e., DocNADE[13] (section 1.3.4) by introducing pre-trained word embeddings as static priors and external knowledge while topic learning. The proposed model is named as *DocNADEe* (Gupta et al., 2019a), where the 'e' in the suffix refers to word *e*mbedding (pre-trained) vectors.

1.4.4 Local vs. Global Semantics

To extract word meanings, there is a hierarchy of views (or contexts): (1) context by a window-size, (2) sentence, (3) paragraph or (4) document.

The word embedding models such as skip-gram (Mikolov et al., 2013b), ELMo (Peters et al., 2018), etc. learn a vector-space representation for each word, based on the local word co-occurrences that are observed in a text corpus. Specifically, they rely on statistics about how each word occurs within a *context* of another word, where the context is either limited by a (short) window or a function of the sentence in which the word appears. Therefore, the word embedding models take a **local view**. In contrast, topic models take a **global view** in the sense that they

[13] since DocNADE (Larochelle and Lauly, 2012) outperformed LDA models in terms of generalization and performance on information retrieval task

infer topic distributions across a document collection and assign a topic to each word occurrence, where the assignment is equally dependent on all other words appearing in the same document. Therefore, they learn from word occurrences across documents via latent topics.

Word embedding models are promising at learning local semantic and syntactic relationships between words. Unlike word embedding models, topic models capture global semantics in the underlying corpus in form of topics and therefore, capture the long-range semantic meanings across the document. As a result, topic models can better deal with polysemous words based on the underlying themes in document collection.

Corpus of two documents D_1 and D_2
D_1: `There is a chip on the table.`
D_2: `There is a chip on the table. Integrated circuit is an assembly of electronic components that are used for computer processors, memory, etc. There are transistors and diodes, capacitors and resistors made of semiconductor material.`

Table 1.6 – *A corpus of two documents.*

For instance, consider a corpus of two documents D_1 and D_2 as illustrated in Table 1.6. Given the local view, it is difficult for the word embedding models to extract the meaning of the word 'chip': Does it mean a potato chip or an electronic chip? Therefore, the word embeddings from such models have inherently limited expressive power when it comes to global semantic information.

In contrast, a latent topic, for instance, 'electronics' is shared across the document collection and assigned to a word occurrence, e.g., `chip`. Additionally, the topic-word assignment (i.e., 'electronic'-`chip`) is equally dependent on all other words (e.g., 'electronic'-`circuit`, 'electronic'-`computer`, 'electronic'-`memory`, 'electronic'-`transistors`, 'electronic'-`semiconductor`, etc.). It means that the word `chip` occurs with other words related to 'electronics' within or across documents, leading to infer 'electronic' sense for the word `chip`. In a way, a topic model captures long-range dependencies (within or across documents) to resolve word meanings.

To summarize, the two learning paradigms are complementary in how they represent the meaning (*global* and *local*) of word occurrences, however distinctive in how they learn from statistical information observed in text corpora.

Our Contribution(s): To address data sparsity and polysemy issues, we present a novel approach of *Multi-view Transfer* (*MVT*) (Chapter 7) in the DocNADE (section 1.3.4) model that injects the two kinds of external knowledge: word embeddings (local semantics) and latent topics (global semantics) from many sources.

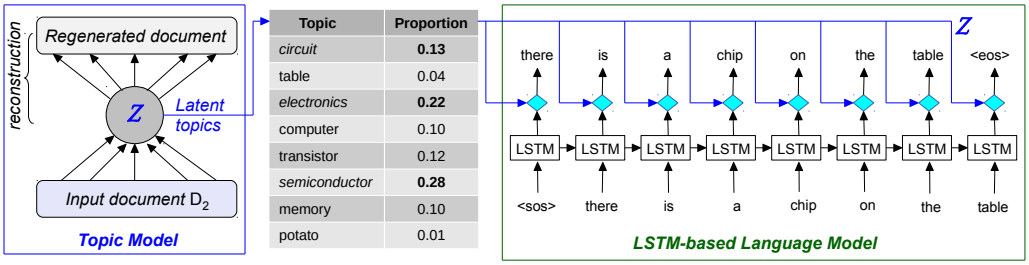


Figure 1.16 – *An overall architecture of a Composite model, consisting of a (neural) topic model and neural language model (LSTM). Observe that the latent topics* $\mathbf{Z}$ *are assimilated with hidden representations of LSTM to improve language modeling. Here, we show a single topic (a distribution over a vocabulary) about 'electronics'. The symbols <sos> and <eos> indicate the start and end of a sentence, respectively.*

1.4.5 Composite Models of Local and Global Semantics

As discussed in section 1.4.4, language models are typically applied at the sentence level, without access to the broader document context. Additionally, recent works in neural language models such as ELMo (Peters et al., 2018) are RNN-based language models and good at capturing the local structure (both semantic and syntactic) of a word sequence, however at the sentence level, under the assumption that sentences are independent of one another within and/or across documents (although related works such as Kiros et al. (2015) employed broader local context such as the preceding/following sentence(s)). In essence, they have a *local view*. Consequently, they can not model the long-term dependencies (Dieng et al., 2017), i.e., global semantics, since global statistical information is not exposed. In contrast, latent topic models take a *global view*, and tend to capture the global underlying semantic structure of a document but do not account for language structure, e.g., word ordering.

Given the merits of the two paradigms of learning complementary representations, an interesting research direction has opened up in recent time to jointly learn a topic and neural language model. Here, such a unified architecture accounting for both the global and local contexts is called as a *composite model*, where a topic model learns topical information in documents and a neural language model is designed to capture word relations within a sentence.

Illustration: Figure 1.16 illustrates an intuition of a composition model, consisting of a topic and language model to jointly learn the complementary representation via global and local views, respectively. Here, the topic model takes the complete document D_2 (Table 1.6), while an LSTM-based language model (LSTM-LM) takes a sentence (say the first one of D_2). Essentially during lan-

guage modeling, the probability distribution of each of the words in a sentence is conditioned on both the local and global contexts. Observe that the meaning of the word `chip` (in documents D_1 and D_2) can be resolved due to an explicit introduction of global semantics (e.g., 'electronic' sense) via latent topics $\mathbf{Z}$, captured by a topic model. Therefore, it helps improving the language modeling task.

Related studies: Major studies, such as TDLM (Lau et al., 2017), Topic-RNN (Dieng et al., 2017) and TCNLM (Wang et al., 2018) have integrated the merits of latent topic and neural language models (NLMs). Specifically, the related works in composite modeling learn the degree of influence of topical information on the neural language model. To do so, the NLM of the composite model incorporates topical information by assimilating the document-topic representation(s) with its hidden output at each time step. While the composite models focused on improving NLM explicitly via global (semantics) dependencies from latent topics, they do not investigate topic models by explicitly incorporating the local syntactic and semantic structures, captured by the NLM. Moreover, the related studies do not address the bag-of-words assumption in topic modeling.

Our Contribution(s): To this end, we have proposed a novel neural network-based *composite* model, which we named as *ctx-DocNADEe* (*textTOvec*) (Gupta et al., 2019b). It integrates the merits of a neural topic model i.e., DocNADE and a neural language model, i.e., LSTM-LM. In this work, we have attempted to introduce the local dynamics (syntactic and semantic structures) of the language into the global (i.e., topics) semantics of DocNADE, where the internal states of LSTM-LM encode the local dynamics. Consequently, we have shown to improve the latent topics due to the injection of language concepts, such as word ordering, latent syntactic and semantic structures, etc.

1.5 Semantic Relation Extraction (RE)

In the digital age, there is an information explosion in form of news, blogs, social media, email communications, governmental documents, chat logs, etc. Much of the information lies in unstructured form (Jurafsky and Martin, 2009) and it is difficult to extract relevant knowledge. This gives rise to Information extraction (IE) technologies that help humans to mine knowledge in a structured form in order to understand the data. Essentially, IE is a task of natural language processing that aims at turning unstructured text into a structured repository such as a relational table or knowledge base by annotating semantic information. It can extract meaningful facts from the web or unstructured text that can be used to populate knowledge bases (Bollacker et al., 2008; Auer et al., 2007) and in applications such as search, question-answering, etc.

Specifically, Information Extraction consists of subtasks such as named entity

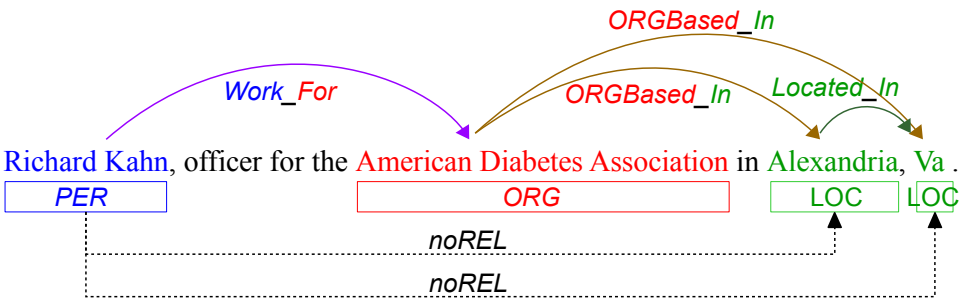


Figure 1.17 – *An illustration of Named Entity Recognition (NER) and Relation Extraction (RE), where the named entities are PER (person), ORG (organization) and LOC (location). noREL indicates the type 'noRelation'.*

recognition (NER), relation extraction (RE), event extraction, coreference resolution, etc. *Named entity recognition* is the task to identify all the mentions or occurrences of a particular named entity type in the given text, where a named entity is often a word or phrase that represents a specific real-world object, e.g., *person* (PER), *organization*, *location*, etc. NER is an important sub-problem, since it forms the basis for relation extraction. In this chapter, we focus on relation extraction task.

Relation extraction is the task to predict whether there exists a relation or not in any pair of entity mentions or nominals, and is modeled as binary classification problem. In other words, the relation arguments (i.e., the entity mentions) participating in a relation are not explicitly provided. In contrast, for a set of known relations, relation classification (RC) refers to predicting the relation type for a known pair of entity mentions participating in a relation. In the supervised settings, the relation extraction and classification tasks are combined by making a multi-class classification problem with an extra *noRelation* (*noREL*)[14] class and thus, they refer to the classification of an entity pair to a set of known relations, given the mentions (i.e., relation arguments) of the entity pair. In other words, relation extraction is treated as a classification task that detects and classifies predefined relationships between entities identified in the text. In this dissertation, we focus on binary relationships, where two arguments participate in a relation.

Figure 1.17 illustrates an example of relation extraction.

1.5.1 Intra- and Inter-sentential Relation Extraction

Based on the location of the mentions of entity pairs that participate in a relation mention, RE can be categorized as: (1) Intra-sentential RE (Figure 1.17), when

[14] a relation need not exist between every pair of named entity mentions in the given text

the mentions are located within the same sentence (2) Inter-sentential RE, when the mentions span sentence boundary. For instance, consider two sentences with entity mentions $e1$ and $e2$:

In 1975, Paul G. Allen co-founded [Microsoft]$_{e2}$ with Bill Gates. Later, [Steve Ballmer]$_{e1}$ became the CEO in 2000, who was hired by Bill Gates in 1980.

The two sentences together convey the fact that the entity $e1$ is associated with $e2$ by the relation type, *Work_for*(*PER-ORG*) and they occur in different sentences. This relationship cannot be inferred from either sentence alone using intra-sentential RE approaches, leading to poor system recall.

Progress in relation extraction is exciting; however most prior works (Kambhatla, 2004; Bunescu and Mooney, 2005; Zhang et al., 2006b; Mesquita et al., 2013; Miwa and Sasaki, 2014; Nguyen and Grishman, 2015; Vu et al., 2016a; Gupta et al., 2016, 2018c) in RE are limited to intra-sentential relationships, and ignore relations in entity pairs spanning sentence boundaries. Previous works on cross-sentence relation extraction such as Gerber and Chai (2010) and Yoshikawa et al. (2011) used coreferences to access entities that occur in a different sentence without modeling inter-sentential relational patterns. Swampillai and Stevenson (2011) described a SVM-based approach to both intra- and inter-sentential relations. Recently, Quirk and Poon (2017) applied distant supervision to cross-sentence relation extraction of entities using binary logistic regression (non-neural network based) classifier and Peng et al. (2017) applied sophisticated graph long short-term memory networks to cross-sentence n-ary relation extraction. However, it still remains challenging due to the need for coreference resolution, noisy text between the entity pairs spanning multiple sentences and lack of labeled corpora.

There are two types of Relation Extraction systems: Closed domain[15] relation extraction systems consider only a closed set of relationships between two arguments. On the other hand, Open-domain relation extraction systems (Banko et al., 2007; Etzioni et al., 2005; Soderland et al., 2010; Etzioni et al., 2011; Gamallo et al., 2012; Zhu et al., 2019) use an arbitrary phrase to specify a relationship. In this context, we focus on the former.

Our Contribution(s): In our efforts to push relation extraction beyond sentence boundary, we present a dependency-based neural network, which we named as *inter-sentential Dependency-based Neural Network* (*iDepNN*) (Gupta et al., 2019c) that precisely extracts relationships within and across sentence boundaries using recurrent and recursive neural networks over dependency parse features, and demonstrates a better balance in precision and recall with an improved F_1 score.

[15]The set of relations of interest has to be named by the human user in advance

Related Work	E2E	classifier	Summary
Intra-sentential Supervised Relation Extraction			
Lee et al. (2019)		LSTM	Entity-aware Attention
Zhang et al. (2018)		GCN	DPT features + pruning
Zheng et al. (2017b)	✓	LSTM	Joint NER and RE as a tagging problem
Zhang et al. (2017)		LSTM	Entity position-aware attention
Adel and Schütze (2017)	✓	CNN+CRF	Global normalization of entity-relation scores
Gupta et al. (2016)	✓	RNN	Table Filling RNN architecture
Miwa and Bansal (2016)	✓	LSTM+RecvNN	LSTM on sequence and TreeLSTM on DPT
Cai et al. (2016)		LSTM+CNN	DPT features
Xiao and Liu (2016)		RNN	Hierarchical RNN with Attention
Zhou et al. (2016)		LSTM	Attention Bi-LSTM
Wang et al. (2016a)		CNN	Multi-Level Attention
Shen and Huang (2016)		CNN	Attention-Based CNN
Xu et al. (2016)		RNN	DPT features and data augmentation
Vu et al. (2016a)		CNN+RNN	Ensemble of RNN and CNN
Xu et al. (2015a)		CNN	DPT features with negative sampling
Gupta et al. (2015a)		RNN	Connectionist bidirectional RNN
Xu et al. (2015b)		LSTM	Shortest Dependency Paths (SDP)
Liu et al. (2015a)		CNN+RecvNN	SDP and Augmented Subtrees
Miwa and Sasaki (2014)	✓	SVM	Table filling approach, lexical features, SDP
Yu et al. (2014)		MLP	Linguistic contexts and word embeddings
Zeng et al. (2014)		CNN	Deep CNN
Socher et al. (2012)		RecvNN	Matrix-vector representations of parse tree
Rink and Harabagiu (2010)		SVM	Lexical, syntactic and semantic features
Kate and Mooney (2010)	✓	SVM	Lexical+syntactic features, Card-Pyramid Parsing
Roth and Yih (2007)	✓	HMM	Lexical+syntactic features, ILP for global inference
Inter-sentential Supervised Relation Extraction			
Singh and Bhatia (2019)	✓	Transformer+MLP	Modeling via second order relations
Gupta et al. (2019c)		RNN+RecvNN	RNN on SDP and TreeRNN on Subtrees
Peng et al. (2017)		GraphLSTM	N-ary RE with DPT and co-reference features
Swampillai et al. (2011)		SVM	DPT, lexical and syntactic features
Intra-sentential Distantly Supervised Relation Extraction			
Vashishth et al. (2018)		GCN	Additional entity type and relation aliases
Lin et al. (2016)		CNN	Remove noisy instances via selective attention
Surdeanu et al. (2012)		logistic	Multi-instance Multi-label Learning
Hoffmann et al. (2011)		-	Multi-instance learning with overlapping relations
Mintz et al. (2009a)		logistic	Distant supervision for RE without labeled data

Table 1.7 – *Summary of the evolution of (Distantly) Supervised Relation Extraction systems. Abbreviations are: E2E: End-to-End (Joint NER and RE) systems. DPT: Dependency parse tree, ILP: Integer Linear Programming, RecvNN: Recursive Neural Network, GCN: Graph Convolution Network*

1.5.2 Supervised Relation Extraction

Supervised relation extraction approaches are popular, where they require carefully designed labeled data such that each pair of entity mentions is labeled with one of the pre-defined relation types. A special relation type *noREL* is introduced to label the pairs that do not hold any of the pre-defined relation types. The data preparation is laborious task in the sense that the human annotations can be time-consuming and costly.

Essentially, the supervised RE is formulated as a multi-class classification problem where it requires a large human annotated training corpus. Approaches to supervised RE are broadly classified into two types: (1) *Feature-based* (Kambhatla, 2004; Zhou et al., 2005; Jiang and Zhai, 2007) methods that rely on a set of relevant features[16] (lexical, syntactic and semantic) designed by domain experts and/or using lexical resources like WordNet (Fellbaum, 1998) and then, a classifier is trained using these features, and (2) *Kernel-based* (Lodhi et al., 2002; Bunescu and Mooney, 2005; Zhang et al., 2006a) methods that compute similarities between representations (sequences, syntactic parse trees etc.) of two relation instances using SVM (Support Vector Machine) (Byun and Lee, 2002) as classifier. Unlike Feature-based methods, the kernel based methods avoid explicit feature engineering.

See Pawar et al. (2017) for a comprehensive survey on relation extraction, outlining different feature-based and kernel-based approaches.

Recently in supervised settings, deep learning based methods such as CNN (Nguyen and Grishman, 2015; dos Santos et al., 2015; Vu et al., 2016a; Le et al., 2018), RNN (Liu et al., 2015b; Zhang and Wang, 2015; Gupta et al., 2015a, 2016; Lin et al., 2016; Miwa and Bansal, 2016; Peng et al., 2017; Gupta and Schütze, 2018; Gupta et al., 2019c), etc. have shown promising results in relation extraction task, in comparison to the traditional RE models that rely on hand-crafted features.

Supervised techniques for machine learning require large amount of training data for learning. Using manually annotated datasets for relation extraction is expensive in terms of time and effort.

Table 1.7 summarizes the evolution of supervised relation extraction systems.

1.5.3 Distantly Supervised Relation Extraction

In order to deal with the unavailability of large labeled data for training supervised methods, Mintz et al. (2009a) proposed a distant supervision (DS) method that produces a large amount of training data by aligning knowledge base facts with unstructured texts; therefore it does not require labeled data. The term 'distant' is used in the sense that no explicit labeled data is provided, however a knowledge

[16] see Kambhatla (2004) for various feature types

base such as Wikipedia or Freebase is used to automatically tag training examples from the text corpora. Distant Supervision combines advantages of both the paradigms: supervised and unsupervised, where such large annotated training examples are used to train supervised systems such as CNNs, LSTMs, etc.

Specifically, the distant supervision uses a knowledge base(s) (KB) (e.g. Freebase (Bollacker et al., 2008)) to find pair of entities $e1$ and $e2$ for which a relation r holds. It assumes that if the relation r exists between the entity pair in the KB, then every document containing the mention of this entity pair would express that relation r. Unfortunately, the assumption leads to large proportion of false positive i.e., noisy training instances because every document containing the entity pair mention may not express the relation r between the pair. Therefore, to reduce the noise in the training instances generated by DS approach, Riedel et al. (2010) proposed to relax the DS assumption by modeling the problem as a Multi-instance learning (a form of supervised learning where a label is given to a bag of instances, instead of a single instance). Further, Hoffmann et al. (2011) extended multi-instance learning to deal with overlapping relations and Surdeanu et al. (2012) proposed MIML-RE (Multi-Instance Multi-Labeling Relation Extraction) system that models latent relation labels for multiple instances (occurrences) of an entity pair.

Moreover, Zeng et al. (2015) modeled MIML-RE paradigm in a neural network (i.e., Piecewise CNN) architecture, however only using the one most-relevant sentence from the bag. Lin et al. (2016) used an attention mechanism over all the instances in the bag for the multi-instance problem, that is further extended by Jiang et al. (2016) by applying a cross-document max-pooling layer in order to address information loss as in the previous models.

Table 1.7 summarizes some of the major distantly supervised RE systems.

1.5.4 Weakly Supervised Relation Extraction: Bootstrapping

While the labeled data is lacking and annotation procedure is expensive, the distant supervised RE approaches exploit existing knowledge bases. However, they introduce noise in training data and can not be applied when the relation of interest is not covered explicitly by the KB. In such a scenario, bootstrapping techniques (Riloff, 1996; Brin, 1998; Agichtein and Gravano, 2000; Ravichandran and Hovy, 2002; Bunescu and Mooney, 2005; Batista et al., 2015) are desirable that require a very small degree of supervision in form of seed instances or patterns for starting the self-learning process to extract more instances from a large unlabeled corpus. In an iterative fashion, the initial seeds extract a new set of patterns which extract more instances that further extract more patterns and so on. Due to lack of labeled data, the bootstrapping approaches often suffer from low precision and semantic drift (if a false positive instance is added during an iteration, then all following

Related Work	Summary
JBM or BREX (Gupta et al., 2018c)	closed-domain RE, use word embeddings in pattern/phrase representation, seed type: tuples and templates, joint scheme of bootstrapping with two types of seed instances, control semantic drift with improve confidence measure
BREDS (Batista et al., 2015)	closed-domain RE, word embeddings in phrase representation, seed type: tuples
TextRunner (Banko et al., 2007)	open-domain RE, relies on a dependency parser, self-learning and self-labeling training data
KnowItAll (Etzioni et al., 2005)	closed-domain RE, use generic patterns to learn domain-specific extraction rules
DIPRE (Brin, 1998)	closed domain RE, uses NER, flexible pattern matching, pattern and tuple evaluation via confidence scores, seed type: tuples
Snowball (Agichtein and Gravano, 2000)	closed-domain RE, string matching over regular expression, hard pattern matching, seed type: tuples

Table 1.8 – *A summary of the evolution of Semi-supervised Bootstrapping Relation Extraction systems.*

iterations are contaminated).

Specifically, a bootstrapping relation extractor is fed with a few seed instances (e.g., <*Google*, *YouTube*>) of the relation type of interest (e.g., '*acquisition*') to extract pattern mentions that express the relation in the large unlabeled corpora. Given the patterns, a set of new entity pairs having the same relation type is extracted (e.g., <*Google*, *DeepMind*>, <*Microsoft*, *LinkedIn*>, <*Siemens*, *Mentor Graphics*>, <*Facebook*, *WhatsApp*>, etc.).

Related Studies in Bootstrapping RE: In initial efforts, DIPRE (Dual Iterative Pattern Relation Expansion) (Brin, 1998) was the first bootstrapping based relation extractor that used string-based regular expressions in order to recognize relations, while the SNOWBALL system (Agichtein and Gravano, 2000) learned similar regular expression patterns over words and named entity tags. Unlike DIPRE, the SNOWBALL has a flexible matching system. For instance, two patterns in DIPRE are different if they only differ by a single punctuation. SNOWBALL further extended DIPRE by introducing patterns and tuples evaluation by computing their confidence scores. Recently, Batista et al. (2015) advances SNOWBALL system by introducing word embeddings (Mikolov et al., 2013c) to represent phrases or patterns in order to improve similarity computations via the distributed representations (Mikolov et al., 2013b; Le and Mikolov, 2014).

Limitations of the Related Studies: The state-of-the-art relation extractors

(Brin, 1998; Agichtein and Gravano, 2000; Batista et al., 2015) bootstrap with only seed entity pairs and suffer due to a surplus of unknown extractions (relation instances or entity pairs). While computing confidence of patterns and tuples, the traditional bootstrapping algorithm erroneously penalizes them due to the lack of labeled data. This in turn leads to low confidence in the system output and hence negatively affects recall. The prior systems do not focus on improving the pattern scores. Additionally, SNOWBALL and BREDS used a weighting scheme to incorporate the importance of contexts around entities and compute a similarity score that introduces additional parameters and does not generalize well.

Our Contribution(s): To alleviate the issue, we introduce BREX (or JBM: Joint Bootstrapping Machine) (Gupta et al., 2018c), a new bootstrapping method that protects against such contamination by highly effective confidence assessments. This is achieved by using entity and template seeds jointly (as opposed to just one as in previous work) by (1) expanding entities and templates in parallel and in a mutually constraining fashion in each iteration, and (2) introducing higher quality similarity measures for templates.

Table 1.8 summarizes some of the major semi-supervised bootstrapping relation extraction systems.

1.5.5 Unsupervised Relation Extraction

Generally, purely unsupervised relation extraction systems extract strings of words between given named entities in large amounts of text, and cluster and simplify these word strings to produce relation-strings (Feldman and Rosenfeld, 2006; Banko et al., 2007; Sun et al., 2011). Such systems extract a large number of relations, where it is difficult to map resulting relations to a particular relation of interest in the knowledge base.

Major prior works in unsupervised relation extractors are based on clustering approaches such as Hasegawa et al. (2004) and Poon and Domingos (2009) that only require a NER tagger to identify named entities in the text, cluster the contexts of the co-occurring named entity pairs and automatically label each of the clusters, representing one relation type. Feldman and Rosenfeld (2006) proposed a non-clustering based approach for unsupervised relation extraction that only requires definitions of the relation types of interest in form of a small set of keywords, indicative of that relation type and entity types of its arguments. Based on the relation specific keywords, it extracts relation facts from the web.

1.5.6 Joint Entity and Relation Extraction

The supervised relation extraction systems (discussed in sections 1.5.2 and 1.5.3) assume that entity boundary and its type is provided beforehand. However, the

approaches can not be used when such a prior knowledge about entity mentions is not supplied. Therefore, most approaches split it into two independent tasks and models them sequentially i.e., 'pipeline' of named entity recognition (NER) and relation extraction (RE). Such techniques are vulnerable to propagation of errors from NER to RE. Additionally, the two tasks are mutually dependent in the sense that given the entity mentions, the search space of relation can be reduced and vice-versa. Also, given a relation for the participating arguments, it helps to better disambiguate the entity types of the arguments.

To avoid error propagation in sequential modeling and exploit the dependence of entity mentions and relations, there is a line of research that jointly models/extracts entities and relations. They are also known as *end-to-end* models.

Related Studies in End-to-End RE: Most of the joint entity and relation models[17] such as Li and Ji (2014), Miwa and Sasaki (2014), Roth and Yih (2007), Kate and Mooney (2010), Yu and Lam (2010) and Singh et al. (2013) are *feature-based structured learning* and do not employ neural networks. Specifically, Roth and Yih (2004) applied global normalization of entity types and relations using constraints in integer linear programming based approach. They first learn independent local classifiers for entity and relation extractions. During inference for a given sentence, a global decision is made to satisfy the domain-specific or task-specific constraints. Roth and Yih (2007) made the first attempt using graphical models to jointly learn entities and relations. Further, Kate and Mooney (2010) proposed an interesting approach via card-pyramid parsing and a graph encoding the mutual dependencies among the entities and relations. Miwa and Sasaki (2014) used a table structure to represent the mutual dependence between entity and relations in a sentence.

With the emergence of *neural networks* and their success in relation extraction, recent works have employed neural network based approaches to jointly model entities and relations. To this end, Miwa and Bansal (2016) proposed a neural network based approach for the end-to-end relation extraction, where a bidirectional tree-structured LSTM captures relations while a bidirectional sequential LSTM extracts entities. They stacked the treeLSTM over sequence LSTM, resulting in a unified network that shares parameters during the joint modeling of entities and relations. Zheng et al. (2017a) proposed a hybrid neural network consisting of a bidirectional encoder-decoder LSTM for entity extraction and a CNN module for relation classification. Moreover, Bekoulis et al. (2018) framed the problem as a multi-head[18] selection problem by using a sigmoid loss to obtain multiple relations and a CRF loss for the NER component. In contrast to joint training and

[17] In most of the approaches for joint extraction of entities and relations, it is assumed that the boundaries of the entity mentions are known

[18] any particular entity may be involved in multiple relations with other entities

multi-task learning, Adel and Schütze (2017) proposed to jointly model entities and relations by a joint classification layer that is globally normalized on the outputs of the two tasks using a linear-chain conditional random field (CRF) (Lafferty et al., 2001) on the top of continuous representations obtained by a CNN model.

All the related works discussed above are based on intra-sentential relationships. However, a recent work by Singh and Bhatia (2019) introduced joint entity and relation extraction approach that spans sentence boundaries by modeling second order relationships.

Our Contribution(s): In our efforts to jointly model entities and relations within a sentence boundary, we present a novel neural network architecture, which we named as *Table-filing Multi-task RNN* (*TF-MTRNN*) (Gupta et al., 2016). Specifically for a given sentence, the proposed model fills an entity-relation table structure (Miwa and Sasaki, 2014) via a recurrent neural network and consequently, performs entity and relation extractions jointly in a unified network by sharing parameters in a multi-task learning (Miwa and Sasaki, 2014) setup.

Table 1.7 summarizes the evolution of major end-to-end (joint entity and relation) relation extraction systems.

1.6 BlackBoxNLP: Interpreting and Analyzing Neural Networks

Recently, neural networks have shown impressive success in the field of natural language processing. However, they come at the cost of human understanding in contrast to feature-rich systems. The feature-based systems are relatively transparent since one can analyze the importance of certain features, such as morphological properties, lexical classes, syntactic categories, semantic relations, etc. in order to achieve a better understanding of the model.

The goal of interpretability is to summarize the reasons for neural network behavior, gain the trust of users and produce insights about the causes of their decisions in a way that is understandable to humans.

1.6.1 Why Interpret and Analyze Neural NLP models?

In the following section, we discuss different aspects of why we need explainable models:

1. To explain and verify that the model behaves as expected in order to avoid incorrect decision that can be costly, for instance in medical domain where precision is the desired objective

2. To gain new insights about the model behavior that would support better interactions with the systems and thus, improve them with human involvement

3. To identify potential flaws or biases in the training data and ensure algorithmic fairness

4. To comply with proposed legislation (e.g., "European General Data Protection Regulation"), assign accountability and build trust, fairness, safety and reliability (Doshi-Velez and Kim, 2017; Lipton, 2018)

5. To understand how linguistic concepts are captured in the neural networks, e.g., what happens when they take in word embeddings as input and generate some output

1.6.2 How to Interpret and Analyze Neural NLP models?

Techniques[19] in explaining and interpreting neural models mostly focus on either (1) *Interpretable models*, i.e., "What information does the network contain in its internal structure?" , or (2) *Explaining decisions*, i.e., "Why does a particular input lead to a particular output?"

Approaches in *Interpretable models* attempt to explain the internal structures of the neural network system. For instance, Erhan et al. (2009), Simonyan et al. (2013) and Nguyen et al. (2016) have focused on model aspects that investigate neural network components and thus, studied the behavior of neurons/activation in order to understand neural networks' behavior. They used *activation maximization* approach that finds patterns (or inputs) maximizing the activity of given neurons. Belinkov and Glass (2018) summarized a long line of research that attempts to analyze different kinds of linguistic information such as sentence length, word position, word presence, or simple word order, morphological, syntactic, semantic information, etc.

Another line of research in *Interpretable models* focuses on data generation aspects that attempts to understand neural network models by generating adversarial examples (Goodfellow et al., 2014). In NLP domain, , the adversarial examples are often inspired by text edits (Sakaguchi et al., 2017; Heigold et al., 2018; Belinkov and Bisk, 2017; Ebrahimi et al., 2018) in form of typos, misspellings, similar word substitutions, etc. Related approaches (Papernot et al., 2017; Narodytska and Kasiviswanathan, 2017; Alzantot et al., 2018) interpret neural models via understanding their failures.

[19] inspired by `iphome.hhi.de/samek/pdf/DTUSummerSchool2017_1.pdf`

Alternative approaches in *explaining decisions* focus on understanding the input-output association to explain which input contributes to classification and which input leads to increases or decreases in prediction score when changed, i.e., *sensitivity analysis*. Bach et al. (2015) proposed a methodology to understand classification decisions by pixel-wise decomposition for multi-layered neural networks to understand the contribution of a single pixel of an image to the prediction made by the classifier. They also visualized the contributions of single pixels to predictions using heat maps. However, this work was focused on vision domain. Recently, (Pörner et al., 2018) introduced techniques to quantitatively evaluate methods, such as (Bach et al., 2015; Ribeiro et al., 2016; Shrikumar et al., 2017) explaining the decisions of neural models.

Visualization is a valuable tool for analyzing neural networks in NLP. Recent studies (Ming et al., 2017; Gupta and Schütze, 2018) in sensitivity analysis for NLP domain have investigated visualization of RNN and its variants. Li et al. (2016) employed heat maps to study sensitivity and meaning composition in recurrent networks for given words in a sentence. Tang et al. (2017) visualized the memory vectors to understand the behavior of an LSTM and gated recurrent unit (GRU) in speech recognition task. Ming et al. (2017) proposed a tool, RNNVis to visualize hidden states based on an RNN's expected response to inputs.

Since, we have mostly employed RNN-based models in our works such as in entity extraction (Gupta et al., 2016), relation extraction (Gupta et al., 2015a; Vu et al., 2016a; Gupta et al., 2016, 2019c), textual similarity (Gupta et al., 2018a) and dynamic topic modeling (Gupta et al., 2018b), therefore we attempt to interpret and analyze RNN models, especially for relation classification task.

Our Contribution(s): We present a technique, which we named as *Layer-wIse-Semantic-Accumulation* (*LISA*) (Gupta and Schütze, 2018) that analyzes the cumulative nature of an RNN for explaining decisions and detecting the most likely (i.e., saliency) patterns that the network relies on while decision making. We further demonstrate (1) how an RNN accumulates or builds semantics during its sequential processing for a given text example and expected response and (2) how the saliency patterns look like for each category in the data according to the network in decision making. We also analyze the sensitiveness of RNNs about different inputs to check the increase or decrease in prediction scores and extract the saliency patterns, learned by the network. In doing so, we employ relation classification datasets.

1.7 Summary

In this introductory chapter, we have first introduced some of the supervised neural networks, including Recurrent (RNNs), Recursive (RecvNNs) and Siamese

(SNNs) Neural Networks. Then, we have described the unsupervised paradigm of learning representations via neural density estimators, especially Restricted Boltzmann Machine (RBM), Neural Autoregressive Distributional Estimation (NADE), Replicated Softmax (RSM) and Neural Autoregressive Topic Model (DocNADE). In context of this dissertation, this class of stochastic graphical models forms the basis for neural topic learning in text documents. Following the two paradigms in neural network learning, the next section has highlighted the foundation of distributed representation learning at word and document levels. Moreover, it has underlined a need for joint leaning in a composite model, consisting of a topic and a neural language model. Then, we have provided an outline for the task of semantic relation extraction (RE) within (intra-) and across (inter-) sentence boundary, where we have also featured major related works in the realms of relation extraction as well as joint entity and relation extraction. Finally, we have reviewed recent questions in explainability of neural network models.

While we are discussing the basic fundamentals in this chapter, we have briefly highlighted our contribution(s) in its corresponding sections.

Chapter 2

Table Filling Multi-Task Recurrent Neural Network for Joint Entity and Relation Extraction

Table Filling Multi-Task Recurrent Neural Network for Joint Entity and Relation Extraction

Pankaj Gupta
Corporate Technology, Siemens AG
CIS, University of Munich (LMU)
Munich, Germany
gupta.pankaj.ext@siemens.com
pankaj.gupta@campus.lmu.de

Hinrich Schütze
CIS, University of Munich (LMU)
inquiries@cis.lmu.org

Bernt Andrassy
Corporate Technology, Siemens AG
Munich, Germany
bernt.andrassy@siemens.com

Abstract

This paper proposes a novel context-aware joint entity and word-level relation extraction approach through semantic composition of words, introducing a Table Filling Multi-Task Recurrent Neural Network (TF-MTRNN) model that reduces the entity recognition and relation classification tasks to a table-filling problem and models their interdependencies. The proposed neural network architecture is capable of modeling multiple relation instances without knowing the corresponding relation arguments in a sentence. The experimental results show that a simple approach of piggybacking candidate entities to model the label dependencies from relations to entities improves performance.

We present state-of-the-art results with improvements of 2.0% and 2.7% for entity recognition and relation classification, respectively on CoNLL04 dataset.

1 Introduction

Relation classification is defined as the task of predicting the semantic relation between the annotated pairs of nominals (also known as relation arguments). These annotations, for example named entity pairs participating in a relation are often difficult to obtain. Traditional methods are often based on a pipeline of two separate subtasks: Entity Recognition (ER[1]) and Relation Classification (RC), to first detect the named entities and then performing relation classification on the detected entity mentions, therefore ignoring the underlying interdependencies and propagating errors from the entity recognition to relation classification. The two subtasks together are known as End-to-End relation extraction.

Relation classification is treated as a sentence-level multi-class classification problem, which often assume a single relation instance in the sentence. It is often assumed that entity recognition affects the relation classification, but it is not affected by relation classification. Here, we reason with experimental evidences that the latter is not true. For example, in Figure 1, relation *Work_For* exists between *PER* and *ORG* entities, *ORGBased_in* between *ORG* and *LOC*, while *Located_In* between *LOC* and *LOC* entities. Inversely, for a given word with associated relation(s), the candidate entity types can be detected. For example, in Figure 2, for a given relation, say *Located_in*, the candidate entity pair is (*LOC*, *LOC*). Therefore, the two tasks are interdependent and optimising a single network for ER and RC to model the interdependencies in the candidate entity pairs and corresponding relations is achieved via the proposed joint modeling of subtasks and a simple piggybacking approach.

Joint learning approaches (Roth and Yih, 2004; Kate and Mooney, 2010) built joint models upon complex multiple individual models for the subtasks. (Miwa and Sasaki, 2014) proposed a joint entity and relation extraction approach using a history-based structured learning with a table representation; however, they explicitly incorporate entity-relation label interdependencies, use complex features and search heuristics to fill table. In addition, their state-of-the-art method is structured prediction and not based on neural network frameworks. However, *deep learning* methods such as recurrent and convolutional neural networks (Zeng et al., 2014; Zhang and Wang, 2015; Nguyen and Grishman, 2015) treat relation

[1]Entity Recognition (ER) = Entity Extraction (EE); Relation Classification (RC) = Relation Extraction (RE)

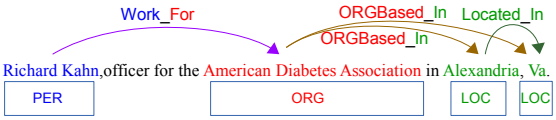


Figure 1: An entity and relation example (CoNLL04 data). *PER*: Person, *ORG*: Organization, *LOC*: Location. Connections are: *PER* and *ORG* by *Work_For*; *ORG* and *LOC* by *OrgBased_In*; *LOC* and *LOC* by *Located_In* relations.

	PER	**LOC**	**ORG**	**Other**
PER	KILL	Live_In	Work_For	⊥
LOC	Live_In	Located _In	ORG Based_In	⊥
ORG	Work_For	ORG Based_In	⊥	⊥
Other	⊥	⊥	⊥	⊥

Figure 2: Entity-Relation dependencies (CoNLL04 dataset).

	Richard	Kahn	,	officer	for	the	American	Diabetes	Association	in	Alexandria	,	Va	.
Richard	*B-PER*, ⊥													
Kahn	⊥	*L-PER*, ⊥												
,	⊥	⊥	*O*, ⊥											
officer	⊥	⊥	⊥	*O*, ⊥										
for	⊥	⊥	⊥	⊥	*O*, ⊥									
the	⊥	⊥	⊥	⊥	⊥	*O*, ⊥								
Ameriacan	⊥	⊥	⊥	⊥	⊥	⊥	*B-ORG*, ⊥							
Diabetes	⊥	⊥	⊥	⊥	⊥	⊥	⊥	*I-ORG*, ⊥						
Association	⊥	*Work_For*	⊥	⊥	⊥	⊥	⊥	⊥	*L-ORG*, ⊥					
in	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	*O*, ⊥				
Alexandria	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	*ORGBased_In*	⊥	*U-LOC*, ⊥			
,	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	*O*, ⊥		
Va	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	*ORGBased_In*	⊥	*Located_In*	⊥	*U-LOC*, ⊥	
.	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	⊥	*O*, ⊥

Table 1: Entity-Relation Table for the example in Figure 1. Demonstrates the word-level relation classification via a Table-Filling problem. The symbol (⊥) indicates *no_relation* word pair. Relations are defined on the words, instead of entities. The diagonal entries have the entity types and ⊥ relations to the words itself, while the off-diagonal entries are the relation types.

classification as a sentence-level multi-class classification, and rely on the relation arguments provided in the sentence. Therefore, they are incapable in handling multiple relation instances in a sentence and can not detect corresponding entity mention pairs participating in the relation detected.

We tackle the limitations of joint and deep learning methods to detect entities and relations. The contributions of this paper are as follows:

1. We propose a novel Table Filling Multi-task Recurrent Neural Network to jointly model entity recognition and relation classification tasks via a unified multi-task recurrent neural network. We detect both entity mention pairs and the corresponding relations in a single framework with an entity-relation table representation. It alleviates the need of search heuristics and explicit entity and relation label dependencies in joint entity and relation learning. As far as we know, it is the first attempt to jointly model the interdependencies in entity and relation extraction tasks via multi-task recurrent neural networks.

   We present a word-level instead sentence-level relation learning via word-pair compositions utilising their contexts via Context-aware RNN framework. Our approach has significant advantage over state-of-the-art methods such as CNN and RNN for relation classification, since we do not need the marked nominals and can model multiple relation instances in a sentence.

2. Having named-entity labels is very informative for finding the relation type between them, and vice versa having the relation type between words eases problem of named-entity tagging. Therefore, a simple approach to piggyback candidate named entities for words (derived from the associated relation type(s) for each word) to model label dependencies improves the performance of our system. In addition, the sequential learning approach in the proposed network learns entity and relation label dependencies via sharing model parameters and representations, instead modeling them explicitly.

3. Our approach outperforms the state-of-the-art method by 2.0% and 2.7% for entity recognition and relation classification, respectively on CoNLL04 dataset.

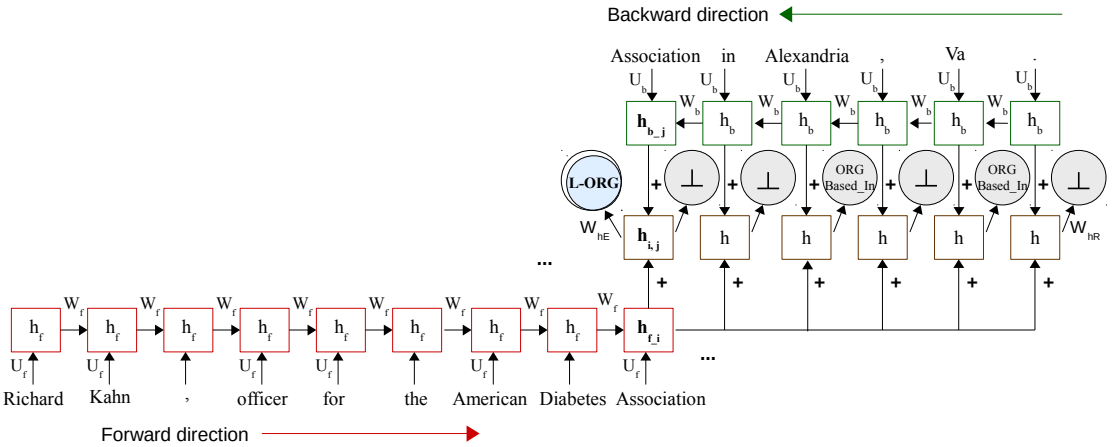


Figure 3: The Table Filling Multi-Task Recurrent Neural Network (TF-MTRNN) for joint entity and word-level relation extraction. Overlapping circle: Entity labels; Single circle: Relation label. In the above illustration, the word *Association* at $t = i$ (where; $t = 0, ..., i, ..., N$) from forward network is combined with each of the remaining words in the sequence (Figure 1), obtained from backward network at each time step, $j = i, ..., N$. Similarly, perform all possible word pair compositions to obtain Table 1. *ORGBased_In* relation in each word-pairs: (*Association*, *Alexandria*) and (*Association*, *Va*).

2 Methodology

2.1 Entity-Relation Table

As the backbone of our model we adopt the table structure proposed by Miwa and Sasaki (2014), shown in Table 1. This structure allows an elegant formalization of joint entity and relation extraction because both entity and relation labels are defined as instances of binary relations between words w_i and w_j in the sentence. An entity label is such a binary relationship for $i = j$, i.e., a cell on the diagonal. A relation label is such a binary relationship for $i \neq j$, i.e., an off-diagonal cell. To eliminate redundancy, we stipulate that the correct label for the pair (w_i, w_j) is relation label r if and only if $i \neq j$, w_i is the last word of a named entity e_i, w_j is the last word of a named entity e_j and $r(e_i, e_j)$ is true.[2] We introduce the special symbol $\perp$ for "no_relation", i.e., no relation holds between two words.

Apart from the fact that it provides a common framework for entity and relation labels, another advantage of the table structure is that modeling multiple relations per sentence comes for free. It simply corresponds to several (more than one) off-diagonal cells being labeled with the corresponding relations.

2.2 The Table Filling Multi-Task RNN Model

Formally, our task for a sentence of length n is to label $n(n+1)/2$ cells. The challenge is that the labeling decisions are highly interdependent. We take a deep learning approach since deep learning models have recently had success in modeling complex dependencies in NLP. More specifically, we apply recurrent neural networks (RNNs) (Elman, 1990; Jordan, 1986; Werbos, 1990) due to their success on complex NLP tasks like machine translation and reasoning.

To apply RNNs, we order the cells of the table into a sequence as indicated in Figure 4 and label – or "fill" – the cells one by one in the order of the sequence. We call this approach *table filling*.

More specifically, we use a bidirectional architecture (Vu et al., 2016b), a forward RNN and a backward RNN, to fill each cell (i, j) as shown in Figure 3. The forward RNN provides a representation of the history $w_1, \ldots, w_i$. The backward network provides a representation of the following context $w_j, \ldots, w_{|s|}$. The figure shows how the named entity tag for "Association" is computed. The forward RNN is shown as the sequence at the bottom. h_{f_i} is the representation of the history and h_{b_j} is the representation of the following context. Both are fed into $h_{i,j}$ which then predicts the label L-ORG. In this case, $i = j$. The prediction of a relation label is similar, except that in that case $i \neq j$.

[2]Relation types (excluding $\perp$) exist only in the word pairs with entity types: (L-*, L-*), (L-*, U-*), (U-*, L-*) or (U-*, U-*), where * indicates any entity type encoded in BILOU (Begin, Inside, Last, Outside, Unit) scheme.

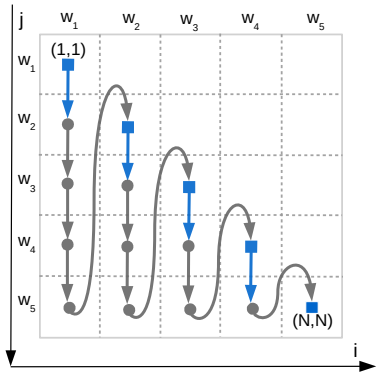


Figure 4: Table Filling/Decoding Order. Filled squares in blue represent both entity and relation label assignments, while filled circles in gray represent only relation label assignments, analogous to entries in Table 1. (i, j) is the cell index in the table, where i and j are the word indices in the sequence.

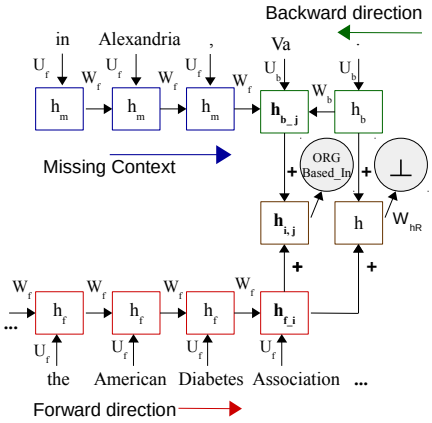


Figure 5: The context-aware TF-MTRNN model. (...) indicates the remaining word pair compositions (Table 1).

Our proposed RNN based framework jointly models the entity and relation extraction tasks to learn the correlations between them, by sharing the model parameters and representations. As illustrated in Figure 3, we use two separate output nodes and weight matrices each for entity and relation classification. An entity label is assigned to a word, while a relation is assigned to a word pair; therefore, EE is performed only when the same words from forward and backward networks are composed.

Dynamics of the proposed TF-MTRNN architecture (Figure 3) are given below:

$$\begin{aligned} s_{R_{i,j\in i:N}} &= g(W_{hR}h_{i,j}); \qquad s_{E_{i,j=i}} = g(W_{hE}h_{i,j}); \qquad h_{i,j} = h_{f_i} + h_{b_j} \\ h_{f_i} &= f(U_f w_i + W_f h_{f_{i-1}}); \qquad h_{b_j} = f(U_b w_j + W_b h_{b_{j+1}}) \end{aligned} \tag{1}$$

where i and j are the time-steps of forward and backward networks, respectively. ith word in the sequence is combined with every jth word, where $j = i, ..., N$ (i.e. combined with itself and the following words in the sequence). N is the total number of words in the sequence. For a given sequence, $s_{R_{i,j}}$ and $s_{E_{i,j}}$ represent the output scores of relation and entity recognition for ith and jth word from forward and backward networks, respectively. Observe that EE is performed on the combined hidden representation $h_{i,j}$, computed from the composition of representations of the same word from forward and backward networks, therefore $i = j$ and resembling the diagonal entries for entities in Table 1. h_{f_i} and h_{b_j} are hidden representations of forward and backward networks, respectively. W_{hR} and W_{hE} are weights between hidden layers ($h_{i,j}$) and the output units of relation and entity, respectively. f and g are activation and loss functions. Applying $argmax$ to $s_{R_{i,j\in i:N}}$ and $s_{E_{i,j=i}}$ gives corresponding table entries for relations and entities, in Table 1 and Figure 4.

2.3 Context-aware TF-MTRNN model

In Figure 3, we observe that when hidden representations for the words *Association* and *Va* are combined, the middle context i.e. all words in the sequence occurring between the word pair in composition are missed. Therefore, we introduce a third direction in the network (Figure 5) with missing context (i.e. *in Alexandria ,*) to accumulate the full context in combined hidden vectors ($h_{i,j}$).

Dynamics of the context-aware TF-MTRNN is similar to Eq. 1, except h_{b_j}, in Figure 5:

$$\begin{aligned} h_{b_j} &= f(U_b w_j + W_b h_{b_{j+1}} + U_f h_{m_{t=T}}) \\ h_{b_{j+1}} &= f(U_b w_{j+1} + W_b h_{b_{j+2}}); \qquad h_{m_t} = f(U_f w_t + W_f h_{m_{t-1}}) \end{aligned} \tag{2}$$

where h_{b_j} is the hidden representation in backward network obtained from the combination of jth word and contexts from backward network and from missing direction, $t = (i + 1, ..., T = j - 1)$, where i and j are the time-steps for forward and backward networks, respectively. $h_{m_{t=i}}$ is initialized with zeros similar to forward and backward networks. There is no missing context when $i = 0$ and $j = 0$ i.e. w_t is NULL and therefore, we introduce an artificial word *PADDING* and use its embedding to initialise w_t.

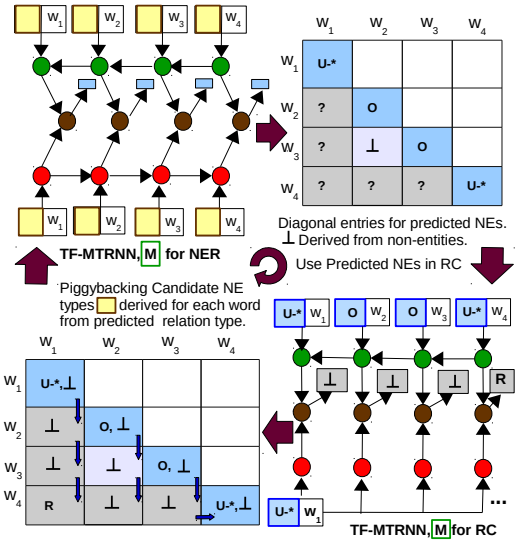


Figure 6: End-to-End Relation Extraction by jointly modeling entity and relation in a unified Multi-task RNN framework, M (TF-MTRNN) and filling an Entity-Relation table. Entity-relation interdependencies modeled by parameter sharing and piggybacking (Section 2.4 and Figure 7). NE: Named Entity; U-* and O: NE in BILOU format; ?:Relation to determine.

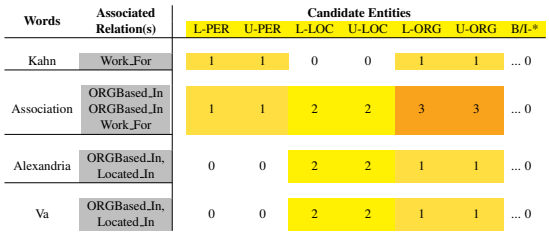


Figure 7: Piggybacking approach to model label dependencies from relations to entities. We do not list all words due to space limitation. * indicates any entity type. Highlight for counts indicate candidate entity importance for corresponding words.

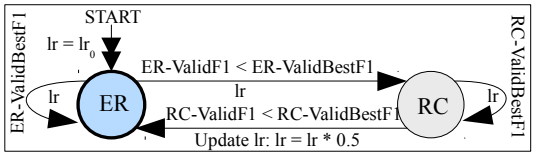


Figure 8: State Machine driven Multi-task Learning. ER:Entity Recognition; RC:Relation Classification; lr:learning rate; $ER - ValidBestF1$:Best entity recognition F_1 score on validation set.

2.4 Piggybacking for Entity-Relation Label Dependencies

Having named-entity labels is very informative for finding the relation type between them, and vice versa having the relation type between words eases problem of named-entity tagging. We model these label interdependencies during the end-to-end relation extraction in Figure 6, where the input vector at time step, t is given by -

$$input_t = \{C_{RE}, E_{ER}, W_{emb}\} \tag{3}$$

where C_{RE} is the count vector to model relation to entity dependencies, E_{ER} is the one-hot vector for predicted entities to model entity to relation dependencies and W_{emb} is the word embedding vector. Therefore, the input vector at each time step, t is the concatenation of these three vectors.

To model *entity to relation* dependency, the TF-MTRNN model, M for NER (Figure 6) first computes entity types, which are represented by diagonal entries of entity-relation table. Each predicted entity type E_{ER} (filled blue-color boxes) is concatenated with its corresponding word embedding vector W_{emb} and then input to the same model, M for relation classification.

To model *relation to entity* dependency, we derive a list of possible candidate entity tags for each word participating in a relation(s), except for $\perp$ relation type. Each word associated with a relation type(s) is determined from relation classification (RC) step (Figure 6). Figure 7 illustrates the entity type count vector for each word of the given sentence (Figure 1). For example, the word *Alexandria* participates in the relation types: *ORGBased_In* and *Located_In*. Possible entity types are {*U-ORG*, *L-ORG*, *U-LOC*, *L-LOC*} for *ORGBased_In*, while {*U-LOC*, *L-LOC*} for *Located_In*. We then compute a count vector C_{RE} from these possible entity types. Therefore, *U-LOC* and *L-LOC* each with occurrence 2, while *U-ORG* and *L-ORG* each with occurrence 1 (Figure 7). The candidate entity types as count vector (filled-yellow color box) for each word is piggybacked to model, M for entity learning by concatenating it with corresponding word embedding vector W_{emb}. This simple approach of piggybacking the count vectors of candidate entities enables learning label dependencies from *relation to entity* in order to improve entity extraction. In addition, multi-tasking by sharing parameters and adapting shared embeddings within a unified network enables learning label interdependencies.

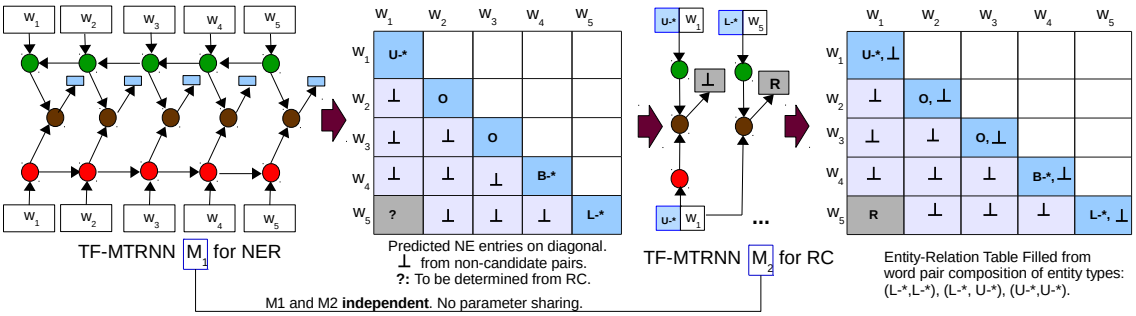


Figure 9: Pipeline Approach in End-to-End Relation Extraction.

2.5 Ranking Bi-directional Recurrent Neural Network (R-biRNN)

Ranking loss has been used in neural architectures (dos Santos et al., 2015) and (Vu et al., 2016b) to handle *artificial* classes. In our experiments, for a given sentence x with class label y^+, the competitive class c^- is chosen the one with the highest score among all competitive classes during SGD step. The basic principle is to learn to maximize the distance between the true label y^+ and the best competitive label c^- for a given data point x. We use the ranking loss to handle the two artificial classes i.e. 'O' and $\perp$ in entity and relation types, respectively. The ranking objective function is defined as-

$$L = \log(1 + \exp(\gamma(m^+ - s_\theta(x)_{y^+}))) + \log(1 + \exp(\gamma(m^- + s_\theta(x)_{c^-}))); \\ c^- = \arg \max_{c \epsilon C; c \neq y^+} s_\theta(x)_c \tag{4}$$

where $s_\theta(x)_{y^+}$ and $s_\theta(x)_{c^-}$ are the scores for positive y^+ and the most competitive c^- classes. γ controls the penalization of the prediction errors while hyperparameters m^+ and m^- are the margins for the true and competitive classes. We set $\gamma = 2, m^+ = 2.5, m^- = 0.5$, following (Vu et al., 2016b).

The unified architecture (Figure 3) can be viewed as being comprised of two individual models, each for NER and RE (Figure 6). We illustrate that the R-biRNN (Figure 12 in Appendix A) is integrated in TF-MTRNN (Figure 3) and therefore, the unified model leverages R-biRNN (Vu et al., 2016b) effectiveness for entity extraction, where the full context information is availed from the forward and backward network at each input word vector along with the ranking loss at each output node. Figure 12 corresponds to the diagonal entries for named entities in Table 1 and enables entity-entity label dependencies (Miwa and Sasaki, 2014) via sequential learning.

3 Model Training

3.1 End-to-End Relation Extraction

In CoNLL04, more than 99% of the whole word pairs lie in the no_relation class. Therefore, named-entity candidates are required to choose the candidate word pairs in relation learning. In Figure 6 and Figure 9, we demonstrate the joint and pipeline approach for end-to-end relation extraction.

In Figure 6, the candidate relation pairs are chosen by filtering out the non-entities pairs. Therefore, in entity-relation table, we insert 'no_relation' label for the non-entities pairs and RC is not performed. Note that a word pair is chosen for RC in which at least one word is an entity. It allows the model M to correct itself at NER by piggybacking candidate named entities (Figure 7). In addition, it reduces a significant number of non-relation word pairs and does not create a bias towards the no_relation class. However, in Figure 9, the two independent models, M_1 and M_2 are trained for NER and RC, respectively. In pipeline approach, the only relation candidates are word pairs with (U-*, U-*),(L-*, L-*) or (U-*, L-*) entity types. Therefore, only w_1 and w_5 from word sequence are composed in M_2 for RC subtask.

	Features	CoNLL04 Dataset					
		NER			RE		
		P	R	F_1	P	R	F_1
separate	basic	.865	.902	.883	.360	.403	.376
	+POS	.877	.906	.892	.440	.376	.395
	+CF	.906	.914	.910	.454	.390	.410
	+CTX				.499	.434	.453
pipeline	basic				.641	.545	.589
	+POS				.663	.555	.604
	+CF				.661	.585	.621
	+CTX				.736	.616	.671
joint	basic	.885	.889	.888	.646	.531	.583
	+POS	.904	.908	.906	.673	.531	.594
	+CF	.913	.914	.914	.691	.562	.620
	+CTX				.745	.595	.661
	+p'backing	.925	.921	.924	.785	.630	.699
	+ensemble	.936	.935	.936	.832	.635	.721

Figure 10: CoNLL04 dataset: Performance on test set for NER and RE; RE in pipeline always used predicted NEs. POS: part-of-speech; CF: capital features; CTX: context awareness (Figure 5); p'backing: piggybacking predicted and candidate entities in RE and NER, respectively; ensemble: majority vote.

Figure 11: T-SNE view of the semantic entity-relation space for the combined hidden representations of each word pair composition. Relations: (0:*LIVEIN*, 1:*ORGBASEDIN*, 2:*LOCATEDIN*, 3:*WORKFOR*, 4:*KILL*, 5:*NORELATION*). Entity-pair and relation denoted by E1-RELATION-E2 and/or count in [0-5]. 5: misclassified entity-pairs.

3.2 Word Representation and Features

Each word is represented by concatenation of pre-trained 50-dimensional word embeddings[3] (Turian et al., 2010) with N-gram, part-of-speech (POS), capital feature (CF: all-capitalized; initial-capitalized) and piggybacked entity vectors (Section 2.4). The word embeddings are shared across entity and relation extraction tasks and are adapted by updating them during training. We use 7-gram $(w_{t-3}w_{t-2}w_{t-1}w_tw_{t+1}w_{t+2}w_{t+3})$ obtained by concatenating corresponding word embeddings.

3.3 State Machine driven Multi-tasking

Multi-task training is performed via switching across multiple tasks in a block of training steps. However, we perform switches between ER and RC subtasks based on the performance of each task on the common validation set and update learning rate only when task is switched from RC to ER (Figure 8). ER is the task to start for multi-tasking and ER/RC is switched in the following training step, when their $ValidF1$ score is not better than $BestValidF1$ score of previous steps on the validation set.

4 Evaluation and Analysis

4.1 Dataset and Experimental Setup

We use CoNLL04[4] corpus of Roth and Yih (2004). Entity and relation types are shown in Figure 2. There are 1441 sentences with at least one relation. We randomly split these into training (1153 sentences) and test (288 sentences), similar to Miwa and Sasaki (2014). We release this train-test split at `https://github.com/pgcool/TF-MTRNN/tree/master/data/CoNLL04`. We introduce the pseudo-label $\perp$ "no_relation" for word pairs with no relation.

To tune hyperparameters, we split (80-20%) the training set (1153 sentences) into $train$ and validation (dev) sets. All final models are trained on $train$+dev. Our evaluation measure is F_1 on entities and relations. An entity is marked correct if NE boundaries and entity type[5] are correct. A relation for a word pair is marked correct if the NE boundaries and relation type are correct. However, in separate approach, a relation for a word pair is marked correct if the relation type is correct.

[3] with a special token PADDING. Also, used when there is no missing context.

[4] conll04.corp at `cogcomp.cs.illinois.edu/page/resource_view/43`

[5] For multi-word entity mention, an entity is marked correct if atleast one token is tagged correctly.

	Roth&Yih			Kate&Mooney			Miwa&Sasaki			TF-MTRNN		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Person	.891	.895	.890	.921	.942	.932	.931	.948	.939	.932	.988	.959
Location	.897	.887	.891	.908	.942	.924	.922	.939	.930	.974	.956	.965
Organization	.895	.720	.792	.905	.887	.895	.903	.896	.899	.873	.939	.905
(Average)	.894	.834	.858	.911	.924	. 917	.919	.927	.923	.926	.961	.943
Live_In	.591	.490	.530	.664	.601	.629	.819	.532	.644	.727	.640	.681
OrgBased_In	.798	.416	.543	.662	.641	.647	.768	.572	.654	.831	.562	.671
Located_In	.539	.557	.513	.539	.557	.513	.821	.549	.654	.867	.553	.675
Work_For	.720	.423	.531	.720	.423	.531	.886	.642	.743	.945	.671	.785
Kill	.775	.815	.790	.775	.815	.790	.933	.797	.858	.857	.894	.875
(Average)	.685	.540	.581	.672	.607	.622	.845	.618	.710	.825	.664	.737

Table 2: State-of-the-art comparison for EE and RE on CoNLL04 dataset.

4.2 Results

Figure 10 shows results for NER[6] and RE. All models use n-grams for $n = 7$ (Section 3.2). Embedding dimensionality is 50. The notation "+" (e.g., +POS) at the beginning of a line indicates that the model of this line is the same as the model on the previous line except that one more model element (e.g., POS) is added. The separate NER model performs NER only. The separate RE model performs RE only, without access to NER results. The pipeline RE model takes the results of the separate NER model and then performs RE. The joint model is trained jointly on NER and RE. For compactness, we show the results of *two different models* (an NER model and an RE model) in the separate part of the table; in contrast, results for a *single model* – evaluated on both NER and RE – are shown in the joint part.

We make the following observations based on Figure 10. (i) All of our proposed model elements (POS, CF, CTX, piggybacking, ensemble) improve performance, in particular CTX and piggybacking provide large improvements. (ii) Not surprisingly, the pipeline RE model that has access to NER classifications performs better than the separate RE model. (iii) The joint model performs better than separate and pipeline models, demonstrating that joint training and decoding is advantageous for joint NER and RE. (iv) Majority voting[7] (ensemble) results in a particularly large jump in performance and in the overall best performing system; F_1 is .936 for NER and .721 for RE, respectively.

4.3 Comparison with Other Systems

Our end-to-end relation extraction system outperform the state-of-the-art results. We compare the entity and relation extraction performance of our model with other systems (Roth and Yih, 2007; Kate and Mooney, 2010; Miwa and Sasaki, 2014). (Roth and Yih, 2007) performed 5-fold cross validation on the complete corpus (1441 sentences), while (Miwa and Sasaki, 2014) performed 5-cross validation on the data set, obtained after splitting the corpus. We report our results on the test set from random split (80-20%) of the corpus, similar to (Miwa and Sasaki, 2014). Since, the standard splits were not available, we cannot directly compare the results, but our proposed model shows an improvement of 2.0% and 2.7% in F_1 scores for entity and relation extraction tasks, respectively (Table 2).

[6]Our NER model reports 86.80% F1 score, comparable to 86.67% from (Lample et al., 2016) on CoNLL03 shared task using the standard NER evaluation script with strict multi-word entity evaluation, and adapted for BILOU encoding.

[7]Randomly pick one of the most frequent classes, in case of a tie

4.4 Word pair Compositions (T-SNE)

Using t-SNE (der Maaten and Hinton, 2008), we visualize the hidden representations obtained on the composition of hidden vectors of every two words (word pair) in the sentence via TF-MTRNN model. In Figure 11, we show all data points i.e. word pair compositions, leading to natural relations (except $\perp$ denoted by 5). We observe that the entity mention pairs with common relation types form clusters corresponding to each relation in the semantic entity-relation space. We observe that the relation clusters with common entity type lie close to each other, for example, *KILL* has (*PER*, *PER*) entity pairs, which is close to relation cluster *LIVEIN* and *WORKFOR*, in which one of the entities i.e. *PER* is common. While, *KILL* relation cluster is at a distance from *LOCATEDIN* cluster, since they have no common entity.

4.5 Hyperparamter Settings

We use stochastic gradient descent with L2 regularization with a weight of .0001. The initial learning rate for entity and relation extraction is .05 with hidden layer size 200. The learning rate update and task switching is driven by the state machine (Figure 8). Models are trained for 40 iterations performing stochastic gradient descent. We initialize the recurrent weight matrix to be identity and biases to be zero. We use Capped Rectified Linear units (CappedReLu) and ranking loss with default parameters (section 2.5). The entity vectors C_{RE} and E_{ER} are initialized with zero when NER is performed for the first time in entity and relation extraction loop (Figure 6). The models are implemented in Theano (Bergstra et al., 2010; Bastien et al., 2012).

5 Related Work

Recurrent and convolutional neural networks (Zeng et al., 2014; Nguyen and Grishman, 2015; Zhang and Wang, 2015; Vu et al., 2016a) have delivered competitive performance for sentence-level relation classification. Socher et al. (2012) and Zhang and Wang (2015) proposed recurrent/recursive type neural networks to construct sentence representations based on dependency parse trees. However, these sentence-level state-of-the-art methods do not model the interdependencies of entity and relation, do not handle multiple relation instances in a sentence and therefore, can not detect entity mention pairs for the sentence-level relations. Our approach is a joint entity and word-level relation extraction capable to model multiple relation instances, without knowing nominal pairs.

Existing systems (Roth and Yih, 2004; Kate and Mooney, 2010; Miwa and Sasaki, 2014) are complex feature-based models for joint entity and relation extraction. The most related work to our method is (Miwa and Sasaki, 2014); however they employ complex search heuristics (Goldberg and Elhadad, 2010; Stoyanov and Eisner, 2012) to fill the entity-relation table based on structured prediction method. They explicitly model the label dependencies and their joint approach is not based on neural networks. Multi-task learning (Caruana, 1998) via neural networks (Zhang and Yeung, 2012; Seltzer and Droppo, 2013; Dong et al., 2015; Li and J, 2014; Collobert and Weston, 2008) have been used to model relationships among the correlated tasks. Therefore, we present a unified neural network based multi-task framework to model the entity-relation table for end-to-end relation extraction.

6 Conclusion

We proposed TF-MTRNN, a novel architecture that jointly models entity and relation extraction, and showed how an entity-relation table is mapped to a neural network framework that learns label interdependencies. We introduced word-level relation classification through composition of words; this is advantageous in modeling multiple relation instances without knowing the corresponding entity mentions in a sentence. We also introduced context-awareness in RNN network to incorporate missing information, and investigated piggybacking approach to model entity-relation label interdependencies.

Experimental results show that TF-MTRNN outperforms state-of-the-art method for both entity and relation extraction tasks.

Acknowledgements
This research was supported by Siemens AG CT MIC - Machine Intelligence, Munich Germany.

References

Frèdèric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. 2012. Theano: new features and speed improvements. *Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop*.

James Bergstra, Olivier Breuleux, Frèdèric Bastien, Pascal Lamblin, Guillaume Desjardins Razvan Pascanu, Joseph Turian, David Warde-Farley, , and Yoshua Bengio. 2010. Theano: a cpu and gpu math expression compiler. *In Proceedings of the Python for Scientific Computing Conference (SciPy).*

Rich Caruana. 1998. Multitask learning. *Springer*.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing:deep neural networks with multitask learning. *In Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland.*

L Van der Maaten and G Hinton. 2008. Visualizing data using t-sne. *Proceedinggs of JMLR*.

Daxiang Dong, Wei He Hua Wu, Dianhai Yu, and Haifeng Wang. 2015. Multi-task learning for multiple language translation. *In Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 1723–1732.

Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. *In Proceedings of the Association for Computational Linguistics*.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2).

Yoav Goldberg and Michael Elhadad. 2010. An efficient algorithm for easy-first non-directional dependency parsing. *In Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 742–750.

M. Jordan. 1986. Serial order: A parallel distributed processing approach. *Published in Tech. Rep. No. 8604. San Diego: University of California, Institute for Cognitive Science*.

Rohit J. Kate and Raymond Mooney. 2010. Joint entity and relation extraction using card-pyramid parsing. *In Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 203–212.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition.

Qi Li and Heng J. 2014. Incremental joint extraction of entity mentions and relations. *In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 402–412.

Makoto Miwa and Yutaka Sasaki. 2014. Modeling joint entity and relation extraction with table representation. *In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1858–1869.

Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. *In Proceedings of the NAACL Workshop on Vector Space Modeling for NLP*.

Dan Roth and Wen-Tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. *In Hwee Tou Ng and Ellen Riloff, editors, HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*, pages 1–8.

Dan Roth and Wen-Tau Yih. 2007. Global inference for entity and relation identification via a linear programming formulation. *In Hwee Tou Ng and Ellen Riloff, editors, HLT-NAACL 2004 Workshop: Eighth Conference on Computational Natural Language Learning (CoNLL-2004)*.

Michael L Seltzer and Jasha Droppo. 2013. Multi-task learning in deep neural networks for improved phoneme recognition. *In Proceedings of the IEEE International Confonference on Acoustics, Speech, and Signal Processing (ICASSP)*.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. In proceedings of emnlp/conll. *Association for Computational Linguistics*.

Veselin Stoyanov and Jason Eisner. 2012. Easy-first coreference resolution. *In Proceedings of COLING 2012*, page 25192534.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. *In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*.

Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schütze. 2016a. Combining recurrent and convolution neural networks for relation classification. *In Proceedings of the NAACL*.

Ngoc Thang Vu, Pankaj Gupta, Heike Adel, and Hinrich Schütze. 2016b. Bi-directional recurrent neural network with ranking loss for spoken language understanding. *IEEE/ACM Trans. on Audio, Speech, and Language Processing*.

Paul J Werbos. 1990. Backpropagation through time:what it does and how to do it. *In Proceedings of the IEEE*.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. *In Proceedings of COLING*.

Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network. *In ArXiv*.

Y. Zhang and D.-Y. Yeung. 2012. A convex formulation for learning task relationships in multi-task learning. *arXiv preprint arXiv:1203.3536*.

Appendix A. R-biRNN discussed in section 2.5.

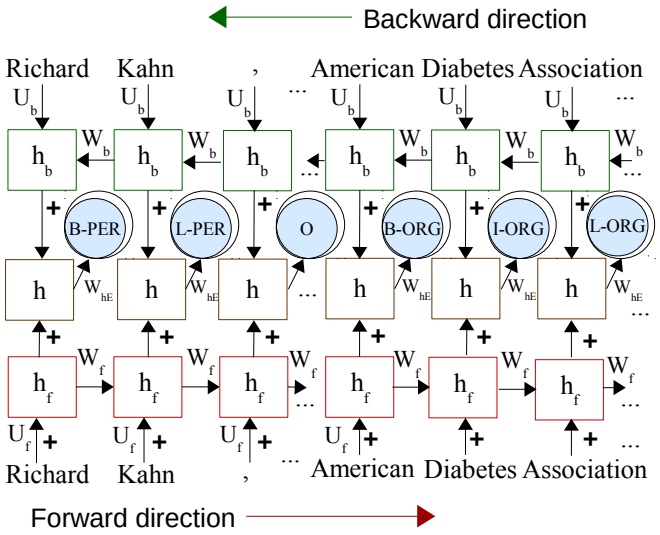


Figure 12: R-biRNN. Disintegrating TF-MTRNN (Figure 3) to illustrate that it is comprised of R-biRNN for entity learning. (...) indicates remaining words in the sentence (Figure 1).

Chapter 3

Joint Bootstrapping Machines for High Confidence Relation Extraction

Joint Bootstrapping Machines for High Confidence Relation Extraction

Pankaj Gupta[1,2], Benjamin Roth[2], Hinrich Schütze[2]
[1]Corporate Technology, Machine-Intelligence (MIC-DE), Siemens AG Munich, Germany
[2]CIS, University of Munich (LMU) Munich, Germany
pankaj.gupta@siemens.com | pankaj.gupta@campus.lmu.de
{beroth, inquiries}@cis.lmu.de

Abstract

Semi-supervised bootstrapping techniques for relationship extraction from text iteratively expand a set of initial seed instances. Due to the lack of labeled data, a key challenge in bootstrapping is semantic drift: if a false positive instance is added during an iteration, then all following iterations are contaminated. We introduce BREX, a new bootstrapping method that protects against such contamination by highly effective confidence assessment. This is achieved by using entity and template seeds jointly (as opposed to just one as in previous work), by expanding entities and templates in parallel and in a mutually constraining fashion in each iteration and by introducing higher-quality similarity measures for templates. Experimental results show that BREX achieves an F_1 that is 0.13 (0.87 vs. 0.74) better than the state of the art for four relationships.

1 Introduction

Traditional semi-supervised bootstrapping relation extractors (REs) such as BREDS (Batista et al., 2015), SnowBall (Agichtein and Gravano, 2000) and DIPRE (Brin, 1998) require an initial set of seed *entity pairs* for the target binary relation. They find occurrences of positive seed entity pairs in the corpus, which are converted into extraction patterns, i.e., *extractors*, where we define an extractor as a cluster of instances generated from the corpus. The initial seed entity pair set is expanded with the relationship entity pairs newly extracted by the extractors from the text iteratively. The augmented set is then used to extract new relationships until a stopping criterion is met.

Due to lack of sufficient labeled data, rule-based systems dominate commercial use (Chiticariu et al., 2013). Rules are typically defined by creating patterns around the entities (entity extraction) or entity pairs (relation extraction). Recently, supervised machine learning, especially deep learning techniques (Gupta et al., 2015; Nguyen and Grishman, 2015; Vu et al., 2016a,b; Gupta et al., 2016), have shown promising results in entity and relation extraction; however, they need sufficient hand-labeled data to train models, which can be costly and time consuming for web-scale extractions. Bootstrapping machine-learned rules can make extractions easier on large corpora. Thus, open information extraction systems (Carlson et al., 2010; Fader et al., 2011; Mausam et al., 2012; Mesquita et al., 2013; Angeli et al., 2015) have recently been popular for domain specific or independent pattern learning.

Hearst (1992) used hand written rules to generate more rules to extract hypernym-hyponym pairs, without distributional similarity. For entity extraction, Riloff (1996) used seed entities to generate extractors with heuristic rules and scored them by counting positive extractions. Prior work (Lin et al., 2003; Gupta et al., 2014) investigated different extractor scoring measures. Gupta and Manning (2014) improved scores by introducing expected number of negative entities.

Brin (1998) developed the bootstrapping relation extraction system DIPRE that generates extractors by clustering contexts based on string matching. SnowBall (Agichtein and Gravano, 2000) is inspired by DIPRE but computes a TF-IDF representation of each context. BREDS (Batista et al., 2015) uses word embeddings (Mikolov et al., 2013) to bootstrap relationships.

Related work investigated adapting extractor scoring measures in bootstrapping entity extraction with either entities or *templates* (Table 1) as seeds (Table 2). The state-of-the-art relation extractors bootstrap with only seed entity pairs and suffer due to a surplus of unknown extractions and the lack of labeled data, leading to low confidence extractors. This in turn leads to to low confidence in the system output. Prior RE sys-

BREE	Bootstrapping Relation Extractor with *Entity pair*
BRET	Bootstrapping Relation Extractor with *Template*
BREJ	Bootstrapping Relation Extractor in Joint learning
type	a named entity type, e.g., `person`
typed entity	a typed entity, e.g., <"Obama",`person`>
entity pair	a pair of two typed entities
template	a triple of vectors ($\vec{v}_{-1}$, $\vec{v}_0$, $\vec{v}_1$) and an entity pair
instance	entity pair and template (types must be the same)
γ	instance set extracted from corpus
i	a member of γ, i.e., an instance
$x(i)$	the entity pair of instance i
$\mathfrak{x}(i)$	the template of instance i
G_p	a set of positive seed entity pairs
G_n	a set of negative seed entity pairs
$\mathfrak{G}_p$	a set of positive seed templates
$\mathfrak{G}_n$	a set of negative seed templates
$\mathcal{G}$	$< G_p, G_n, \mathfrak{G}_p, \mathfrak{G}_n >$
k_{it}	number of iterations
λ_{cat}	cluster of instances (*extractor*)
cat	category of *extractor* λ
λ_{NNHC}	Non-Noisy-High-Confidence extractor (True Positive)
λ_{NNLC}	Non-Noisy-Low-Confidence extractor (True Negative)
λ_{NHC}	Noisy-High-Confidence extractor (False Positive)
λ_{NLC}	Noisy-Low-Confidence extractor (False Negative)

Table 1: Notation and definition of key terms

tems do not focus on improving the extractors' scores. In addition, SnowBall and BREDS used a weighting scheme to incorporate the importance of contexts around entities and compute a similarity score that introduces additional parameters and does not generalize well.

Contributions. **(1)** We propose a *Joint Bootstrapping Machine*[1] (JBM), an alternative to the entity-pair-centered bootstrapping for relation extraction that can take advantage of both entity-pair and template-centered methods to jointly learn extractors consisting of instances due to the occurrences of both entity pair and template seeds. It scales up the number of positive extractions for *non-noisy* extractors and boosts their confidence scores. We focus on improving the scores for *non-noisy-low-confidence* extractors, resulting in higher *recall*. The relation extractors bootstrapped with entity pair, template and joint seeds are named as *BREE*, *BRET* and *BREJ* (Table 1), respectively.

(2) Prior work on embedding-based context comparison has assumed that relations have *consistent syntactic expression* and has mainly addressed synonymy by using embeddings (e.g.,"acquired" – "bought"). In reality, there is *large variation in the syntax* of how relations are expressed, e.g., "MSFT to acquire NOK for $8B" vs. "MSFT earnings hurt by NOK acquisition". We introduce cross-context similarities that compare all parts of the context (e.g., "to acquire" and "acquisition") and show that these perform better (in terms of recall) than measures assuming consistent syntactic expression of relations.

(3) Experimental results demonstrate a 13% gain in $F1$ score on average for four relationships and suggest eliminating four parameters, compared to the state-of-the-art method.

The *motivation* and *benefits* of the proposed JBM for relation extraction is discussed in depth in section 2.3. The method is applicable for both entity and relation extraction tasks. However, in *context of relation extraction*, we call it *BREJ*.

2 Method

2.1 Notation and definitions

We first introduce the notation and terms (Table 1).

Given a relationship like "x acquires y", the task is to extract pairs of entities from a corpus for which the relationship is true. We assume that the arguments of the relationship are typed, e.g., x and y are organizations. We run a named entity tagger in preprocessing, so that the types of all candidate entities are given. The objects the bootstrapping algorithm generally handles are therefore *typed entities* (an entity associated with a type).

For a particular sentence in a corpus that states that the relationship (e.g., "acquires") holds between x and y, a *template* consists of three vectors that represent the context of x and y. $\vec{v}_{-1}$ represents the context before x, $\vec{v}_0$ the context between x and y and $\vec{v}_1$ the context after y. These vectors are simply sums of the embeddings of the corresponding words. A template is "typed", i.e., in addition to the three vectors it specifies the types of the two entities. An *instance* joins an entity pair and a template. The types of entity pair and template must be the same.

The first step of bootstrapping is to extract a set of instances from the input corpus. We refer to this set as γ. We will use i and j to refer to instances. $x(i)$ is the entity pair of instance i and $\mathfrak{x}(\mathrm{i})$ is the template of instance i.

A required input to our algorithm are sets of positive and negative seeds for either entity pairs (G_p and G_n) or templates ($\mathfrak{G}_p$ and $\mathfrak{G}_n$) or both. We define $\mathcal{G}$ to be a tuple of all four seed sets.

We run our bootstrapping algorithm for k_{it} iterations where k_{it} is a parameter.

[1] github.com/pgcool/Joint-Bootstrapping-Machines

A key notion is the similarity between two instances. We will experiment with different similarity measures. The baseline is (Batista et al., 2015)'s measure given in Figure 4, first line: the similarity of two instances is given as a weighted sum of the dot products of their before contexts ($\vec{v}_{-1}$), their between contexts ($\vec{v}_0$) and their after contexts ($\vec{v}_1$) where the weights w_p are parameters. We give this definition for instances, but it also applies to templates since only the context vectors of an instance are used, not the entities.

The similarity between an instance i and a cluster λ of instances is defined as the maximum similarity of i with any member of the cluster; see Figure 2, right, Eq. 5. Again, there is a straightforward extension to a cluster of templates: see Figure 2, right, Eq. 6.

The extractors Λ can be categorized as follows:

$$\Lambda_{NNHC} = \{\lambda \in \Lambda | \underbrace{\lambda \mapsto \Re}_{non-noisy} \wedge \text{cnf}(\lambda, \mathcal{G}) \geqslant \tau_{cnf}\} \quad (1)$$

$$\Lambda_{NNLC} = \{\lambda \in \Lambda | \lambda \mapsto \Re \wedge \text{cnf}(\lambda, \mathcal{G}) < \tau_{cnf}\} \quad (2)$$

$$\Lambda_{NHC} = \{\lambda \in \Lambda | \underbrace{\lambda \nmapsto \Re}_{noisy} \wedge \text{cnf}(\lambda, \mathcal{G}) \geqslant \tau_{cnf}\} \quad (3)$$

$$\Lambda_{NLC} = \{\lambda \in \Lambda | \lambda \nmapsto \Re \wedge \text{cnf}(\lambda, \mathcal{G}) < \tau_{cnf}\} \quad (4)$$

where $\Re$ is the relation to be bootstrapped. The λ_{cat} is a member of Λ_{cat}. For instance, a λ_{NNLC} is called as a *non-noisy-low-confidence* extractor if it represents the target relation (i.e., $\lambda \mapsto \Re$), however with the confidence below a certain threshold (τ_{cnf}). Extractors of types Λ_{NNHC} and Λ_{NLC} are desirable, those of types Λ_{NHC} and Λ_{NNLC} undesirable within bootstrapping.

2.2 The Bootstrapping Machines: BREX

To describe BREX (Figure 1) in its most general form, we use the term *item* to refer to an entity pair, a template or both.

The input to BREX (Figure 2, left, line 01) is a set γ of instances extracted from a corpus and $\mathcal{G}_{\text{seed}}$, a structure consisting of one set of positive and one set of negative seed items. $\mathcal{G}_{\text{yield}}$ (line 02) collects the items that BREX extracts in several iterations. In each of k_{it} iterations (line 03), BREX first initializes the cache $\mathcal{G}_{\text{cache}}$ (line 04); this cache collects the items that are extracted in this iteration. The design of the algorithm balances elements that ensure high recall with elements that ensure high precision.

High recall is achieved by starting with the seeds and making three "hops" that consecutively consider order-1, order-2 and order-3 neighbors

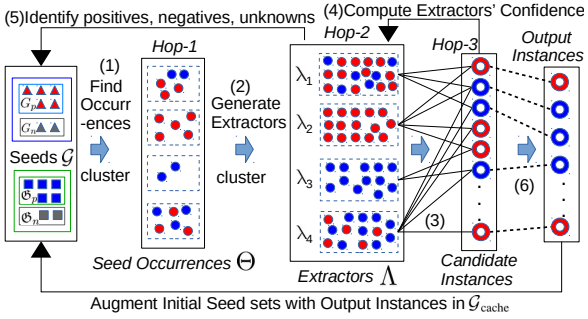


Figure 1: Joint Bootstrapping Machine. The red and blue filled circles/rings are the instances generated due to seed entity pairs and templates, respectively. Each dashed rectangular box represents a cluster of instances. Numbers indicate the flow. Follow the notations from Table 1 and Figure 2.

of the seeds. On line 05, we make the first hop: all instances that are similar to a seed are collected where "similarity" is defined differently for different BREX configurations (see below). The collected instances are then clustered, similar to work on bootstrapping by Agichtein and Gravano (2000) and Batista et al. (2015). On line 06, we make the second hop: all instances that are within τ_{sim} of a hop-1 instance are added; each such instance is only added to one cluster, the closest one; see definition of μ: Figure 2, Eq. 8. On line 07, we make the third hop: we include all instances that are within τ_{sim} of a hop-2 instance; see definition of ψ: Figure 2, Eq. 7. In summary, every instance that can be reached by three hops from a seed is being considered at this point. A cluster of hop-2 instances is named as *extractor*.

High precision is achieved by imposing, on line 08, a stringent check on each instance before its information is added to the cache. The core function of this check is given in Figure 2, Eq. 9. This definition is a soft version of the following hard max, which is easier to explain:

$$\text{cnf}(i, \Lambda, \mathcal{G}) \approx \max_{\{\lambda \in \Lambda | i \in \psi(\lambda)\}} \text{cnf}(i, \lambda, \mathcal{G})$$

We are looking for a cluster λ in Λ that licenses the extraction of i with high confidence. $\text{cnf}(i, \lambda, \mathcal{G})$ (Figure 2, Eq. 10), the *confidence* of a single cluster (i.e., extractor) λ for an instance, is defined as the product of the overall reliability of λ (which is independent of i) and the similarity of i to λ, the second factor in Eq. 10, i.e., $\text{sim}(i, \lambda)$. This factor $\text{sim}(i, \lambda)$ prevents an extraction by a cluster whose members are all distant from the instance – even if the cluster itself is highly reliable.

Algorithm: *BREX*

01 INPUT: γ, $\mathcal{G}_{\text{seed}}$
02 $\mathcal{G}_{\text{yield}} := \mathcal{G}_{\text{seed}}$
03 for k_{it} iterations:
04 $\quad\mathcal{G}_{\text{cache}} := \varnothing$
05 $\quad\Theta := \biguplus(\{i \in \gamma | \text{match}(i, \mathcal{G}_{\text{yield}})\}$
06 $\quad\Lambda := \{\mu(\theta, \Theta) | \theta \in \Theta\}$
07 $\quad$for each $i \in \bigcup_{\lambda \in \Lambda} \psi(\lambda)$:
08 $\quad\quad$if check$(i, \Lambda, \mathcal{G}_{\text{yield}})$:
09 $\quad\quad\quad$add$(i, \mathcal{G}_{\text{cache}})$
10 $\quad\mathcal{G}_{\text{yield}} \cup= \mathcal{G}_{\text{cache}}$
11 OUTPUT: $\mathcal{G}_{\text{yield}}$, Λ

$$\text{sim}(i, \lambda) = \max_{i' \in \lambda} \text{sim}(i, i') \quad (5)$$

$$\text{sim}(i, \mathfrak{G}) = \max_{\mathfrak{t} \in \mathfrak{G}} \text{sim}(i, \mathfrak{t}) \quad (6)$$

$$\psi(\lambda) = \{i \in \gamma | \text{sim}(i, \lambda) \geqslant \tau_{\text{sim}}\} \quad (7)$$

$$\mu(\theta, \Theta) = \{i \in \gamma | \text{sim}(i, \theta) = d \wedge d = \max_{\theta \in \Theta} \text{sim}(i, \theta) \geqslant \tau_{\text{sim}}\} \quad (8)$$

$$\text{cnf}(i, \Lambda, \mathcal{G}) = 1 - \prod_{\{\lambda \in \Lambda | i \in \psi(\lambda)\}} (1 - \text{cnf}(i, \lambda, \mathcal{G})) \quad (9)$$

$$\text{cnf}(i, \lambda, \mathcal{G}) = \text{cnf}(\lambda, \mathcal{G}) \text{sim}(i, \lambda) \quad (10)$$

$$\text{cnf}(\lambda, \mathcal{G}) = \frac{1}{1 + w_n \frac{N_+(\lambda, \mathcal{G}_n)}{N_+(\lambda, \mathcal{G}_p)} + w_u \frac{N_0(\lambda, \mathcal{G})}{N_+(\lambda, \mathcal{G}_p)}} \quad (11)$$

$$N_0(\lambda, \mathcal{G}) = |\{i \in \lambda | x(i) \notin (G_p \cup G_n)\}| \quad (12)$$

Figure 2: BREX algorithm (left) and definition of key concepts (right)

	BREE	**BRET**	**BREJ**
Seed Type	*Entity pairs*	*Templates*	*Joint (Entity pairs + Templates)*
(i) $N_+(\lambda, \mathcal{G}_l)$	$\lvert\{i \in \lambda \lvert x(i) \in G_l\}\rvert$	$\lvert\{i \in \lambda \lvert \text{sim}(i, \mathfrak{G}_l) \geqslant \tau_{\text{sim}}\}\rvert$	$\lvert\{i \in \lambda \lvert x(i) \in G_l\}\rvert + \lvert\{i \in \lambda \lvert \text{sim}(i, \mathfrak{G}_l) \geqslant \tau_{\text{sim}}\}\rvert$
(ii) (w_n, w_u)	$(1.0, 0.0)$	$(1.0, 0.0)$	$(1.0, 0.0)$
05 match$(i, \mathcal{G})$	$x(i) \in G_p$	$\text{sim}(i, \mathfrak{G}_p) \geqslant \tau_{\text{sim}}$	$x(i) \in G_p \vee \text{sim}(i, \mathfrak{G}_p) \geqslant \tau_{\text{sim}}$
08 check$(i, \Lambda, \mathcal{G})$	$\text{cnf}(i, \Lambda, \mathcal{G}) \geqslant \tau_{\text{cnf}}$	$\text{cnf}(i, \Lambda, \mathcal{G}) \geqslant \tau_{\text{cnf}}$	$\text{cnf}(i, \Lambda, \mathcal{G}) \geqslant \tau_{\text{cnf}} \wedge \text{sim}(i, \mathfrak{G}_p) \geqslant \tau_{\text{sim}}$
09 add$(i, \mathcal{G})$	$G_p \cup= \{x(i)\}$	$\mathfrak{G}_p \cup= \{\mathfrak{x}(i)\}$	$G_p \cup= \{x(i)\}$, $\mathfrak{G}_p \cup= \{\mathfrak{x}(i)\}$

Figure 3: BREX configurations

The first factor in Eq. 10, i.e., $\text{cnf}(\lambda, \mathcal{G})$, assesses the reliability of a cluster λ: we compute the ratio $\frac{N_+(\lambda, \mathcal{G}_n)}{N_+(\lambda, \mathcal{G}_p)}$, i.e., the ratio between the number of instances in λ that match a negative and positive gold seed, respectively; see Figure 3, line (i). If this ratio is close to zero, then likely false positive extractions are few compared to likely true positive extractions. For the simple version of the algorithm (for which we set $w_n = 1$, $w_u = 0$), this results in $\text{cnf}(\lambda, \mathcal{G})$ being close to 1 and the reliability measure it not discounted. On the other hand, if $\frac{N_+(\lambda, \mathcal{G}_n)}{N_+(\lambda, \mathcal{G}_p)}$ is larger, meaning that the relative number of likely false positive extractions is high, then $\text{cnf}(\lambda, \mathcal{G})$ shrinks towards 0, resulting in progressive discounting of $\text{cnf}(\lambda, \mathcal{G})$ and leading to *non-noisy-low-confidence* extractor, particularly for a reliable λ. Due to lack of labeled data, the scoring mechanism cannot distinguish between noisy and non-noisy extractors. Therefore, an extractor is judged by its ability to extract more positive and less negative extractions. Note that we carefully designed this precision component to give good assessments while at the same time making maximum use of the available seeds. The reliability statistics are computed on λ, i.e., on hop-2 instances (not on hop-3 instances). The ratio $\frac{N_+(\lambda, \mathcal{G}_n)}{N_+(\lambda, \mathcal{G}_p)}$ is computed on instances that directly match a gold seed – this is the most reliable information we have available.

After all instances have been checked (line 08) and (if they passed muster) added to the cache (line 09), the inner loop ends and the cache is merged into the yield (line 10). Then a new loop (lines 03–10) of hop-1, hop-2 and hop-3 extensions and cluster reliability tests starts.

Thus, the algorithm consists of k_{it} iterations. There is a tradeoff here between τ_{sim} and k_{it}. We will give two extreme examples, assuming that we want to extract a fixed number of m instances where m is given. We can achieve this goal either by setting k_{it}=1 and choosing a small τ_{sim}, which will result in very large hops. Or we can achieve this goal by setting τ_{sim} to a large value and running the algorithm for a larger number of k_{it}. The flexibility that the two hyperparameters k_{it} and τ_{sim} afford is important for good performance.

$$\text{sim}_{\text{match}}(i,j) = \sum_{p\in\{-1,0,1\}} w_p \vec{v}_p(i)\vec{v}_p(j) \quad ; \quad \text{sim}^{asym}_{cc}(i,j) = \max_{p\in\{-1,0,1\}} \vec{v}_p(i)\vec{v}_0(j) \tag{13}$$

$$\text{sim}^{sym1}_{cc}(i,j) = \max\left(\max_{p\in\{-1,0,1\}} \vec{v}_p(i)\vec{v}_0(j), \max_{p\in\{-1,0,1\}} \vec{v}_p(j)\vec{v}_0(i)\right) \tag{14}$$

$$\text{sim}^{sym2}_{cc}(i,j) = \max\Big(\big(\vec{v}_{-1}(i) + \vec{v}_1(i)\big)\vec{v}_0(j), \big(\vec{v}_{-1}(j) + \vec{v}_1(j)\big)\vec{v}_0(i), \vec{v}_0(i)\vec{v}_0(j)\Big) \tag{15}$$

Figure 4: Similarity measures. These definitions for instances equally apply to templates since the definitions only depend on the "template part" of an instance, i.e., its vectors. (value is 0 if types are different)

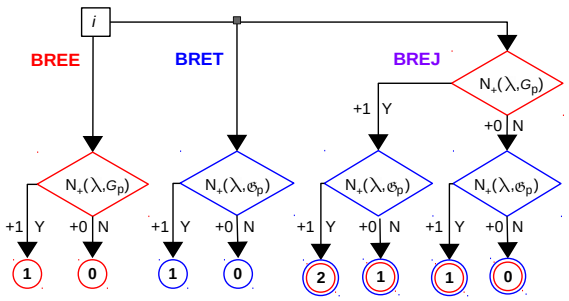


Figure 5: Illustration of Scaling-up Positive Instances. i: an instance in extractor, λ. Y: YES and N: NO

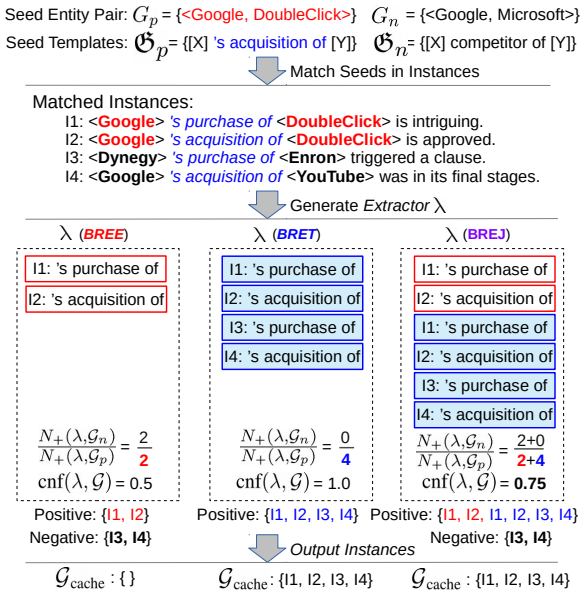


Figure 6: An illustration of scaling positive extractions and computing confidence for a non-noisy extractor generated for *acquired* relation. The dashed rectangular box represents an extractor λ, where λ (BREJ) is *hybrid* with 6 instances. Text segments matched with seed template are shown in italics. Unknowns (bold in black) are considered as negatives. $\mathcal{G}_{cache}$ is a set of output instances where $\tau_{cnf} = 0.70$.

2.3 BREE, BRET and BREJ

The main contribution of this paper is that we propose, as an alternative to entity-pair-centered BREE (Batista et al., 2015), template-centered BRET as well as BREJ (Figure 1), an instantiation of BREX that can take advantage of both entity pairs and templates. The differences and advantages of BREJ over BREE and BRET are:

(1) Disjunctive Matching of Instances: The first difference is realized in how the three algorithms match instances with seeds (line 05 in Figure 3). BREE checks whether the entity pair of an instance is one of the entity pair seeds, BRET checks whether the template of an instance is one of the template seeds and BREJ checks whether the disjunction of the two is true. The disjunction facilitates a higher hit rate in matching instances with seeds. The introduction of a few handcrafted templates along with seed entity pairs allows BREJ to leverage discriminative patterns and learn similar ones via distributional semantics. In Figure 1, the joint approach results in *hybrid* extractors Λ that contain instances due to seed occurrences Θ of both entity pairs and templates.

(2) Hybrid Augmentation of Seeds: On line 09 in Figure 3, we see that the bootstrapping step is defined in a straightforward fashion: the entity pair of an instance is added for BREE, the template for BRET and both for BREJ. Figure 1 demonstrates the hybrid augmentation of seeds via *red* and *blue* rings of *output instances*.

(3) Scaling Up Positives in Extractors: As discussed in section 2.2, a good measure of the quality of an extractor is crucial and N_+, the number of instances in an extractor λ that match a seed, is an important component of that. For BREE and BRET, the definition follows directly from the fact that these are entity-pair and template-centered instantiations of BREX, respectively. However, the disjunctive matching of instances for an extractor with entity pair and template seeds in BREJ (Figure 3 line "(i)") boosts the likelihood of finding positive instances. In Figure 5, we demonstrate computing the count of positive instances

Relationship	*Seed Entity Pairs*	*Seed Templates*
acquired	{Adidas;Reebok},{Google;DoubleClick}, {Widnes;Warrington},{Hewlett-Packard;Compaq}	{[X] acquire [Y]},{[X] acquisition [Y]},{[X] buy [Y]}, {[X] takeover [Y]},{[X] merger with [Y]}
founder-of	{CNN;Ted Turner},{Facebook;Mark Zuckerberg}, {Microsoft;Paul Allen},{Amazon;Jeff Bezos},	{[X] founded by [Y]},{[X] co-founder [Y]},{[X] started by [Y]}, {[X] founder of [Y]},{[X] owner of [Y]}
headquartered	{Nokia;Espoo},{Pfizer;New York}, {United Nations;New York},{NATO;Brussels},	{[X] based in [Y]},{[X] headquarters in [Y]},{[X] head office in [Y]}, {[X] main office building in [Y]},{[X] campus branch in [Y]}
affiliation	{Google;Marissa Mayer},{Xerox;Ursula Burns}, { Microsoft;Steve Ballmer},{Microsoft;Bill Gates},	{[X] CEO [Y]},{[X] resign from [Y]},{[X] founded by [Y]}, {[X] worked for [Y]},{[X] chairman director [Y]}

Table 2: Seed Entity Pairs and Templates for each relation. [X] and [Y] are slots for entity type tags.

$N_+(\lambda, \mathcal{G})$ for an extractor λ within the three systems. Observe that an instance i in λ can scale its $N_+(\lambda, \mathcal{G})$ by a factor of maximum 2 in BREJ if i is matched in both entity pair and template seeds. The reliability cnf$(\lambda, \mathcal{G})$ (Eq. 11) of an extractor λ is based on the ratio $\frac{N_+(\lambda,\mathcal{G}_n)}{N_+(\lambda,\mathcal{G}_p)}$, therefore suggesting that the scaling boosts its confidence.

In Figure 6, we demonstrate with an example how the joint bootstrapping scales up the positive instances for a *non-noisy* extractor λ, resulting in λ_{NNHC} for BREJ compared to λ_{NNLC} in BREE.

Due to unlabeled data, the instances not matching in seeds are considered either to be ignored/unknown N_0 or negatives in the confidence measure (Eq. 11). The former leads to high confidences for noisy extractors by assigning high scores, the latter to low confidences for non-noisy extractors by penalizing them. For a simple version of the algorithm in the illustration, we consider them as negatives and set $w_n = 1$. Figure 6 shows the three extractors (λ) generated and their confidence scores in BREE, BRET and BREJ. Observe that the scaling up of positives in BREJ due to BRET extractions (without w_n) discounts cnf$(\lambda, \mathcal{G})$ relatively lower than BREE. The discounting results in λ_{NNHC} in BREJ and λ_{NNLC} in BREE. The discounting in BREJ is adapted for *non-noisy* extractors facilitated by BRET in generating mostly non-noisy extractors due to stringent checks (Figure 3, line "(i)" and 05). Intuitively, the intermixing of non-noisy extractors (i.e., *hybrid*) promotes the scaling and boosts recall.

2.4 Similarity Measures

The before ($\vec{v}_{-1}$) and after ($\vec{v}_1$) contexts around the entities are highly sparse due to large variation in the syntax of how relations are expressed. SnowBall, DIPRE and BREE assumed that the between ($\vec{v}_0$) context mostly defines the syntactic expression for a relation and used weighted mechanism on the three contextual similarities in

	ORG-ORG	*ORG-PER*	*ORG-LOC*
count	58,500	75,600	95,900

Table 3: Count of entity-type pairs in corpus

Parameter	*Description/ Search*	*Optimal*
$\|v_{-1}\|$	maximum number of tokens in before context	2
$\|v_0\|$	maximum number of tokens in between context	6
$\|v_1\|$	maximum number of tokens in after context	2
τ_{sim}	similarity threshold [0.6, 0.7, 0.8]	0.7
τ_{cnf}	instance confidence thresholds [0.6, 0.7, 0.8]	0.7
w_n	weights to negative extractions [0.0, 0.5, 1.0, 2.0]	0.5
w_u	weights to unknown extractions [0.0001, 0.00001]	0.0001
k_{it}	number of bootstrapping epochs	3
dim_{emb}	dimension of embedding vector, V	300
PMI	PMI threshold in evaluation	0.5
Entity Pairs	Ordered Pairs (OP) or Bisets (BS)	OP

Table 4: Hyperparameters in BREE, BRET and BREJ

pairs, sim$_{match}$ (Figure 4). They assigned higher weights to the similarity in between ($p = 0$) contexts, that resulted in lower recall. We introduce attentive (max) similarity across all contexts (for example, $\vec{v}_{-1}(i)\vec{v}_0(j)$) to automatically capture the large variation in the syntax of how relations are expressed, without using any weights. We investigate asymmetric (Eq 13) and symmetric (Eq 14 and 15) similarity measures, and name them as *cross-context attentive* (sim$_{cc}$) similarity.

3 Evaluation

3.1 Dataset and Experimental Setup

We re-run BREE (Batista et al., 2015) for **baseline** with a set of 5.5 million news articles from AFP and APW (Parker et al., 2011). We use processed dataset of 1.2 million sentences (released by BREE) containing at least two entities linked to FreebaseEasy (Bast et al., 2014). We extract four relationships: *acquired* (ORG-ORG), *founder-of* (ORG-PER), *headquartered* (ORG-LOC) and *affiliation* (ORG-PER) for Organization (ORG), Person (PER) and Location (LOC) entity types. We bootstrap relations in BREE, BRET and BREJ, each with 4 similarity measures using seed entity

	Relationships	$\#out$	P	R	$F1$	$\#out$	P	R	$F1$	$\#out$	P	R	$F1$	$\#out$	P	R	$F1$
		baseline: BREE+sim$_{match}$				**config$_2$**: BREE+sim$_{cc}^{asym}$				**config$_3$**: BREE+sim$_{cc}^{sym1}$				**config$_4$**: BREE+sim$_{cc}^{sym2}$			
BREE	acquired	2687	0.88	0.48	0.62	5771	0.88	0.66	0.76	3471	0.88	0.55	0.68	3279	0.88	0.53	0.66
	founder-of	628	0.98	0.70	0.82	9553	0.86	0.95	0.89	1532	0.94	0.84	0.89	1182	0.95	0.81	0.87
	headquartered	16786	0.62	0.80	0.69	21299	0.66	0.85	0.74	17301	0.70	0.83	0.76	9842	0.72	0.74	0.73
	affiliation	20948	0.99	0.73	0.84	27424	0.97	0.78	0.87	36797	0.95	0.82	0.88	28416	0.97	0.78	0.87
	avg	10262	0.86	0.68	0.74	16011	0.84	0.81	0.82	14475	0.87	0.76	0.80	10680	0.88	0.72	0.78
		config$_5$: BRET+sim$_{match}$				**config$_6$**: BRET+sim$_{cc}^{asym}$				**config$_7$**: BRET+sim$_{cc}^{sym1}$				**config$_8$**: BRET+sim$_{cc}^{sym2}$			
BRET	acquired	4206	0.99	0.62	0.76	15666	0.90	0.85	0.87	18273	0.87	0.86	0.87	14319	0.92	0.84	0.87
	founder-of	920	0.97	0.77	0.86	43554	0.81	0.98	0.89	41978	0.81	0.99	0.89	46453	0.81	0.99	0.89
	headquartered	3065	0.98	0.55	0.72	39267	0.68	0.92	0.78	36374	0.71	0.91	0.80	56815	0.69	0.94	0.80
	affiliation	20726	0.99	0.73	0.85	28822	0.99	0.79	0.88	44946	0.96	0.85	0.90	33938	0.97	0.81	0.89
	avg	7229	0.98	0.67	0.80	31827	0.85	0.89	0.86	35393	0.84	0.90	0.86	37881	0.85	0.90	0.86
		config$_9$: BREJ+sim$_{match}$				**config$_{10}$**: BREJ+sim$_{cc}^{asym}$				**config$_{11}$**: BREJ+sim$_{cc}^{sym1}$				**config$_{12}$**: BREJ+sim$_{cc}^{sym2}$			
BREJ	acquired	20186	0.82	**0.87**	**0.84**	35553	0.80	**0.92**	0.86	22975	0.86	0.89	0.87	22808	0.85	0.90	**0.88**
	founder-of	45005	0.81	**0.99**	**0.89**	57710	0.81	**1.00**	**0.90**	50237	0.81	0.99	0.89	45374	0.82	0.99	0.90
	headquartered	47010	0.64	**0.93**	**0.76**	66563	0.68	**0.96**	**0.80**	60495	0.68	0.94	0.79	57853	0.68	0.94	0.79
	affiliation	40959	0.96	**0.84**	**0.89**	57301	0.94	**0.88**	**0.91**	55811	0.94	0.87	0.91	51638	0.94	0.87	0.90
	avg	**38290**	0.81	**0.91**	**0.85**	**54282**	0.81	**0.94**	**0.87**	47380	0.82	0.92	0.87	44418	0.82	0.93	0.87

Table 5: Precision (P), Recall (R) and $F1$ compared to the state-of-the-art (*baseline*). $\#out$: count of output instances with cnf$(i, \Lambda, \mathcal{G}) \geqslant 0.5$. **avg**: average. **Bold** and underline: Maximum due to BREJ and sim$_{cc}$, respectively.

pairs and templates (Table 2). See Tables 3, 4 and 5 for the count of candidates, hyperparameters and different configurations, respectively.

Our evaluation is based on Bronzi et al. (2012)'s framework to estimate precision and recall of large-scale RE systems using FreebaseEasy (Bast et al., 2014). Also following Bronzi et al. (2012), we use Pointwise Mutual Information (PMI) (Turney, 2001) to evaluate our system automatically, in addition to relying on an external knowledge base. We consider only extracted relationship instances with confidence scores cnf$(i, \Lambda, \mathcal{G})$ equal or above 0.5. We follow the same approach as BREE (Batista et al., 2015) to detect the correct order of entities in a relational triple, where we try to identify the presence of passive voice using part-of-speech (POS) tags and considering any form of the verb to be, followed by a verb in the past tense or past participle, and ending in the word 'by'. We use GloVe (Pennington et al., 2014) embeddings.

3.2 Results and Comparison with baseline

Table 5 shows the experimental results in the three systems for the different relationships with *ordered* entity pairs and similarity measures (sim$_{match}$, sim$_{cc}$). Observe that BRET (config$_5$) is *precision-oriented* while BREJ (config$_9$) *recall-oriented* when compared to BREE (baseline). We see the number of output instances $\#out$ are also higher in BREJ, therefore the higher recall. The BREJ system in the different similarity configurations outperforms the baseline BREE and BRET in terms of $F1$ score. On an average for the four relations, BREJ in configurations config$_9$ and config$_{10}$ results in $F1$ that is 0.11 (0.85 vs 0.74) and 0.13 (0.87 vs 0.74) better than the baseline BREE.

We discover that sim$_{cc}$ improves $\#out$ and *recall* over sim$_{\text{match}}$ correspondingly in all three systems. Observe that sim$_{cc}$ performs better with BRET than BREE due to *non-noisy* extractors in BRET. The results suggest an alternative to the weighting scheme in sim$_{\text{match}}$ and therefore, the state-of-the-art (sim$_{cc}$) performance with the 3 parameters (w_{-1}, w_0 and w_1) ignored in bootstrap-

τ	k_{it}	$\#out$	P	R	$F1$
0.6	1	691	0.99	0.21	0.35
	2	11288	0.85	**0.79**	0.81
0.7	1	610	1.0	0.19	0.32
	2	7948	**0.93**	0.75	**0.83**
0.8	1	522	1.0	0.17	0.29
	2	2969	0.90	0.51	0.65

Table 6: Iterations (k_{it}) Vs Scores with thresholds (τ) for relation *acquired* in BREJ. τ refers to τ_{sim} and τ_{cnf}

	τ	$\#out$	P	R	$F1$	τ	$\#out$	P	R	$F1$
BREE	.60	1785	.91	.39	.55	.70	1222	.94	.31	.47
	.80	868	.95	.25	.39	.90	626	.96	.19	.32
BRET	.60	2995	.89	.51	.65	.70	1859	.90	.40	.55
	.80	1312	.91	.32	.47	.90	752	.94	.22	.35
BREJ	.60	18271	.81	.85	.83	.70	14900	.84	.83	.83
	.80	8896	.88	.75	.81	.90	5158	.93	.65	.77

Table 7: Comparative analysis using different thresholds τ to evaluate the extracted instances for *acquired*

BREX	*acquired*			*founder-of*			*headquartered*			*affiliation*		
	E	**T**	**J**	**E**	**T**	**J**	**E**	**T**	**J**	**E**	**T**	**J**
$\#hit$	71	682	743	135	956	1042	715	3447	4023	603	14888	15052

Table 8: Disjunctive matching of Instances. $\#hit$: the count of instances matched to positive seeds in $k_{it} = 1$

Attributes		$\|\Lambda\|$	*AIE*	*AES*	*ANE*	*ANNE*	*ANNLC*	*AP*	*AN*	*ANP*
acquired	BREE	167	12.7	0.51	0.84	0.16	0.14	37.7	93.1	2.46
	BRET	17	305.2	1.00	0.11	0.89	0.00	671.8	0.12	0.00
	BREJ	555	**41.6**	**0.74**	**0.71**	**0.29**	**0.03**	**313.2**	**44.8**	**0.14**
founder-of	BREE	8	13.3	0.46	0.75	0.25	0.12	44.9	600.5	13.37
	BRET	5	179.0	1.00	0.00	1.00	0.00	372.2	0.0	0.00
	BREJ	492	**109.1**	**0.90**	0.94	0.06	**0.00**	**451.8**	**79.5**	**0.18**
headquartered	BREE	655	18.4	0.60	0.97	0.03	0.02	46.3	82.7	1.78
	BRET	7	365.7	1.00	0.00	1.00	0.00	848.6	0.0	0.00
	BREJ	1311	**45.5**	**0.80**	0.98	0.02	**0.00**	**324.1**	**77.5**	**0.24**
affiliation	BREE	198	99.7	0.55	0.25	0.75	0.34	240.5	152.2	0.63
	BRET	19	846.9	1.00	0.00	1.00	0.00	2137.0	0.0	0.00
	BREJ	470	**130.2**	**0.72**	**0.21**	**0.79**	**0.06**	**567.6**	**122.7**	**0.22**

Table 9: Analyzing the attributes of extractors Λ learned for each relationship. Attributes are: number of extractors ($|\Lambda|$), avg number of instances in Λ (AIE), avg Λ score (AES), avg number of noisy Λ (ANE), avg number of non-noisy Λ (ANNE), avg number of Λ_{NNLC} below confidence 0.5 (ANNLC), avg number of positives (AP) and negatives (AN), ratio of AN to AP (ANP). The **bold** indicates comparison of BREE and BREJ with sim_{match}. avg: average

ping. Observe that sim_{cc}^{asym} gives higher recall than the two symmetric similarity measures.

Table 6 shows the performance of BREJ in different iterations trained with different similarity τ_{sim} and confidence τ_{cnf} thresholds. Table 7 shows a comparative analysis of the three systems, where we consider and evaluate the extracted relationship instances at different confidence scores.

3.3 Disjunctive Seed Matching of Instances

As discussed in section 2.3, BREJ facilitates disjunctive matching of instances (line 05 Figure 3) with seed entity pairs and templates. Table 8 shows $\#hit$ in the three systems, where the higher values of $\#hit$ in BREJ conform to the desired property. Observe that some instances in BREJ are found to be matched in both the seed types.

3.4 Deep Dive into Attributes of Extractors

We analyze the extractors Λ generated in BREE, BRET and BREJ for the 4 relations to demonstrate the impact of joint bootstrapping. Table 9 shows the attributes of Λ. We manually annotate the extractors as *noisy* and *non-noisy*. We compute $ANNLC$ and the lower values in BREJ compared to BREE suggest fewer non-noisy extractors with lower confidence in BREJ due to the scaled confi-

	Relationships	$\#out$	P	R	$F1$
BREE	acquired	387	0.99	0.13	0.23
	founder-of	28	0.96	0.09	0.17
	headquartered	672	0.95	0.21	0.34
	affiliation	17516	0.99	0.68	0.80
	avg	4651	0.97	0.28	0.39
BRET	acquired	4031	1.00	0.61	0.76
	founder-of	920	0.97	0.77	0.86
	headquartered	3522	0.98	0.59	0.73
	affiliation	22062	0.99	0.74	0.85
	avg	7634	0.99	0.68	0.80
BREJ	acquired	12278	0.87	0.81	**0.84**
	founder-of	23727	0.80	0.99	**0.89**
	headquartered	38737	0.61	0.91	**0.73**
	affiliation	33203	0.98	0.81	**0.89**
	avg	26986	0.82	0.88	**0.84**

Table 10: BREX+$\text{sim}_{\text{match}}$:Scores when w_n ignored

dence scores. *ANNE* (higher), *ANNLC* (lower), *AP* (higher) and *AN* (lower) collectively indicate that BRET mostly generates *NNHC* extractors. *AP* and *AN* indicate an average of $N_+(\lambda, \mathcal{G}_l)$ (line " (i)" Figure 3) for positive and negative seeds, respectively for $\lambda \in \Lambda$ in the three systems. Observe the impact of scaling positive extractions (*AP*) in BREJ that shrink $\frac{N_+(\lambda,\mathcal{G}_n)}{N_+(\lambda,\mathcal{G}_p)}$ i.e., *ANP*. It facilitates λ_{NNLC} to boost its confidence, i.e., λ_{NNHC} in BREJ suggested by *AES* that results in higher $\#out$ and recall (Table 5, BREJ).

3.5 Weighting Negatives Vs Scaling Positives

As discussed, Table 5 shows the performance of BREE, BRET and BREJ with the parameter $w_n = 0.5$ in computing extractors' confidence $\text{cnf}(\lambda, \mathcal{G})$(Eq. 11). In other words, config_9 (Table 5) is combination of both weighted negative and scaled positive extractions. However, we also investigate ignoring $w_n(= 1.0)$ in order to demonstrate the capability of BREJ with only scaling positives and without weighting negatives. In Table 10, observe that BREJ outperformed both BREE and BRET for all the relationships due to higher $\#out$ and recall. In addition, BREJ scores are comparable to config_9 (Table 5) suggesting that the scaling in BREJ is capable enough to remove the parameter w_n. However, the combination of both weighting negatives and scaling positives results in the state-of-the-art performance.

3.6 Qualitative Inspection of Extractors

Table 11 lists some of the non-noisy extractors (simplified) learned in different configurations to illustrate boosting extractor confidence $\text{cnf}(\lambda, \mathcal{G})$. Since, an extractor λ is a cluster of instances, therefore to simplify, we show one in-

config$_1$: BREE + sim$_{\text{match}}$	cnf$(\lambda, \mathbf{G})$	config$_5$: BRET + sim$_{\text{match}}$	cnf$(\lambda, \mathbf{G})$	config$_9$: BREJ + sim$_{\text{match}}$	cnf$(\lambda, \mathbf{G})$	config$_{10}$: BREJ + sim$_{cc}^{\text{asym}}$	cnf$(\lambda, \mathbf{G})$
acquired							
[X] acquired [Y]	0.98	[X] acquired [Y]	1.00	[X] acquired [Y]	1.00	acquired by [X] , [Y] †	0.93
[X] takeover of [Y]	0.89	[X] takeover of [Y]	1.00	[X] takeover of [Y]	0.98	takeover of [X] would boost [Y] 's earnings †	0.90
[X] 's planned acquisition of [Y]	0.87	[X] 's planned acquisition of[Y]	1.00	[X] 's planned acquisition of [Y]	0.98	acquisition of [X] by [Y] †	0.95
[X] acquiring [Y]	0.75	[X] acquiring [Y]	1.00	[X] acquiring [Y]	0.95	[X] acquiring [Y]	0.95
[X] has owned part of [Y]	0.67	[X] has owned part of [Y]	1.00	[X] has owned part of [Y]	0.88	owned by [X] 's parent [Y]	0.90
[X] took control of [Y]	0.49	[X] 's ownership of [Y]	1.00	[X] took control of [Y]	0.91	[X] takes control of [Y]	1.00
[X] 's acquisition of [Y]	0.35	[X] 's acquisition of [Y]	1.00	[X] 's acquisition of [Y]	0.95	acquisition of [X] would reduce [Y] 's share †	0.90
[X] 's merger with [Y]	0.35	[X] 's merger with[Y]	1.00	[X] 's merger with [Y]	0.94	[X] - [Y] merger between †	0.84
[X] 's bid for [Y]	0.35	[X] 's bid for [Y]	1.00	[X] 's bid for [Y]	0.97	part of [X] which [Y] acquired †	0.83
founder-of							
[X] founder [Y]	0.68	[X] founder [Y]	1.00	[X] founder [Y]	0.99	founder of [X] , [Y] †	0.97
[X] CEO and founder [Y]	0.15	[X] CEO and founder [Y]	1.00	[X] CEO and founder [Y]	0.99	co-founder of [X] 's millennial center , [Y] †	0.94
[X] 's co-founder [Y]	0.09	[X] owner [Y]	1.00	[X] owner [Y]	1.00	owned by [X] cofounder [Y]	0.95
		[X] cofounder [Y]	1.00	[X] cofounder [Y]	1.00	Gates co-founded [X] with school friend [Y] †	0.99
		[X] started by [Y]	1.00	[X] started by [Y]	1.00	who co-founded [X] with [Y] †	0.95
		[X] was founded by [Y]	1.00	[X] was founded by [Y]	0.99	to co-found [X] with partner [Y] †	0.68
		[X] begun by [Y]	1.00	[X] begun by [Y]	1.00	[X] was started by [Y] , cofounder	0.98
		[X] has established [Y]	1.00	[X] has established [Y]	0.99	set up [X] with childhood friend [Y] †	0.96
		[X] chief executive and founder [Y]	1.00	[X] co-founder and billionaire [Y] *	0.99	[X] co-founder and billionaire [Y]	0.97
headquartered							
[X] headquarters in [Y]	0.95	[X] headquarters in [Y]	1.00	[X] headquarters in [Y]	0.98	[X] headquarters in [Y]	0.98
[X] relocated its headquarters from [Y]	0.94	[X] relocated its headquarters from [Y]	1.00	[X] relocated its headquarters from [Y]	0.98	based at [X] 's suburban [Y] headquarters †	0.98
[X] head office in [Y]	0.84	[X] head office in [Y]	1.00	[X] head office in [Y]	0.87	head of [X] 's operations in [Y] †	0.65
[X] based in [Y]	0.75	[X] based in [Y]	1.00	[X] based in [Y]	0.98	branch of [X] company based in [Y]	0.98
[X] headquarters building in [Y]	0.67	[X] headquarters building in [Y]	1.00	[X] headquarters building in [Y]	0.94	[X] main campus in [Y]	0.99
[X] headquarters in downtown [Y]	0.64	[X] headquarters in downtown [Y]	1.00	[X] headquarters in downtown [Y]	0.94	[X] headquarters in downtown [Y]	0.96
[X] branch offices in [Y]	0.54	[X] branch offices in [Y]	1.00	[X] branch offices in [Y]	0.98	[X] 's [Y] headquarters represented †	0.98
[X] 's corporate campus in [Y]	0.51	[X] 's corporate campus in [Y]	1.00	[X] 's corporate campus in [Y]	0.99	[X] main campus in [Y]	0.99
[X] 's corporate office in [Y]	0.51	[X] 's corporate office in [Y]	1.00	[X] 's corporate office in [Y]	0.89	[X] , [Y] 's corporate †	0.94
affiliation							
[X] chief executive [Y]	0.92	[X] chief executive [Y]	1.00	[X] chief executive [Y]	0.97	[X] chief executive [Y] resigned monday	0.94
[X] secretary [Y]	0.88	[X] secretary [Y]	1.00	[X] secretary [Y]	0.94	worked with [X] manager [Y]	0.85
[X] president [Y]	0.87	[X] president [Y]	1.00	[X] president [Y]	0.96	[X] voted to retain [Y] as CEO †	0.98
[X] leader [Y]	0.72	[X] leader [Y]	1.00	[X] leader [Y]	0.85	head of [X] , [Y] †	0.99
[X] party leader [Y]	0.67	[X] party leader [Y]	1.00	[X] party leader [Y]	0.87	working with [X] , [Y] suggested †	1.00
[X] has appointed [Y]	0.63	[X] executive editor [Y]	1.00	[X] has appointed [Y]	0.81	[X] president [Y] was fired	0.90
[X] player [Y]	0.38	[X] player [Y]	1.00	[X] player [Y]	0.89	[X] 's [Y] was fired †	0.43
[X] 's secretary-general [Y]	0.36	[X] 's secretary-general [Y]	1.00	[X] 's secretary-general [Y]	0.93	Chairman of [X] , [Y] †	0.88
[X] hired [Y]	0.21	[X] director [Y]	1.00	[X] hired [Y]	0.56	[X] hired [Y] as manager †	0.85

Table 11: Subset of the non-noisy extractors (simplified) with their confidence scores cnf$(\lambda, \mathcal{G})$ learned in different configurations for each relation. $*$ denotes that the extractor was never learned in config$_1$ and config$_5$. † indicates that the extractor was never learned in config$_1$, config$_5$ and config$_9$. [X] and [Y] indicate placeholders for entities.

stance (mostly populated) from every λ. Each cell in Table 11 represents either a simplified representation of λ or its confidence. We demonstrate how the confidence score of a non-noisy extractor in BREE (config$_1$) is increased in BREJ (config$_9$ and config$_{10}$). For instance, for the relation *acquired*, an extractor $\{$*[X] acquiring [Y]*$\}$ is generated by BREE, BRET and BREJ; however, its confidence is boosted from 0.75 in BREE (config$_1$) to 0.95 in BREJ (config$_9$). Observe that BRET generates high confidence extractors. We also show extractors (marked by †) learned by BREJ with sim$_{cc}$ (config$_{10}$) but not by config$_1$, config$_5$ and config$_9$.

3.7 Entity Pairs: Ordered Vs Bi-Set

In Table 5, we use ordered pairs of typed entities. Additionally, we also investigate using entity sets and observe improved recall due to higher $\#out$ in both BREE and BREJ, comparing correspondingly Table 12 and 5 (*baseline* and config$_9$).

4 Conclusion

We have proposed a Joint Bootstrapping Machine for relation extraction (BREJ) that takes advantage of both entity-pair-centered and template-centered approaches. We have demonstrated that the joint approach scales up positive instances that boosts the confidence of NNLC extractors and improves recall. The experiments showed that the cross-context similarity measures improved recall and suggest removing in total four parameters.

Relationships	**BREE** + sim$_{\text{match}}$				**BREJ** + sim$_{\text{match}}$			
	$\#out$	P	R	$F1$	$\#out$	P	R	$F1$
acquired	2786	.90	.50	.64	21733	.80	.87	.83
founder-of	543	1.0	.67	.80	31890	.80	.99	.89
headquartered	16832	.62	.81	.70	52286	.64	.94	.76
affiliation	21812	.99	.74	.85	42601	.96	.85	.90
avg	10493	.88	.68	.75	37127	.80	.91	.85

Table 12: BREX+sim$_{\text{match}}$:Scores with entity *bisets*

Acknowledgments

We thank our colleagues Bernt Andrassy, Mark Buckley, Stefan Langer, Ulli Waltinger and Usama Yaseen, and anonymous reviewers for their review comments. This research was supported by Bundeswirtschaftsministerium (bmwi.de), grant 01MD15010A (Smart Data Web) at Siemens AG-CT Machine Intelligence, Munich Germany.

References

Eugene Agichtein and Luis Gravano. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the 15th ACM conference on Digital libraries*. Association for Computing Machinery, Washington, DC USA, pages 85–94.

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D Manning. 2015. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, Beijing, China, volume 1, pages 344–354.

Hannah Bast, Florian Bäurle, Björn Buchhold, and Elmar Haußmann. 2014. Easy access to the freebase dataset. In *Proceedings of the 23rd International Conference on World Wide Web*. Association for Computing Machinery, Seoul, Republic of Korea, pages 95–98.

David S. Batista, Bruno Martins, and Mário J. Silva. 2015. Semi-supervised bootstrapping of relationship extractors with distributional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 499–504.

Sergey Brin. 1998. Extracting patterns and relations from the world wide web. In *International Workshop on The World Wide Web and Databases*. Springer, Valencia, Spain, pages 172–183.

Mirko Bronzi, Zhaochen Guo, Filipe Mesquita, Denilson Barbosa, and Paolo Merialdo. 2012. Automatic evaluation of relation extraction systems on large-scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*. Association for Computational Linguistics, Montrèal, Canada, pages 19–24.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the 24th National Conference on Artificial Intelligence (AAAI)*. Atlanta, Georgia USA, volume 5, page 3.

Laura Chiticariu, Yunyao Li, and Frederick R. Reiss. 2013. Rule-based information extraction is dead! long live rule-based information extraction systems! In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington USA, pages 827–832.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Edinburgh, Scotland UK, pages 1535–1545.

Pankaj Gupta, Thomas Runkler, Heike Adel, Bernt Andrassy, Hans-Georg Zimmermann, and Hinrich Schütze. 2015. Deep learning methods for the extraction of relations in natural language text. Technical report, Technical University of Munich, Germany.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan, pages 2537–2547.

Sonal Gupta, Diana L. MacLean, Jeffrey Heer, and Christopher D. Manning. 2014. Induced lexico-syntactic patterns improve information extraction from online medical forums. *Journal of the American Medical Informatics Association* 21(5):902–909.

Sonal Gupta and Christopher Manning. 2014. Improved pattern learning for bootstrapped entity extraction. In *Proceedings of the 18th Conference on Computational Natural Language Learning (CoNLL)*. Association for Computational Linguistics, Baltimore, Maryland USA, pages 98–108.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 15th International Conference on Computational Linguistics*. Nantes, France, pages 539–545.

Winston Lin, Roman Yangarber, and Ralph Grishman. 2003. Bootstrapped learning of semantic classes from positive and negative examples. In *Proceedings of ICML 2003 Workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*. Washington, DC USA, page 21.

Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. 2012. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, Jeju Island, Korea, pages 523–534.

Filipe Mesquita, Jordan Schmidek, and Denilson Barbosa. 2013. Effectiveness and efficiency of open relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington USA, pages 447–457.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the Workshop at the International Conference on Learning Representations*. ICLR, Scottsdale, Arizona USA.

Thien Huu Nguyen and Ralph Grishman. 2015. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*. Association for Computational Linguistics, Denver, Colorado USA, pages 39–48.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword. *Linguistic Data Consortium* .

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Doha, Qatar, pages 1532–1543.

Ellen Riloff. 1996. Automatically generating extraction patterns from untagged text. In *Proceedings of the 13th National Conference on Artificial Intelligence (AAAI)*. Portland, Oregon USA, pages 1044–1049.

Peter D. Turney. 2001. Mining the web for synonyms: Pmi-ir versus lsa on toefl. In *Proceedings of the 12th European Conference on Machine Learning*. Springer, Freiburg, Germany, pages 491–502.

Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schütze. 2016a. Combining recurrent and convolutional neural networks for relation classification. In *Proceedings of the 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*. Association for Computational Linguistics, San Diego, California USA, pages 534–539.

Ngoc Thang Vu, Pankaj Gupta, Heike Adel, and Hinrich Schütze. 2016b. Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Shanghai, China, pages 6060–6064.

Chapter 4

Neural Relation Extraction Within and Across Sentence Boundaries

Neural Relation Extraction Within and Across Sentence Boundaries

Pankaj Gupta[1,2], Subburam Rajaram[1], Hinrich Schütze[2], Thomas Runkler[1]
[1]Corporate Technology, Machine-Intelligence (MIC-DE), Siemens AG Munich, Germany
[2]CIS, University of Munich (LMU) Munich, Germany
{pankaj.gupta, subburam.rajaram}@siemens.com| pankaj.gupta@campus.lmu.de

Abstract

Past work in relation extraction mostly focuses on binary relation between entity pairs *within single sentence*. Recently, the NLP community has gained interest in relation extraction in entity pairs *spanning multiple sentences*. In this paper, we propose a novel architecture for this task: inter-sentential dependency-based neural networks (iDepNN). iDepNN models the shortest and augmented dependency paths via recurrent and recursive neural networks to extract relationships within (intra-) and across (inter-) sentence boundaries. Compared to SVM and neural network baselines, iDepNN is more robust to false positives in relationships spanning sentences. We evaluate our models on four datasets from newswire (MUC6) and medical (BioNLP shared task) domains that achieve state-of-the-art performance and show a better balance in precision and recall for inter-sentential relationships. We perform better than 11 teams participating in the BioNLP shared task 2016 and achieve a gain of 5.2% (0.587 vs 0.558) in F_1 over the winning team. We also release the cross-sentence annotations for MUC6.

Introduction

The task of relation extraction (RE) aims to identify semantic relationship between a pair of nominals or entities *e1* and *e2* in a given sentence S. Due to a rapid growth in information, it plays a vital role in knowledge extraction from unstructured texts and serves as an intermediate step in a variety of NLP applications in newswire, web and high-valued biomedicine (Bahcall 2015) domains. Consequently, there has been increasing interest in relation extraction, particularly in augmenting existing knowledge bases.

Progress in relation extraction is exciting; however most prior work (Zhang et al. 2006; Kambhatla 2004; Vu et al. 2016a; Gupta, Schütze, and Andrassy 2016) is limited to single sentences, i.e., *intra-sentential* relationships, and ignores relations in entity pairs spanning sentence boundaries, i.e., *inter-sentential*. Thus, there is a need to move beyond single sentences and devise methods to extract relationships spanning sentences. For instance, consider the sentences:

Paul Allen has started a company and named [*Vern Raburn*]$_{e1}$ its President. The company, to be called [*Paul Allen Group*]$_{e2}$ will be based in Bellevue, Washington.

The two sentences together convey the fact that the entity *e1* is associated with *e2*, which cannot be inferred from either sentence alone. The missed relations impact the system performance, leading to poor recall. But precision is equally important; e.g., in high-valued biomedicine domain, significant inter-sentential relationships must be extracted, especially in medicine that aims toward accurate diagnostic testing and precise treatment, and extraction errors can have severe negative consequences. In this work, we present a neural network (NN) based approach to precisely extract relationships within and across sentence boundaries, and show a better balance in precision and recall with an improved F_1.

Previous work on cross-sentence relation extraction used coreferences to access entities that occur in a different sentence (Gerber and Chai 2010; Yoshikawa et al. 2011) without modeling inter-sentential relational patterns. Swampillai and Stevenson (2011) described a SVM-based approach to both intra- and inter-sentential relations. Recently, Quirk and Poon (2016) applied distant supervision to cross-sentence relation extraction of entities using binary logistic regression (non-neural network based) classifier and Peng et al. (2017) applied sophisticated graph long short-term memory networks to cross-sentence n-ary relation extraction. However, it still remains challenging due to the need for coreference resolution, noisy text between the entity pairs spanning multiple sentences and lack of labeled corpora.

Bunescu and Mooney (2005), Nguyen, Matsuo, and Ishizuka (2007) and Mintz et al. (2009) have shown that the shortest dependency path (SDP) between two entities in a dependency graph and the dependency subtrees are the most useful dependency features in relation classification. Further, Liu et al. (2015) developed these ideas using Recursive Neural Networks (*RecNNs*, Socher et al. (2014)) and combined the two components in a precise structure called Augmented Dependency Path (ADP), where each word on a SDP is attached to a dependency subtree; however, limited to single sentences. In this paper, we aspire from these methods to extend shortest dependency path across sentence boundary and effectively combine it with dependency subtrees in NNs that can capture semantic representation of the structure and boost relation extraction spanning sentences.

The *contributions* are: **(1)** Introduce a novel dependency-based neural architecture, named as inter-sentential Dependency-based Neural Network (iDepNN) to extract re-

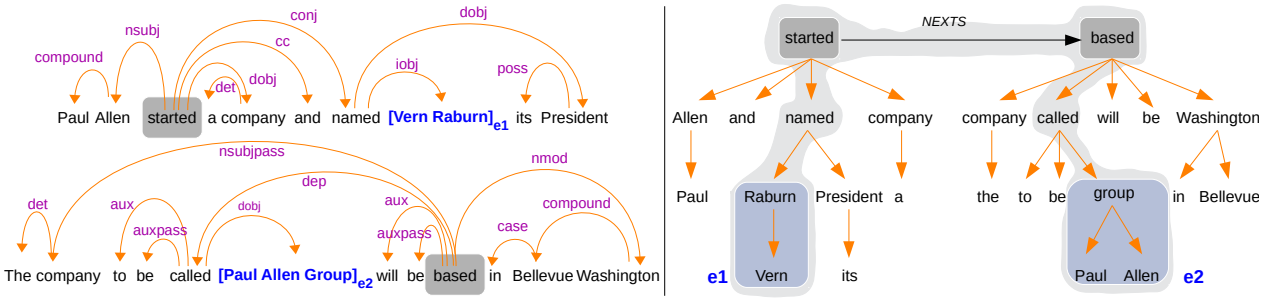


Figure 1: Left: Sentences and their dependency graphs. Right: Inter-sentential Shortest Dependency Path (iSDP) across sentence boundary. Connection between the roots of adjacent sentences by *NEXTS*.

lations within and across sentence boundaries by modeling shortest and augmented dependency paths in a combined structure of bidirectional RNNs (biRNNs) and RecNNs. **(2)** Evaluate different linguistic features on four datasets from newswire and medical domains, and report an improved performance in relations spanning sentence boundary. We show amplified precision due to robustness towards false positives, and a better balance in precision and recall. We perform better than 11 teams participating in in the BioNLP shared task 2016 and achieve a gain of 5.2% (0.587 vs 0.558) in F_1 over the winning team. **(3)** Release relation annotations for the MUC6 dataset for intra- and inter-sentential relationships. *Code*, *data* and *supplementary* are available at `https://github.com/pgcool/Cross-sentence-Relation-Extraction-iDepNN`.

Methodology

Inter-sentential Dependency-Based Neural Networks (iDepNN)

Dependency-based neural networks (DepNN) (Bunescu and Mooney 2005; Liu et al. 2015) have been investigated for relation extraction between entity pairs limited to single sentences, using the dependency information to explore the semantic connection between two entities. In this work, we introduce *iDepNN*, the inter-sentential Dependency-based Neural Network, an NN that models relationships between entity pairs spanning sentences, i.e., inter-sentential within a document. We refer to the iDepNN that only models the shortest dependency path (SDP) spanning sentence boundary as *iDepNN-SDP* and to the iDepNN that models augmented dependency paths (ADPs) as *iDepNN-ADP*; see below. biRNNs (bidirectional RNNs, Schuster and Paliwal (1997)) and RecNNs (recursive NNs, Socher et al. (2012)) are the backbone of iDepNN.

Modeling Inter-sentential Shortest Dependency Path (iDepNN-SDP): We compute the inter-sentential Shortest Dependency Path (iSDP) between entities spanning sentence boundaries for a relation. To do so, we obtain the dependency parse tree for each sentence using the Stanford-CoreNLP dependency parser (Manning et al. 2014). We then use NetworkX (Hagberg, Swart, and S Chult 2008) to represent each token as a node and the dependency relation as a link between the nodes. In the case of multiple sentences, the root node of the parse tree of a sentence is connected to the root of the subsequent tree, leading to the shortest path from one entity to another across sentences.

Figure 1 (Left) shows dependency graphs for the example sentences where the two entities *e1* and *e2* appear in nearby sentences and exhibit a relationship. Figure 1 (Right) illustrates that the dependency trees of the two adjacent sentences and their roots are connected by *NEXTS* to form an *iSDP*, an inter-Sentential Dependency Path, (highlighted in gray) between the two entities. The shortest path spanning sentence boundary is seen as a sequence of words between two entities. Figure 2 shows how a biRNN (Schuster and Paliwal 1997; Vu et al. 2016b) uses iSDP to detect relation between *e1* and *e2*, positioned one sentence apart.

Modeling Inter-sentential Dependency Subtrees: To effectively represent words on the shortest dependency path within and across sentence boundary, we model dependency subtrees assuming that each word w can be seen as the word itself and its children on the dependency subtree. The notion of representing words using subtree vectors within the dependency neural network (DepNN) is similar to (Liu et al. 2015); however, our proposed structures are based on iSDPs and ADPs that span sentences.

To represent each word w on the subtree, its word embedding vector $\mathbf{x}_w \in \mathcal{R}^d$ and subtree representation $\mathbf{c}_w \in \mathcal{R}^{d'}$ are concatenated to form its final representation $\mathbf{p}_w \in \mathcal{R}^{d+d'}$. We use 200-dimensional pretrained GloVe embeddings (Pennington, Socher, and Manning 2014). The subtree representation of a word is computed through recursive transformations of the representations of its children words. A RecNN is used to construct subtree embedding $\mathbf{c}_w$, traversing bottom-up from its leaf words to the root for entities spanning sentence boundaries, as shown in Figure 2. For a word which is a leaf node, i.e., it does not have a subtree, we set its subtree representation as $\mathbf{c}_{\text{LEAF}}$. Figure 2 illustrates how subtree-based word representations are constructed via iSDP.

Each word is associated with a dependency relation r, e.g., r = *dobj*, during the bottom-up construction of the subtree. For each r, a transformation matrix $\mathbf{W}_r \in \mathcal{R}^{d' \times (d+d')}$ is learned. The subtree embedding is computed as:

$$\mathbf{c}_w = f\Big(\sum_{q \in Children(w)} \mathbf{W}_{R_{(w,q)}} \cdot \mathbf{p}_q + \mathbf{b}\Big) \text{ and } \mathbf{p}_q = [\mathbf{x}_q, \mathbf{c}_q]$$

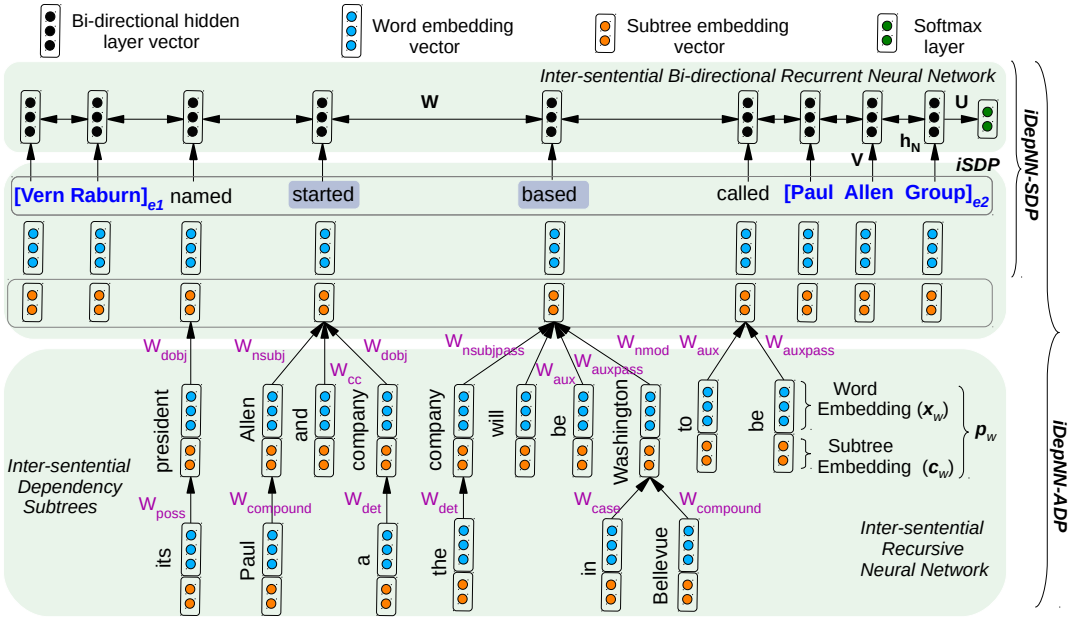


Figure 2: Inter-sentential Dependency-based Neural Network variants: iDepNN-SDP and iDepNN-ADP

where $R_{(w,q)}$ is the dependency relation between word w and its child word q and $\mathbf{b} \in \mathcal{R}^{d'}$ is a bias. This process continues recursively up to the root word such as the word "named" on the iSDP in the figure.

Modeling Inter-sentential Augmented Dependency Path (iDepNN-ADP): Following Liu et al. (2015), we combine the two components: iSDP and *dependency subtrees* spanning sentence boundaries to form a combined structure which we name as inter-sentential Augmented Dependency Path (*iDepNN-ADP*). As shown in Figure 2, each word on iSDP is attached to its subtree representation $\mathbf{c}_w$. An attached subtree enriches each word on the iSDP with additional information about how this word functions in specific sentence to form a more precise structure for classifying relationships within and across sentences.

To capture the semantic representation of *iDepNN-ADP*, we first adopt a RecNN to model the dependency subtrees for each word on the iSDP. Then, we design a biRNN to obtain salient semantic features on the iSDP. The overall structure of *iDepNN-ADP* (Figure 2) is built upon the combination of recursive and recurrent NNs spanning sentences.

Learning: We develop a biRNN over the two structures: iDepNN-SDP and iDepNN-ADP, and pass the last hidden vector $\mathbf{h_N}$ (in the iSDP word sequence, Figure 2) to a softmax layer whose output is the probability distribution $\mathbf{y}$ over relation labels R, as $\mathbf{y} = \text{softmax}(\mathbf{U} \cdot \mathbf{h}_N + \mathbf{b}_y)$ where $\mathbf{U} \in \mathcal{R}^{R \times H}$ is the weight matrix connecting hidden vector of dimension H to output of dimension R and $\mathbf{b}_y \in \mathcal{R}^R$ is the bias. $\mathbf{h}_N$ is the last hidden vector of the biRNN.

To compute semantic representation $\mathbf{h}_w$ for each word w on the iSDP, we adopt the *Connectionist biRNN* (Vu et al. 2016a) that combines the forward and backward pass by adding their hidden layers ($\mathbf{h}_{f_t}$ and $\mathbf{h}_{b_t}$) at each time step t and also adds a weighted connection to the previous combined hidden layer $\mathbf{h}_{t-1}$ to include all intermediate hidden layers into the final decision.

$$\mathbf{h}_{f_t} = f(\mathbf{V} \cdot \mathbf{i}_t + \mathbf{W} \cdot \mathbf{h}_{f_{t-1}})$$
$$\mathbf{h}_{b_t} = f(\mathbf{V} \cdot \mathbf{i}_{N-t+1} + \mathbf{W} \cdot \mathbf{h}_{b_{t+1}})$$
$$\mathbf{h}_t = f(\mathbf{h}_{f_t} + \mathbf{h}_{b_t} + \mathbf{W} \cdot \mathbf{h}_{t-1})$$

where $\mathbf{V} \in \mathcal{R}^{H \times |\mathbf{i}|}$, N is the total number of words on iSDP and $\mathbf{i}_t$ the input vector at t, defined by:

$$\text{iDepNN-SDP} : \mathbf{i}_t = [\mathbf{x}_t, \mathbf{L}_t] \qquad \text{iDepNN-ADP} : \mathbf{i}_t = [\mathbf{p}_t, \mathbf{L}_t]$$

where $\mathbf{L}_t$ are lexical level features (e.g., part-of-speech tag, position indicators, entity types) for each word at t. Observe, in order to minimize the number of parameters, we share the same weight matrix $\mathbf{W}$ in three parts: forward pass, backward pass and combination of both. The optimization objective is to minimize the cross-entropy error between the ground-truth label and softmax output. The parameters are learned using backpropagation (Werbos 1990).

Key Features: The features focus on characteristics of the full sentence, dependency path or individual entities. The various features used in our experiments are: (1) *Position-Indicator* (PI): A one-hot vector for SVM which indicates the position of the entity in the vocabulary. Four additional words ($<e_1>$, $</e_1>$, $<e_2>$, $</e_2>$) to mark start and end of entity mentions *e1* and *e2*, used in NNs. See details about PI in Gupta (2015). (2) *Entity Types* (ET): A one-hot vector to represent the entity type in SVM and embedding vectors in NNs. (3) *Part-of-speech* (POS): A bag-of-words (BoW) in SVM and embedding vector for each POS type in NNs. (4) *Dependency*: In SVM, the specific edge types in the dependency path are captured with a BoW vector, similar to

Relation	*Intra*	*Inter*	*Relation*	*Intra*	*Inter*
BioNLP ST 2011 (*Medical*)			BioNLP ST 2013 (*Medical*)		
*P*artOf	99	103	*P*artOf	104	83
*L*ocalization	261	732	*L*ocalization	246	677
Total	360	835 (**70%**)	Total	350	760 (**69%**)
BioNLP ST 2016 (*Medical*)			MUC6 (*News*)		
*L*ives_In	363	135	*P*er-Org	245	112
			*P*er-Post	407	66
			*O*rg-Post	268	113
Total	363	135 (**27%**)	Total	920	291 (**24%**)

Table 1: Count of intra- and inter-sentential relationships in datasets (train+dev) from two domains

Grouin (2016). In NNs, it refers to *iDepNN-ADP*. (5) [*inter-sentential-*]*Shortest-Dependency-Path* ([i-]SDP): Sequence of Words on the [i-]SDP.

Evaluation and Analysis

Dataset. We evaluate our proposed methods on four datasets from medical and news domain. Table 1 shows counts of intra- and inter-sentential relationships. The three medical domain datasets are taken from the BioNLP shared task (ST) of relation/event extraction (Bossy et al. 2011; Nédellec et al. 2013; Delèger et al. 2016). We compare our proposed techniques with the systems published at these venues. The Bacteria Biotope task (Bossy et al. 2011) of the BioNLP ST 2011 focuses on extraction of habitats of bacteria, which is extended by the BioNLP ST 2013 (Nédellec et al. 2013), while the BioNLP ST 2016 focuses on extraction of *Lives_in* events. We have standard train/dev/test splits for the BioNLP ST 2016 dataset, while we perform 3-fold cross-validation[1] on BioNLP ST 2011 and 2013. For BioNLP ST 2016, we generate negative examples by randomly sampling co-occurring entities without known interactions. Then we sample the same number as positives to obtain a balanced dataset during training and validation for different sentence range. See supplementary for further details.

The MUC6 (Grishman and Sundheim 1996) dataset contains information about management succession events from newswire. The task organizers provided a training corpus and a set of templates that contain the management succession events, the names of people who are starting or leaving management posts, the names of their respective posts and organizations and whether the named person is currently in the job. **Entity Tagging**: We tag entities *Person* (*Per*), *Organization* (*Org*) using Stanford NER tagger (Finkel, Grenager, and Manning 2005). The entity type *Position* (*Post*) is annotated based on the templates. **Relation Tagging**: We have three types of relations: *Per-Org*, *Per-Post* and *Post-Org*. We follow Swampillai and Stevenson (2010) and annotate binary relations (within and across sentence boundaries) using management succession events between two entity pairs. We randomly split the collection 60/20/20 into train/dev/test.

Experimental Setup. For MUC6, we use the pretrained GloVe (Pennington, Socher, and Manning 2014) embeddings (200-dimension). For the BioNLP datasets, we use 200-dimensional embedding[2] vectors from six billion words of biomedical text (Moen and Ananiadou 2013). We randomly initialize a 5-dimensional vectors for PI and POS. We initialize the recurrent weight matrix to identity and biases to zero. We use the macro-averaged F_1 score (the official evaluation script by SemEval-2010 Task 8 (Hendrickx et al. 2010)) on the development set to choose hyperparameters (see supplementary). To report results on BioNLP ST 2016 test set, we use the official web service[3].

[1] the official evaluation is not accessible any more and therefore, the annotations for their test sets are not available

Dataset: BioNLP ST 2016						
Features	**SVM**			**iDepNN**		
	P	R	F_1	P	R	F_1
iSDP	.217	.816	.344	.352	.574	.436
+ PI + ET	.218	.819	.344	.340	.593	.432
+ POS	.269	.749	.396	.348	.568	.431
+ Dependency	.284	.746	.411	.402	.509	**.449**
Dataset: MUC6						
Features	**SVM**			**iDepNN**		
	P	R	F_1	P	R	F_1
iSDP	.689	.630	.627	.916	.912	.913
+ PI	.799	.741	.725	.912	.909	.909
+ POS	.794	.765	.761	.928	.926	.926
+ Dependency	.808	.768	.764	.937	.934	**.935**

Table 2: SVM vs iDepNN: Features in inter-sentential ($k\leqslant 1$) training and inter-sentential ($k \leqslant 1$) evaluation. iSDP+Dependency refers to iDepNN-ADP structure.

Baselines. Swampillai and Stevenson's (2010) annotation of MUC6 intra- and inter-sentential relationships is not available. They investigated SVM with dependency and linguistic features for relationships spanning sentence boundaries. In BioNLP shared tasks, the top performing systems are SVM-based and limited to relationships within single sentences. As an NN baseline, we also develop Connectionist biRNN (Vu et al. 2016a) that spans sentence boundaries; we refer to it as i-biRNN (architecture in supplementary). Similarly, we also investigate using a bidirectional LSTM (i-biLSTM). As a competitive baseline in the inter-sentential relationship extraction, we run[4] graphLSTM (Peng et al. 2017). This work compares SVM and graphLSTM with i-biRNN, i-biLSTM, iDepNN-SDP and iDepNN-ADP for different values of the sentence range parameter k (the distance in terms of the number of sentences between the entity pairs for a relation) , i.e., k ($= 0, \leqslant 1, \leqslant 2$ and $\leqslant 3$).

Contribution of different components. Table 2 shows the contribution of each feature, where both training and evaluation is performed over relationships within and across sentence boundaries for sentence range parameter $k\leqslant 1$. Note: iSDP+Dependency refers to iDepNN-ADP structure

[2] http://bio.nlplab.org/

[3] http://bibliome.jouy.inra.fr/demo/BioNLP-ST-2016-Evaluation/index.html

[4] hyperparameters in supplementary

train param	Model	Evaluation for different values of sentence range k															
		$k=0$				$k \leqslant 1$				$k \leqslant 2$				$k \leqslant 3$			
		pr	P	R	F_1	**pr**	P	R	F_1	**pr**	P	R	F_1	**pr**	P	R	F_1
$k=0$	SVM	363	.474	.512	.492	821	.249	.606	.354	1212	.199	.678	.296	1517	.153	.684	.250
	graphLSTM	473	.472	.668	.554	993	.213	.632	.319	1345	.166	.660	.266	2191	.121	.814	.218
	i-biLSTM	480	.475	.674	.556	998	.220	.652	.328	1376	.165	.668	.265	1637	.132	.640	.219
	i-biRNN	286	.517	.437	.474	425	.301	.378	.335	540	.249	.398	.307	570	.239	.401	.299
	iDepNN-SDP	297	.519	.457	.486	553	.313	.510	.388	729	.240	.518	.328	832	.209	.516	.298
	iDepNN-ADP	266	.526	.414	.467	476	.311	.438	.364	607	.251	.447	.320	669	.226	.447	.300
$k \leqslant 1$	SVM	471	.464	.645	.540	888	.284	.746	.411	1109	.238	.779	.365	1196	.221	.779	.344
	graphLSTM	406	.502	.607	.548	974	.226	.657	.336	1503	.165	.732	.268	2177	.126	.813	.218
	i-biLSTM	417	.505	.628	.556	1101	.224	.730	.343	1690	.162	.818	.273	1969	.132	.772	.226
	i-biRNN	376	.489	.544	.515	405	.393	.469	.427	406	.391	.469	.426	433	.369	.472	.414
	iDepNN-SDP	303	.561	.503	.531	525	.358	.555	.435	660	.292	.569	.387	724	.265	.568	.362
	iDepNN-ADP	292	**.570**	.491	.527	428	**.402**	.509	**.449**	497	**.356**	.522	**.423**	517	**.341**	.521	**.412**
$k \leqslant 2$	SVM	495	.461	.675	.547	1016	.259	.780	.389	1296	.218	.834	.345	1418	.199	.834	.321
	graphLSTM	442	.485	.637	.551	1016	.232	.702	.347	1334	.182	.723	.292	1758	.136	.717	.230
	i-biLSTM	404	.487	.582	.531	940	.245	.682	.360	1205	.185	.661	.289	2146	.128	.816	.222
	i-biRNN	288	.566	.482	.521	462	.376	.515	.435	556	.318	.524	.396	601	..296	.525	.378
	iDepNN-SDP	335	.537	.531	.534	633	.319	.598	.416	832	.258	.634	.367	941	.228	.633	.335
	iDepNN-ADP	309	.538	.493	.514	485	.365	.525	.431	572	.320	.542	.402	603	.302	.540	.387
$k \leqslant 3$	SVM	507	.458	**.686**	.549	1172	.234	**.811**	.363	1629	.186	**.894**	.308	1874	.162	**.897**	.275
	graphLSTM	429	.491	.624	.550	1082	.230	.740	.351	1673	.167	.833	.280	2126	.124	.787	.214
	i-biLSTM	417	.478	.582	.526	1142	.224	.758	.345	1218	.162	.833	.273	2091	.128	.800	.223
	i-biRNN	405	.464	.559	.507	622	.324	.601	.422	654	.310	.604	.410	655	.311	.607	.410
	iDepNN-SDP	351	.533	.552	.542	651	.315	.605	.414	842	.251	.622	.357	928	.227	.622	.333
	iDepNN-ADP	313	.553	.512	.532	541	.355	.568	.437	654	.315	.601	.415	687	.300	.601	.401
$\mathbf{k \leqslant 1}$	*ensemble*	480	.478	.680	**.561**	837	.311	.769	.443	1003	.268	.794	.401	1074	.252	.797	.382

Table 3: BioNLP ST 2016 Dataset: Performance of the intra-and-inter-sentential training/evaluation for different k. Underline: Better precision by *iDepNN-ADP* over *iDepNN-SDP*, graphLSTM and SVM. **Bold**: Best in column. **pr**: Count of predictions

that exhibits a better precision, F_1 and balance in precision and recall, compared to SVM. See supplementary for feature analysis on BioNLP ST 2011 / 2013.

State-of-the-Art Comparisons

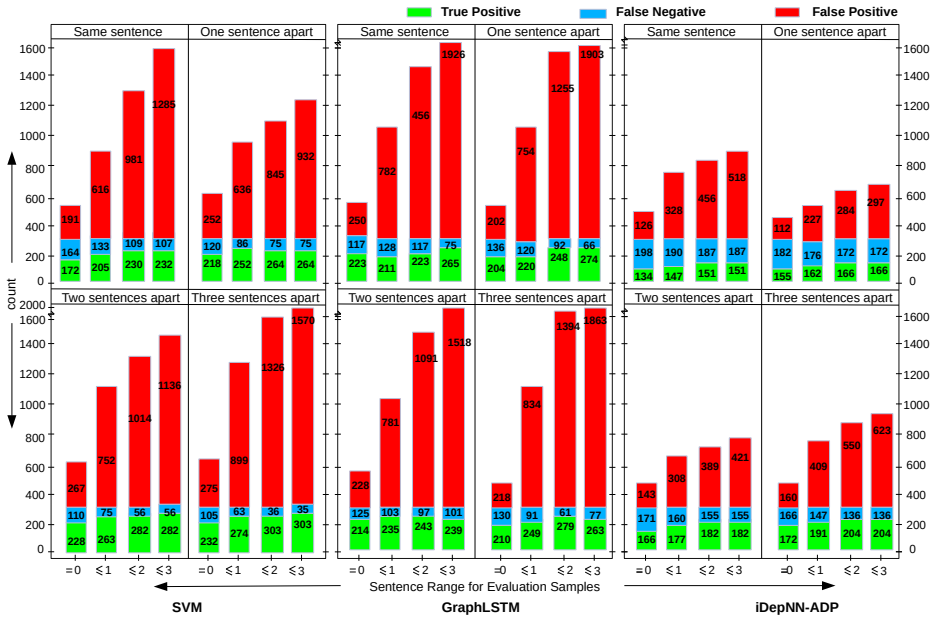
BioNLP ST 2016 dataset: Table 3 shows the performance of {SVM, graphLSTM} vs {i-biRNN, iDepNN-SDP, iDepNN-ADP} for relationships within and across sentence boundaries. *Moving left to right* for *each* training parameter, the recall increases while precision decreases due to increasing noise with larger k. In the inter-sentential evaluations ($k \leqslant 1, \leqslant 2, \leqslant 3$ columns) for *all* the training parameters, the iDepNN variants outperform both SVM and graphLSTM in terms of F_1 and maintain a better precision as well as balance in precision and recall with increasing k; e.g., at $k \leqslant 1$ (train/eval), precision and F_1 are (.402 vs .226) and (.449 vs .336), respectively for (iDepNN-ADP vs graphLSTM). We find that SVM mostly leads in recall.

In comparison to graphLSTM, i-biRNN and i-biLSTM, we observe that iDepNN-ADP offers precise structure in relation extraction within and across sentence boundaries. For instance, at training $k \leqslant 1$ and evaluation $k = 0$, iDepNN-ADP reports precision of .570 vs .489 and .561 in i-biRNN and iDepNN-SDP, respectively. During training at $k \leqslant 1$, iDepNN-SDP and iDepNN-ADP report better F_1 than i-biRNN for evaluations at all k, suggesting that the shortest and augmented paths provide useful dependency features via recurrent and recursive compositions, respectively. Between the proposed architectures iDepNN-SDP and iDepNN-ADP, the former achieves higher recall for all k. We find that the training at $k \leqslant 1$ is optimal for intra- and inter-sentential relations over development set (see supplementary). We also observe that i-biRNN establishes a strong NN baseline for relationships spanning sentences. The proposed architectures consistently outperform graphLSTM in both precision and F_1 across sentence boundaries.

threshold	*ensemble* (train $k \leqslant 1$ and evaluation $k = 0$)							
	Dev (official scores)				*Test* (official scores)			
	pr	P	R	F_1	**pr**	P	R	F_1
$p \geqslant 0.85$	160	.694	.514	.591	419	.530	.657	**.587**
$p \geqslant 0.90$	151	.709	.496	.583	395	.539	.630	.581
$p \geqslant 0.95$	123	.740	.419	.535	293	.573	.497	.533

Table 4: Ensemble scores at various thresholds for BioNLP ST 2016 dataset. p: output probability

Ensemble: We exploit the precision and recall bias of the different models via an ensemble approach, similar to TurkuNLP (Mehryary et al. 2016) and UMS (Delėger et al. 2016) systems that combined predictions from SVM and NNs. We aggregate the prediction outputs of the neural (i-



Teams	Lives_In			
	F_1	R	P	pr
this work	**.587**	**.657**	.530	419
VERSE	.558	.615	.510	408
TurkuNLP	.522	.448	.626	243
LIMSI	.485	.646	.388	559
HK	.474	.392	.599	222
WhuNlpRE	.471	.407	.559	247
UMS	.463	.399	.551	245
DUTIR	.456	.382	.566	228
WXU	.455	.383	.560	232

Figure 3: Left: SVM, graphLSTM & iDepNN-ADP on BioNLP ST 2016: Performance analysis on relations that span sentence boundaries, with different sentence range parameters Right: BioNLP 2016 ST dataset (official results on test set): Comparison with the published systems in the BioNLP ST, where *pr* is the count of predictions. This work demonstrates a better balance in precision and recall, and achieves the highest F_1 and *recall*. We extract 419 predictions within and across sentence boundaries, which is closer to the count of gold predictions, i.e., 340 (Delèger et al. 2016).

biRNN, iDepNN-SDP and iDepNN-ADP) and non-neural (SVM) classifiers, i.e., a relation to hold if any classifier has predicted it. We perform the *ensemble* scheme on the development and official test sets for intra- and inter-sentential (optimal at $k \leqslant 1$) relations. Table 3 shows the ensemble scores on the official test set for relations within and across sentence boundaries, where *ensemble* achieves the highest F_1 (.561) over individual models.

Confident Extractions: We consider the high confidence prediction outputs by the different models participating in the *ensemble*, since it lacks precision (.478). Following Peng et al. (2017), we examine three values of the output probability p, i.e., ($\geqslant 0.85, 0.90$ and 0.95) of each model in the *ensemble*. Table 4 shows the *ensemble* performance on the development and official test sets, where the predictions with $p \geqslant 0.85$ achieve the state-of-the-art performance and rank us at the top out of 11 systems (Figure 3, right).

This Work vs Competing Systems in BioNLP ST 2016: As shown in Figure 3 (right), we rank at the top and achieve a gain of 5.2% (.587 vs .558) in F_1 compared to VERSE. We also show a better balance in precision and recall, and report the highest recall compared to all other systems. Most systems do not attempt to predict relations spanning sentences. The most popular algorithms are SVM (VERSE, HK, UTS, LIMSI) and NNs (TurkuNLP, WhuNlpRE, DUTIR). UMS combined predictions from an SVM and an NN. Most systems rely on syntactic parsing, POS, word embeddings and entity recognition features (VERSE, TurkuNLP, UMS, HK, DUTIR, UTS). VERSE and TurkuNLP obtained top scores on intra-sentential relations relying on the dependency path features between entities; however they are limited to intra-sentential relations. TurkuNLP employed an ensemble of 15 different LSTM based classifiers. DUTIR is based on CNN for intra-sentential relationsips. LIMSI is the only system that considers inter-sentential relationships during training; however it is SVM-based and used additional manually annotated training data, linguistic features using biomedical resources (PubMed, Cocoa web API, OntoBiotope ontology, etc.) and post-processing to annotate biomedical abbreviations. We report a noticeable gain of 21% (.587 vs .485) in F_1 with an improved precision and recall over LIMSI.

BioNLP ST 2011 and 2013 datasets: Following the BioNLP ST 2016 evaluation, we also examine two additional datasets from the same domain. iDepNN-ADP (Table 5) is the leading performer in terms of precision and F_1 within and across boundaries for BioNLP ST 2013. Examining BioNLP ST 2011, the iDepNN variants lead both SVM and i-biRNN for $k \leqslant 1$ and $k \leqslant 2$.

MUC6 dataset: Similar to BioNLP ST 2016, we perform training and evaluation of SVM, i-biRNN, iDepNN-SDP and iDepNN-ADP for different sentence range with best feature combination (Table 2) using MUC6 dataset. Table 6 shows that both iDepNN variants consistently outperform graphLSTM and SVM for relationships within and across sentences. For within (k=0) sentence evaluation, iDepNN-ADP reports .963 F_1, compared to .779 and .783 by SVM and graphLSTM, respectively. iDepNN-ADP is observed more precise than iDepNN-SDP and graphLSTM with in-

Model	*Dataset: BioNLP ST 2013*												*Dataset: BioNLP ST 2011*											
	$k=0$			$k\leqslant 1$			$k\leqslant 2$			$k\leqslant 3$			$k=0$			$k\leqslant 1$			$k\leqslant 2$			$k\leqslant 3$		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
SVM	.95	.90	.92	.87	.83	.85	.95	.90	.92	.95	.90	.92	.98	.96	.97	.87	.87	.87	.95	.94	.94	.91	.88	.90
graphLSTM	.98	.97	.97	.94	**.95**	.94	.95	.89	.92	.90	**.97**	**.93**	**.99**	.99	**.99**	.95	**.98**	**.96**	.95	.97	.96	.96	.92	.93
i-biLSTM	.98	.97	.97	.96	**.95**	**.95**	.93	**.96**	.94	.91	.93	.92	**.99**	.99	**.99**	.95	**.98**	**.96**	.96	.97	.96	.95	.92	.93
i-biRNN	.95	.94	.94	.93	.90	.91	.94	.92	.93	.94	.84	.89	.97	.99	.98	.88	.94	.90	.92	.94	.93	**.96**	**.96**	**.96**
iDepNN-SDP	.94	.96	.95	.94	**.95**	.94	.87	.92	.89	.91	.94	.92	.97	**.99**	.98	**.96**	.92	.93	**.97**	.97	**.97**	.94	.91	.92
iDepNN-ADP	**.99**	**.98**	**.99**	**.97**	.94	**.95**	**.98**	.95	**.96**	**.96**	.91	**.93**	.97	.97	.97	.93	.96	94	.92	**.98**	.95	.93	.94	.93

Table 5: BioNLP ST 2011 and 2013 datasets: Performance for training ($k \leqslant 1$) and evaluation for different k. Underline: Better precision in iDepNN-ADP than iDepNN-SDP, graphLSTM, i-biLSTM, i-biRNN and SVM. **Bold**: best in column.

train param	Model	Evaluation for different k								
		$k=0$			$k\leqslant 1$			$k\leqslant 3$		
		P	R	F_1	P	R	F_1	P	R	F_1
$k=0$	SVM	.796	.765	.760	.775	.762	.759	.791	.779	.776
	graphLSTM	.910	.857	.880	.867	.897	.870	.870	.867	.870
	i-biLSTM	.917	.837	.873	.833	.896	.863	.853	.87 0	.863
	i-biRNN	.875	.859	.864	.828	.822	.824	.830	.827	.827
	iDepNN-SDP	.958	.948	.952	.934	.928	.930	.935	.930	.932
	iDepNN-ADP	.933	.927	.929	.924	.920	.921	.930	.927	.927
$k\leqslant 1$	SVM	.815	.772	.769	.808	.768	.764	.802	.775	.770
	graphLSTM	.730	.900	.783	.727	.907	.773	.730	.913	.770
	i-biLSTM	.760	.880	.780	.670	.950	.767	.697	.937	.770
	i-biRNN	.925	.934	.927	.870	.872	.860	.868	.866	.858
	iDepNN-SDP	.949	.945	.946	.928	.926	.926	.934	.932	.932
	iDepNN-ADP	.961	.955	.957	**.937**	**.934**	**.935**	**.942**	**.940**	**.940**
$k\leqslant 3$	SVM	.840	.785	.779	.816	.779	.774	.822	.788	.781
	graphLSTM	.737	.907	.783	.703	.927	.773	.710	.927	.767
	i-biLSTM	.720	.920	.780	.680	.943	.770	.700	.932	.770
	i-biRNN	.944	.934	.938	.902	.890	.895	.926	.923	.924
	iDepNN-SDP	.956	.947	.951	.920	.916	.917	.939	.938	.936
	iDepNN-ADP	**.965**	**.963**	**.963**	.933	.933	.931	.939	.938	.936

Table 6: MUC6 Dataset: Performance over the intra- and inter-sentential training and evaluation for different k. Underline signifies better precision by iDepNN-ADP over iDepNN-SDP, graphLSTM, i-biLSTM, i-biRNN and SVM. **Bold** indicates the best score column-wise.

creasing k, e.g., at $k\leqslant 3$. Training at sentence range $k\leqslant 1$ is found optimal in extracting inter-sentential relationships.

Error Analysis and Discussion

In Figure 3 (left), we analyze predictions using different values of sentence range k (=0, $\leqslant 1$, $\leqslant 2$ and $\leqslant 3$) during both training and evaluation of SVM, graphLSTM and iDepNN-ADP for BioNLP ST 2016 dataset. For instance, an SVM (top-left) is trained for intra-sentential (*same sentence*) relations, while iDepNN-ADP (bottom-right) for both intra- and inter-sentential spanning three sentences (*three sentences apart*). We show how the count of true positives (TP), false negatives (FN) and false positives (FP) varies with k.

Observe that as the distance of the relation increases, the classifiers predict larger ratios of false positives to true positives. However, as the sentence range increases, iDepNN-ADP outperforms both SVM and graphLSTM due to fewer false positives (red colored bars). On top, the ratio of FP to TP is better in iDepNN-ADP than graphLSTM and SVM for all values of k. Correspondingly in Table 3, iDepNN-ADP reports better precision and balance between precision and recall, signifying its robustness to noise in handling inter-sentential relationships.

iDepNN vs graphLSTM: Peng et al. (2017) focuses on general relation extraction framework using graphLSTM with challenges such as potential cycles in the document graph leading to expensive model training and difficulties in convergence due to loopy gradient backpropagation. Therefore, they further investigated different strategies to back-propagate gradients. The graphLSTM introduces a number of parameters with a number of edge types and thus, requires abundant supervision/training data. On other hand, our work introduces simple and robust neural architectures (iDepNN-SDP and iDepNN-ADP), where the iDepNN-ADP is a special case of document graph in form of a parse tree spanning sentence boundaries. We offer a smooth gradient backpropagation in the complete structure (e.g., in iDepNN-ADP via recurrent and recursive hidden vectors) that is more efficient than graphLSTM due to non-cyclic (i.e., tree) architecture. We have also shown that iDepNN-ADP is robust to false positives and maintains a better balance in precision and recall than graphLSTM for inter-sentential relationships (Figure 3).

Conclusion

We have proposed to classify relations between entities within and across sentence boundaries by modeling the inter-sentential shortest and augmented dependency paths within a novel neural network, named as inter-sentential Dependency-based Neural Network (iDepNN) that takes advantage of both recurrent and recursive neural networks to model the structures in the intra- and inter-sentential relationships. Experimental results on four datasets from newswire and medical domains have demonstrated that iDepNN is robust to false positives, shows a better balance in precision and recall and achieves the state-of-the-art performance in extracting relationships within and across sentence boundaries. We also perform better than 11 teams participating in the BioNLP shared task 2016.

References

Bahcall, O. 2015. Precision medicine. *Nature* 526:335.

Bossy, R.; Jourde, J.; Bessieres, P.; Van De Guchte, M.; and Nédellec, C. 2011. Bionlp shared task 2011: bacteria biotope. In *Proceedings of the BioNLP Shared Task 2011 Workshop*, 56–64. Association for Computational Linguistics.

Bunescu, R. C., and Mooney, R. J. 2005. A shortest path dependency kernel for relation extraction. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*, 724–731. Association for Computational Linguistics.

Delèger, L.; Bossy, R.; Chaix, E.; Ba, M.; Ferrè, A.; Bessieres, P.; and Nèdellec, C. 2016. Overview of the bacteria biotope task at bionlp shared task 2016. In *Proceedings of the 4th BioNLP Shared Task Workshop*, 12–22.

Finkel, J. R.; Grenager, T.; and Manning, C. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd annual meeting on association for computational linguistics*, 363–370. Association for Computational Linguistics.

Gerber, M., and Chai, J. Y. 2010. Beyond nombank: A study of implicit arguments for nominal predicates. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, 1583–1592. Association for Computational Linguistics.

Grishman, R., and Sundheim, B. 1996. Message understanding conference-6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, volume 1.

Grouin, C. 2016. Identification of mentions and relations between bacteria and biotope from pubmed abstracts. *ACL 2016* 64.

Gupta, P.; Schütze, H.; and Andrassy, B. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2537–2547.

Gupta, P. 2015. Deep Learning Methods for the Extraction of Relations in Natural Language Text. Master's thesis, Technical University of Munich, Germany.

Hagberg, A.; Swart, P.; and S Chult, D. 2008. Exploring network structure, dynamics, and function using networkx. Technical report, Los Alamos National Laboratory (LANL).

Hendrickx, I.; Kim, S. N.; Kozareva, Z.; Nakov, P.; Séaghdha, D. O.; Padó, S.; Pennacchiotti, M.; Romano, L.; and Szpakowicz, S. 2010. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. *ACL 2010* 33.

Kambhatla, N. 2004. Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations. In *Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, 22. Association for Computational Linguistics.

Liu, Y.; Wei, F.; Li, S.; Ji, H.; Zhou, M.; and Wang, H. 2015. A dependency-based neural network for relation classification. *arXiv preprint arXiv:1507.04646*.

Manning, C. D.; Surdeanu, M.; Bauer, J.; Finkel, J. R.; Bethard, S.; and McClosky, D. 2014. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, 55–60.

Mehryary, F.; Björne, J.; Pyysalo, S.; Salakoski, T.; and Ginter, F. 2016. Deep learning with minimal training data: Turkunlp entry in the bionlp shared task 2016. In *Proceedings of the 4th BioNLP Shared Task Workshop*, 73–81.

Mintz, M.; Bills, S.; Snow, R.; and Jurafsky, D. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, 1003–1011. Association for Computational Linguistics.

Moen, S., and Ananiadou, T. S. S. 2013. Distributional semantics resources for biomedical text processing.

Nédellec, C.; Bossy, R.; Kim, J.-D.; Kim, J.-J.; Ohta, T.; Pyysalo, S.; and Zweigenbaum, P. 2013. Overview of bionlp shared task 2013. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, 1–7. Association for Computational Linguistics Sofia, Bulgaria.

Nguyen, D. P.; Matsuo, Y.; and Ishizuka, M. 2007. Relation extraction from wikipedia using subtree mining. In *Proceedings of the National Conference on Artificial Intelligence*, volume 22, 1414. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

Peng, N.; Poon, H.; Quirk, C.; Toutanova, K.; and Yih, W.-t. 2017. Cross-sentence n-ary relation extraction with graph lstms. *Transactions of the Association for Computational Linguistics* 5:101–115.

Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Quirk, C., and Poon, H. 2016. Distant supervision for relation extraction beyond the sentence boundary. *arXiv preprint arXiv:1609.04873*.

Schuster, M., and Paliwal, K. K. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45(11):2673–2681.

Socher, R.; Huval, B.; Manning, C. D.; and Ng, A. Y. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, 1201–1211. Association for Computational Linguistics.

Socher, R.; Karpathy, A.; Le, Q. V.; Manning, C. D.; and Ng, A. Y. 2014. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics* 2:207–218.

Swampillai, K., and Stevenson, M. 2010. Inter-sentential relations in information extraction corpora. In *LREC*.

Swampillai, K., and Stevenson, M. 2011. Extracting relations within and across sentences. In *RANLP*, 25–32.

Vu, N. T.; Adel, H.; Gupta, P.; and Schütze, H. 2016a. Combining recurrent and convolutional neural networks for relation classification. In *Proceedings of the NAACL-HLT*, 534–539. San Diego, California USA: Association for Computational Linguistics.

Vu, N. T.; Gupta, P.; Adel, H.; and Schütze, H. 2016b. Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *Proceedings of the Acoustics, Speech and Signal Processing (ICASSP)*, 6060–6064. Shanghai, China: IEEE.

Werbos, P. J. 1990. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE* 78(10):1550–1560.

Yoshikawa, K.; Riedel, S.; Hirao, T.; Asahara, M.; and Matsumoto, Y. 2011. Coreference based event-argument relation extraction on biomedical text. *Journal of Biomedical Semantics* 2(5):S6.

Zhang, M.; Zhang, J.; Su, J.; and Zhou, G. 2006. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, 825–832. Association for Computational Linguistics.

Chapter 5

Deep Temporal-Recurrent-Replicated-Softmax for Topical Trends over Time

Deep Temporal-Recurrent-Replicated-Softmax for Topical Trends over Time

Pankaj Gupta[1,2], Subburam Rajaram[1], Hinrich Schütze[2], Bernt Andrassy[1]
[1]Corporate Technology, Machine-Intelligence (MIC-DE), Siemens AG Munich, Germany
[2]CIS, University of Munich (LMU) Munich, Germany
{pankaj.gupta, subburam.rajaram, bernt.andrassy}@siemens.com
pankaj.gupta@campus.lmu.de | inquiries@cislmu.org

Abstract

Dynamic topic modeling facilitates the identification of topical trends over time in temporal collections of unstructured documents. We introduce a novel unsupervised neural dynamic topic model named as Recurrent Neural Network-Replicated Softmax Model (RNN-RSM), where the discovered topics at each time influence the topic discovery in the subsequent time steps. We account for the temporal ordering of documents by explicitly modeling a joint distribution of latent topical dependencies over time, using distributional estimators with temporal recurrent connections. Applying RNN-RSM to 19 years of articles on NLP research, we demonstrate that compared to state-of-the art topic models, RNN-RSM shows better generalization, topic interpretation, evolution and trends. We also introduce a metric (named as SPAN) to quantify the capability of dynamic topic model to capture word evolution in topics over time.

1 Introduction

Topic Detection and Tracking (Allan et al., 1998) is an important area of natural language processing to find topically related ideas that evolve over time in a sequence of text collections and exhibit temporal relationships. The temporal aspects of these collections can present valuable insight into the topical structure of the collections and can be quantified by modeling the dynamics of the underlying topics discovered over time.

Problem Statement: We aim to generate temporal topical trends or automatic overview timelines of topics for a time sequence collection of documents. This involves the following three tasks in dynamic topic analysis: **(1)** *Topic Structure Detection* (TSD): Identifying main topics in the document collection. **(2)** *Topic Evolution Detection* (TED): Detecting the emergence of a new topic and recognizing how it grows or decays over time (Allan, 2002). **(3)** *Temporal Topic Characterization* (TTC): Identifying the characteristics for each of the main topics in order to track the words' usage (*keyword trends*) for a topic over time i.e. *topical trend analysis for word evolution* (Fig 1, Left).

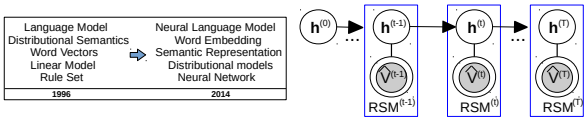


Figure 1: (Left): Word Usage over time for Topic (*Word Representation*) in scholarly articles. (Right): RSM-based dynamic topic model with explicit temporal topic dependence

Probabilistic static topic models, such as Latent Dirichlet Allocation (LDA) (Blei et al., 2003) and its variants (Wang and McCallum, 2006; Hall et al., 2008; Gollapalli and Li, 2015) have been investigated to examine the emergence of topics from historical documents. Another variant known as Replicated Softmax (RSM) (Hinton and Salakhutdinov, 2009) has demonstrated better generalization in log-probability and retrieval, compared to LDA. Prior works (Iwata et al., 2010; Pruteanu-Malinici et al., 2010; Saha and Sindhwani, 2012; Schein et al., 2016) have investigated Bayesian modeling of topics in time-stamped documents. Particularly, Blei and Lafferty (2006) developed a LDA based dynamic topic model (DTM) to capture the evolution of topics in a time sequence collection of documents; however they do not capture explicitly the topic popularity and usage of specific terms over time. We propose a family of probabilistic time series models with distributional estimators to explicitly model the dynamics of the underlying topics, introducing temporal latent topic dependencies (Fig 1, Right).

To model temporal dependencies in high dimen-

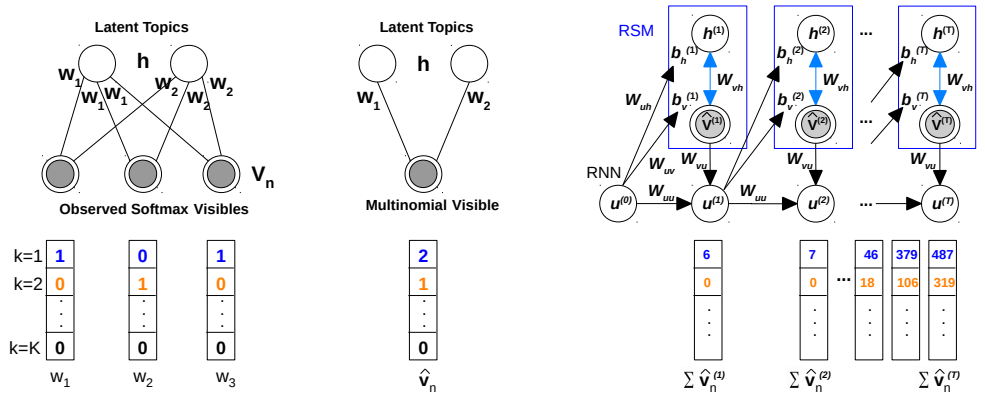


Figure 2: (Left): RSM for a document $\mathbf{V}_n$ of D_n=3 words (w). The bottom layer represents the softmax visible units, that share the same set of weights connected to binary hidden units $\mathbf{h}$. (Middle): Interpretation of RSM in which D_n softmax units with identical weights are replaced by a single multinomial unit, sampled D_n times. (Right): Graphical structure of 2-layered **RNN-RSM**, unfolded in time. Single and double headed arrows represent deterministic and stochastic-symmetric connections, respectively. $\widehat{\mathbf{V}}^{(t)}$ and $\mathbf{h}^{(t)}$ are binary visible and hidden layers of RSM for a document collection at time, t. $\mathbf{u}$ is RNN hidden layer. k: dictionary index for a word w

sional sequences, such as polyphonic music, the temporal stack of RBMs (Smolensky, 1986; Hinton, 2002) has been investigated to model complex distributions. The Temporal RBM (Taylor et al., 2007; Sutskever and Hinton, 2007), Recurrent Temporal RBM (RTRBM) (Sutskever et al., 2009) and RNN-RBM (Boulanger-Lewandowski et al., 2012) show success in modeling the temporal dependencies in such symbolic sequences. In addition, RNNs (Gupta et al., 2015a; Vu et al., 2016a,b; Gupta et al., 2016) have been recognized for sentence modeling in natural language tasks. We aspire to build neural dynamic topic model called RNN-RSM to model document collections over time and learn temporal topic correlations.

We consider RSM for TSD and introduce the explicit latent topical dependencies for TED and TTC tasks. Fig 1 illustrates our *motivation*, where temporal ordering in document collection $\widehat{\mathbf{V}}^{(t)}$ at each time step t, is modeled by conditioning the latent topic $\mathbf{h}^{(t)}$ on the sequence history of latent topics $\mathbf{h}^{(0)}$, ..., $\mathbf{h}^{(t-1)}$, accumulated with temporal lag. Each RSM discovers latent topics, where the introduction of a bias term in each RSM via the time-feedback latent topic dependencies enables to explicitly model topic evolution and specific topic term usage over time. The temporal connections and RSM biases allow to convey topical information and model relation among the words, in order to deeply analyze the dynamics of the underlying topics. We demonstrate the applicability of proposed **RNN-RSM** by analyzing 19 years of scientific articles from NLP research.

The *contributions* in this work are:
(1) Introduce an unsupervised neural dynamic topic model based on recurrent neural network and RSMs, named as RNN-RSM to explicitly model discovered latent topics (evolution) and word relations (topic characterization) over time.
(2) Demonstrate better generalization (log-probability and time stamp prediction), topic interpretation (coherence), evolution and characterization, compared to the state-of-the-art.
(3) It is the first work in dynamic topic modeling using undirected stochastic graphical models and deterministic recurrent neural network to model collections of different-sized documents over time, within the generative and neural network framework. The code and data are available at `https://github.com/pgcool/RNN-RSM`.

2 The RNN-RSM model

RSM (Fig 2, Left) models are a family of different-sized Restricted Boltzmann Machines (RBMs) (Gehler et al., 2006; Xing et al., 2005; Gupta et al., 2015b,c) that models *word counts* by sharing the same parameters with multinomial distribution over the observable i.e. it can be interpreted as a single multinomial unit (Fig 2, Middle) sampled as many times as the document size. This facilitates in dealing with the documents of different lengths.

The proposed RNN-RSM model (Fig 2, Right) is a sequence of conditional RSMs[1] such that at any time step t, the RSM's bias parameters $\mathbf{b_v}^{(t)}$

[1]Notations: $\widehat{\mathbf{U}}=\{\mathbf{U}_n\}_{n=1}^{N}$; **U**:2D-Matrix; **l**:vector; **U/l**:Upper/lower-case; Scalars in unbold

and $\mathbf{b_h}^{(t)}$ depend on the output of a deterministic RNN with hidden layer $\mathbf{u}^{(t-1)}$ in the previous time step, $t-1$. Similar to RNN-RBM (Boulanger-Lewandowski et al., 2012), we constrain RNN hidden units ($\mathbf{u}^{(t)}$) to convey temporal information, while RSM hidden units ($\mathbf{h}^{(t)}$) to model conditional distributions. Therefore, parameters ($\mathbf{b_v}^{(t)}$, $\mathbf{b_h}^{(t)}$) are time-dependent on the sequence history at time t (via a series of conditional RSMs) denoted by $\Theta^{(t)} \equiv \{\widehat{\mathbf{V}}^{(\tau)}, \mathbf{u}^{(\tau)} | \tau < t\}$, that captures temporal dependencies. The RNN-RSM is defined by its joint probability distribution:

$$P(\widehat{\mathfrak{V}}, \mathbf{H}) = P(\{\widehat{\mathbf{V}}^{(t)}, \mathbf{h}^{(t)}\}_{t=1}^{T}) = \prod_{t=1}^{T} P(\widehat{\mathbf{V}}^{(t)}, \mathbf{h}^{(t)} | \Theta^{(t)})$$

where $\widehat{\mathfrak{V}} = [\widehat{\mathbf{V}}^{(1)}, ...\widehat{\mathbf{V}}^{(T)}]$ and $\mathbf{H} = [\mathbf{h}^{(1)}, ...\mathbf{h}^{(T)}]$. Each $\mathbf{h}^{(t)} \in \{0,1\}^F$ be a binary stochastic hidden topic vector with size F and $\widehat{\mathbf{V}}^{(t)} = \{\mathbf{V}_n^{(t)}\}_{n=1}^{N^{(t)}}$ be a collection of N documents at time step t. Let $\mathbf{V}_n^{(t)}$ be a $K \times D_n^{(t)}$ observed binary matrix of the n^{th} document in the collection where, $D_n^{(t)}$ is the document size and K is the dictionary size over all the time steps. The conditional distribution (for each unit in hidden or visible) in each RSM at time step, is given by softmax and logistic functions:

$$P(v_{n,i}^{k,(t)} = 1 | \mathbf{h}_n^{(t)}) = \frac{\exp(b_{v,i}{}^{k,(t)} + \sum_{j=1}^{F} h_{n,j}^{(t)} W_{ij}^{k})}{\sum_{q=1}^{K} \exp(b_{v,i}{}^{q,(t)} + \sum_{j=1}^{F} h_{n,j}^{(t)} W_{ij}^{q})}$$

$$P(h_{n,j}^{(t)} = 1 | \mathbf{V}_n^{(t)}) = \sigma(b_{h,j}^{(t)} + \sum_{i=1}^{D_n^{(t)}} \sum_{k=1}^{K} v_{n,i}^{k,(t)} W_{ij}^{k})$$

where $P(v_{n,i}^{k,(t)} = 1 | \mathbf{h}_n^{(t)})$ and $P(h_{n,j}^{(t)} = 1 | \mathbf{V}_n^{(t)})$ are conditional distributions for i^{th} visible $v_{n,i}$ and j^{th} hidden unit $h_{n,j}$ for the n^{th} document at t. W_{ij}^k is a symmetric interaction term between i that takes on value k and j. $v_n^{k,(t)}$ is sampled $D_n^{(t)}$ times with identical weights connected to binary hidden units, resulting in multinomial visibles, therefore the name *Replicated Softmax*. The conditionals across layers are factorized as: $P(\mathbf{V}_n^{(t)} | \mathbf{h}_n^{(t)}) = \prod_{i=1}^{D_n^{(t)}} P(\mathbf{v}_{n,i}^{(t)} | \mathbf{h}_n^{(t)})$; $P(\mathbf{h}_n^{(t)} | \mathbf{V}_n^{(t)}) = \prod_j P(h_{n,j}^{(t)} | \mathbf{V}_n^{(t)})$.

Since biases of RSM depend on the output of RNN at previous time steps, that allows to propagate the estimated gradient at each RSM backward through time (BPTT). The *RSM biases* and RNN hidden state $\mathbf{u}^{(t)}$ at each time step t are given by-

$$\begin{aligned} \mathbf{b_v}^{(t)} &= \mathbf{b_v} + \mathbf{W_{uv}} \mathbf{u}^{(t-1)} \\ \mathbf{b_h}^{(t)} &= \mathbf{b_h} + \mathbf{W_{uh}} \mathbf{u}^{(t-1)} \end{aligned} \quad (1)$$

$$\mathbf{u}^{(t)} = \tanh(\mathbf{b_u} + \mathbf{W_{uu}} \mathbf{u}^{(t-1)} + \mathbf{W_{vu}} \sum_{n=1}^{N^{(t)}} \hat{\mathbf{v}}_n^{(t)}) \quad (2)$$

Algorithm 1 *Training RNN-RSM with BPTT*

Input: *Observed visibles*, $\widehat{\mathfrak{V}} = \{\widehat{\mathbf{V}}^{(0)}, \widehat{\mathbf{V}}^{(1)}, ..., \widehat{\mathbf{V}}^{(t)}, ..., \widehat{\mathbf{V}}^{(T)}\}$
RNN-RSM Parameters: $\theta = \{\mathbf{W_{uh}}, \mathbf{W_{vh}}, \mathbf{W_{uv}}, \mathbf{W_{vu}}, \mathbf{W_{uu}}, \mathbf{b_v}, \mathbf{b_u}, \mathbf{b_h}, \mathbf{b_v}^{(t)}, \mathbf{b_h}^{(t)}, \mathbf{u}^{(0)}\}$
1: Propagate $\mathbf{u}^{(t)}$ in RNN portion of the graph using eq 2.
2: Compute $\mathbf{b_v}^{(t)}$ and $\mathbf{b_h}^{(t)}$ using eq 1.
3: Generate negatives $\mathbf{V}^{(t)*}$ using k-step Gibbs sampling.
4: Estimate the gradient of the cost C w.r.t. parameters of RSM $\mathbf{W_{vh}}$, $\mathbf{b_v}^{(t)}$ and $\mathbf{b_h}^{(t)}$ using eq 5.
5: Compute gradients (eq 6) w.r.t. RNN connections ($\mathbf{W_{uh}}$, $\mathbf{W_{uv}}$, $\mathbf{W_{uu}}$, $\mathbf{W_{vu}}$, $\mathbf{u}^0$) and biases ($\mathbf{b_v}$, $\mathbf{b_h}$, $\mathbf{b_u}$).
6: **Goto** *step 1* until stopping_criteria (early stopping or maximum iterations reached)

where $\mathbf{W_{uv}}$, $\mathbf{W_{uh}}$ and $\mathbf{W_{vu}}$ are weights connecting RNN and RSM portions (Figure 2). $\mathbf{b_u}$ is the bias of $\mathbf{u}$ and $\mathbf{W_{uu}}$ is the weight between RNN hidden units. $\hat{\mathbf{v}}_n^{(t)}$ is a vector of $\hat{v}_n^k$ (denotes the count for the k^{th} word in n^{th} document). $\sum_{n=1}^{N^{(t)}} \hat{\mathbf{v}}_n^{(t)}$ refers to the sum of observed vectors across documents at time step t where each document is represented as-

$$\hat{\mathbf{v}}_n^{(t)} = [\{\hat{v}_n^{k,(t)}\}_{k=1}^{K}] \text{ and } \hat{v}_n^{k,(t)} = \sum_{i=1}^{D_n^{(t)}} v_{n,i}^{k,(t)} \quad (3)$$

where $v_{n,i}^{k,(t)}$=1 if visible unit i takes on k^{th} value.

In each RSM, a separate RBM is created for each document in the collection at time step t with $D_n^{(t)}$ softmax units, where $D_n^{(t)}$ is the count of words in the n^{th} document. Consider a document of $D_n^{(t)}$ words, the *energy* of the state $\{\mathbf{V}_n^{(t)}, \mathbf{h}_n^{(t)}\}$ at time step, t is given by-

$$\begin{aligned} E(\mathbf{V}_n^{(t)}, \mathbf{h}_n^{(t)}) = &- \sum_{j=1}^{F} \sum_{k=1}^{K} h_{n,j}^{(t)} W_j^k \hat{v}_n^{k,(t)} \\ &- \sum_{k=1}^{\mathrm{K}} \hat{v}_n^{k,(t)} b_v^k - D_n^{(t)} \sum_{j=1}^{F} b_{h,j} h_{n,j}^{(t)} \end{aligned}$$

Observe that the bias terms on hidden units are scaled up by document length to allow hidden units to stabilize when dealing with different-sized documents. The corresponding energy-probability relation in the energy-based model is-

$$P(\mathbf{V}_n^{(t)}) = \frac{1}{Z_n^{(t)}} \sum_{\mathbf{h}_n^{(t)}} \exp(-E(\mathbf{V}_n^{(t)}, \mathbf{h}_n^{(t)})) \quad (4)$$

where $Z_n^{(t)} = \sum_{\mathbf{V}_n^{(t)}} \sum_{\mathbf{h}_n^{(t)}} \exp(-E(\mathbf{V}_n^{(t)}, \mathbf{h}_n^{(t)}))$ is the normalization constant. The lower bound on the log likelihood of the data takes the form:

$$\begin{aligned} \ln P(\mathbf{V}_n^{(t)}) &\geq \sum_{\mathbf{h}^{(t)}} Q(\mathbf{h}_n^{(t)} | \mathbf{V}_n^{(t)}) \ln P(\mathbf{V}_n^{(t)}, \mathbf{h}_n^{(t)}) + H(Q) \\ &= \ln P(\mathbf{V}_n^{(t)}) - KL[Q(\mathbf{h}_n^{(t)} | \mathbf{V}_n^{(t)}) || P(\mathbf{h}_n^{(t)} | \mathbf{V}_n^{(t)})] \end{aligned}$$

Year	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	Total
ACL	58	73	250	83	79	70	177	112	134	134	307	204	214	243	270	349	227	398	331	3713
EMNLP	15	24	15	36	29	21	42	29	58	28	75	132	115	164	125	149	140	206	228	1756
ACL+EMNLP	73	97	265	119	108	91	219	141	192	162	382	336	329	407	395	498	367	604	559	**5469**

Table 1: Number of papers from ACL and EMNLP conferences over the years

where $H(\cdot)$ is the entropy and Q is the approximating posterior. Similar to Deep Belief Networks (Hinton et al., 2006), adding an extra layer improves lower bound on the log probability of data, we introduce the extra layer via RSM biases that propagates the prior via RNN connections. The dependence analogy follows-

$$E(\mathbf{V}_n^{(t)}, \mathbf{h}_n^{(t)}) \propto \tfrac{1}{\mathbf{b_v}^{(t)}} \text{ and } E(\mathbf{V}_n^{(t)}, \mathbf{h}_n^{(t)}) \propto \tfrac{1}{\mathbf{b_h}^{(t)}}$$

$$\ln P(\mathbf{V}_n^{(t)}) \propto \tfrac{1}{E(\mathbf{V}_n^{(t)}, \mathbf{h}_n^{(t)})}; \ln P(\widehat{\mathbf{V}}_n^{(t)}) \propto \ln P(\{\widehat{\mathbf{V}}_n^{\tau}\}_{\tau<t})$$

Observe that the prior is seen as the deterministic hidden representation of latent topics and injected into each hidden state of RSMs, that enables the likelihood of the data to model complex temporal densities i.e. heteroscedasticity in document collections ($\widehat{\mathfrak{V}}$) and temporal topics ($\mathbf{H}$).

Gradient Approximations: The *cost* in RNN-RSM is: $C = \sum_{t=1}^{T} C_t \equiv \sum_{t=1}^{T} -\ln P(\widehat{\mathbf{V}}^{(t)})$

Due to intractable Z, the gradient of cost at time step t w.r.t. (with respect to) RSM parameters are approximated by k-step Contrastive Divergence (CD) (Hinton, 2002). The gradient of the negative log-likelihood of a document collection $\{\mathbf{V}_n^{(t)}\}_{n=1}^{N^{(t)}}$ w.r.t. RSM parameter $\mathbf{W_{vh}}$,

$$\begin{aligned}
&\frac{1}{N^{(t)}} \sum_{n=1}^{N^{(t)}} \frac{\partial(-\ln P(\mathbf{V}_n^{(t)}))}{\partial \mathbf{W_{vh}}} \\
&= \frac{1}{N^{(t)}} \sum_{n=1}^{N^{(t)}} \frac{\partial \mathfrak{F}(\mathbf{V}_n^{(t)})}{\partial \mathbf{W_{vh}}} - \frac{\partial(-\ln Z_n^{(t)})}{\partial \mathbf{W_{vh}}} \\
&= \underbrace{\mathsf{E}_{P_{data}}[\frac{\partial \mathfrak{F}(\mathbf{V}_n^{(t)})}{\partial \mathbf{W_{vh}}}]}_{\text{data-dependent expectation}} - \underbrace{\mathsf{E}_{P_{model}}[\frac{\partial \mathfrak{F}(\mathbf{V}_n^{(t)})}{\partial \mathbf{W_{vh}}}]}_{\text{model's expectation}} \\
&\simeq \frac{1}{N^{(t)}} \sum_{n=1}^{N^{(t)}} \frac{\partial \mathfrak{F}(\mathbf{V}_n^{(t)})}{\partial \mathbf{W_{vh}}} - \frac{\partial \mathfrak{F}(\mathbf{V}_n^{(t)*})}{\partial \mathbf{W_{vh}}}
\end{aligned}$$

The second term is estimated by negative samples $\mathbf{V}_n^{(t)*}$ obtained from k-step Gibbs chain starting at $\mathbf{V}_n^{(t)}$ samples. $P_{data}(\widehat{\mathbf{V}}^{(t)}, \mathbf{h}^{(t)}) = P(\mathbf{h}^{(t)}|\widehat{\mathbf{V}}^{(t)}) P_{data}(\widehat{\mathbf{V}}^{(t)})$ and $P_{data}(\widehat{\mathbf{V}}^{(t)}) = \frac{1}{N^{(t)}} \sum_{n}^{N^{(t)}} \delta(\widehat{\mathbf{V}}^{(t)} - \mathbf{V}_n^{(t)})$ is the empirical distribution on the observable. $P_{model}(\mathbf{V}_n^{(t)*}, \mathbf{h}_n^{(t)})$ is defined in eq. 4. The free energy $\mathfrak{F}(\mathbf{V}_n^{(t)})$ is related to normalized probability of $\mathbf{V}_n^{(t)}$ as $P(\mathbf{V}_n^{(t)}) \equiv \exp^{-\mathfrak{F}(\mathbf{V}_n^{(t)})} / Z_n^{(t)}$ and as follows-

$$\begin{aligned}
\mathfrak{F}(\mathbf{V}_n^{(t)}) = &- \sum_{k=1}^{K} \hat{v}_n^{k,(t)} b_v^k - \sum_{j=1}^{F} \log(1+ \\
&\exp(D_n^{(t)} b_{h,j} + \sum_{k=1}^{K} \hat{v}_n^{k,(t)} W_j^k))
\end{aligned}$$

Gradient approximations w.r.t. RSM parameters,

$$\begin{aligned}
\frac{\partial C_t}{\partial \mathbf{b_v}^{(t)}} &\simeq \sum_{n=1}^{N^{(t)}} \hat{\mathbf{v}}_n^{(t)*} - \hat{\mathbf{v}}_n^{(t)} \\
\frac{\partial C_t}{\partial \mathbf{b_h}^{(t)}} &\simeq \sum_{n=1}^{N^{(t)}} \sigma(\mathbf{W_{vh}} \hat{\mathbf{v}}_n^{(t)*} - D_n^{(t)} \mathbf{b_h}^{(t)}) \\
&\quad - \sigma(\mathbf{W_{vh}} \hat{\mathbf{v}}_n^{(t)} - D_n^{(t)} \mathbf{b_h}^{(t)}) \\
\frac{\partial C_t}{\partial \mathbf{W_{vh}}} &\simeq \sum_{t=1}^{T} \sum_{n=1}^{N^{(t)}} \sigma(\mathbf{W_{vh}} \hat{\mathbf{v}}_n^{(t)*} - D_n^{(t)} \mathbf{b_h}^{(t)}) \\
&\quad \hat{\mathbf{v}}_n^{(t)*T} - \sigma(\mathbf{W_{vh}} \hat{\mathbf{v}}_n^{(t)} - D_n^{(t)} \mathbf{b_h}^{(t)}) \hat{\mathbf{v}}_n^{(t)T}
\end{aligned} \tag{5}$$

The estimated gradients w.r.t. RSM biases are back-propagated via hidden-to-bias parameters (eq 1) to compute gradients w.r.t. RNN connections ($\mathbf{W_{uh}}$, $\mathbf{W_{uv}}$, $\mathbf{W_{vu}}$ and $\mathbf{W_{uu}}$) and biases ($\mathbf{b_h}$, $\mathbf{b_v}$ and $\mathbf{b_u}$).

$$\begin{aligned}
\frac{\partial C}{\partial \mathbf{W_{uh}}} &= \sum_{t=1}^{T} \frac{\partial C_t}{\partial \mathbf{b_h}^{(t)}} \mathbf{u}^{(t-1)T} \\
\frac{\partial C}{\partial \mathbf{W_{uv}}} &= \sum_{t=1}^{T} \frac{\partial C_t}{\partial \mathbf{b_v}^{(t)}} \mathbf{u}^{(t-1)T} \\
\frac{\partial C}{\partial \mathbf{W_{vu}}} &= \sum_{t=1}^{T} \frac{\partial C_t}{\partial \mathbf{u}^{(t)}} \mathbf{u}^{(t)}(1 - \mathbf{u}^{(t)}) \sum_{n=1}^{N^{(t)}} \hat{\mathbf{v}}_n^{(t)T} \\
\frac{\partial C}{\partial \mathbf{b_h}} &= \sum_{t=1}^{T} \frac{\partial C_t}{\partial \mathbf{b_h}^{(t)}} \text{ and } \frac{\partial C}{\partial \mathbf{b_v}} = \sum_{t=1}^{T} \frac{\partial C_t}{\partial \mathbf{b_v}^{(t)}} \\
\frac{\partial C}{\partial \mathbf{b_u}} &= \sum_{t=1}^{T} \frac{\partial C_t}{\partial \mathbf{u}^{(t)}} \mathbf{u}^{(t)}(1 - \mathbf{u}^{(t)}) \\
\frac{\partial C}{\partial \mathbf{W_{uu}}} &= \sum_{t=1}^{T} \frac{\partial C_t}{\partial \mathbf{u}^{(t)}} \mathbf{u}^{(t)}(1 - \mathbf{u}^{(t)}) \mathbf{u}^{(t-1)T}
\end{aligned} \tag{6}$$

Parameter	Value(s)	Optimal
epochs	1000	1000
CD iterations	15	15
learning rate	0.1, 0.03, 0.001	0.001
hidden size	20, 30, 50	30

Table 2: Hyperparameters for RNN-RSM model

model	metric				
	SumPPL	*Err*	*mean-COH*	*median-COH*	*TTD*
DTM	10.9	8.10	0.1514	0.1379	0.084
RNN-RSM	**3.8**	**7.58**	**0.1620**	**0.1552**	0.268

Table 3: State-of-the-art Comparison: Generalization (*PPL* and *Err*), Topic Interpretation (*COH*) and Evolution (*TTD*) in DTM and RNN-RSM models

For the single-layer RNN-RSM, the BPTT recurrence relation for $0 \leq t < T$ is given by-

$$\frac{\partial C_t}{\partial \mathbf{u}^{(t)}} = \mathbf{W_{uu}} \frac{\partial C_{t+1}}{\partial \mathbf{u}^{(t+1)}} \mathbf{u}^{(t+1)}(1 - \mathbf{u}^{(t+1)}) + \mathbf{W_{uh}} \frac{\partial C_{t+1}}{\partial \mathbf{b_h}^{(t+1)}} + \mathbf{W_{uv}} \frac{\partial C_{t+1}}{\partial \mathbf{b_v}^{(t+1)}}$$

where $\mathbf{u}^{(0)}$ being a parameter and $\frac{\partial C_T}{\partial \mathbf{u}^{(T)}} = 0$.

See *Training RNN-RSM with BPTT* in Algo 1.

3 Evaluation

3.1 Dataset and Experimental Setup

We use the processed dataset (Gollapalli and Li, 2015), consisting of EMNLP and ACL conference papers from the year 1996 through 2014 (Table 1). We combine papers for each year from the two venues to prepare the document collections over time. We use ExpandRank (Wan and Xiao, 2008) to extract top 100 keyphrases for each paper, including unigrams and bigrams. We split the bigrams to unigrams to create a dictionary of all unigrams and bigrams. The dictionary size (K) and word count are 3390 and 5.19 M, respectively.

We evaluate RNN-RSM against static (RSM, LDA) and dynamic (DTM) topics models for topic and keyword evolution in NLP research over time. Individual 19 different RSM and LDA models are trained for each year, while DTM[2] and RNN-RSM are trained over the years with 19 time steps, where paper collections for a year is input at each time step. RNN-RSM is initialized with RSM ($\mathbf{W_{vh}}$, $\mathbf{b_v}$, $\mathbf{b_h}$) trained for the year 2014.

We use perplexity to choose the number of topics (=30). See Table 2 for hyperparameters.

3.2 Generalization in Dynamic Topic Models

Perplexity: We compute the perplexity on unobserved documents ($\widehat{\mathbf{V}}^{(t)}$) at each time step as

$$\text{PPL}(\widehat{\mathbf{V}}^{(t)}, t) = \exp\left(-\frac{1}{N^{(t)}} \frac{\sum_{n=1}^{N^{(t)}} \log P(\mathbf{V}_n^{(t)})}{\sum_{n=1}^{N^{(t)}} D_n^{(t)}}\right)$$

where t is the time step. $N^{(t)}$ is the number of documents in a collection ($\widehat{\mathbf{V}}^{(t)}$) at time t. Better models have lower perplexity values, suggesting less uncertainties about the documents. For held-out documents, we take 10 documents from each time step i.e. total 190 documents and compute perplexity for 30 topics. Fig 3d shows the comparison of perplexity values for unobserved documents from DTM and RNN-RSM at each time step. The *SumPPL* (Table 3) is the sum of PPL values for the held-out sets of each time step.

Document Time Stamp Prediction: To further assess the dynamic topics models, we split the document collections at each time step into 80-20% train-test, resulting in 1067 held-out documents. We predict the time stamp (dating) of a document by finding the most likely (with the lowest perplexity) location over the time line. See the *mean absolute error* (*Err*) in year for the held-out in Table 3. Note, we do not use the time stamp as observables during training.

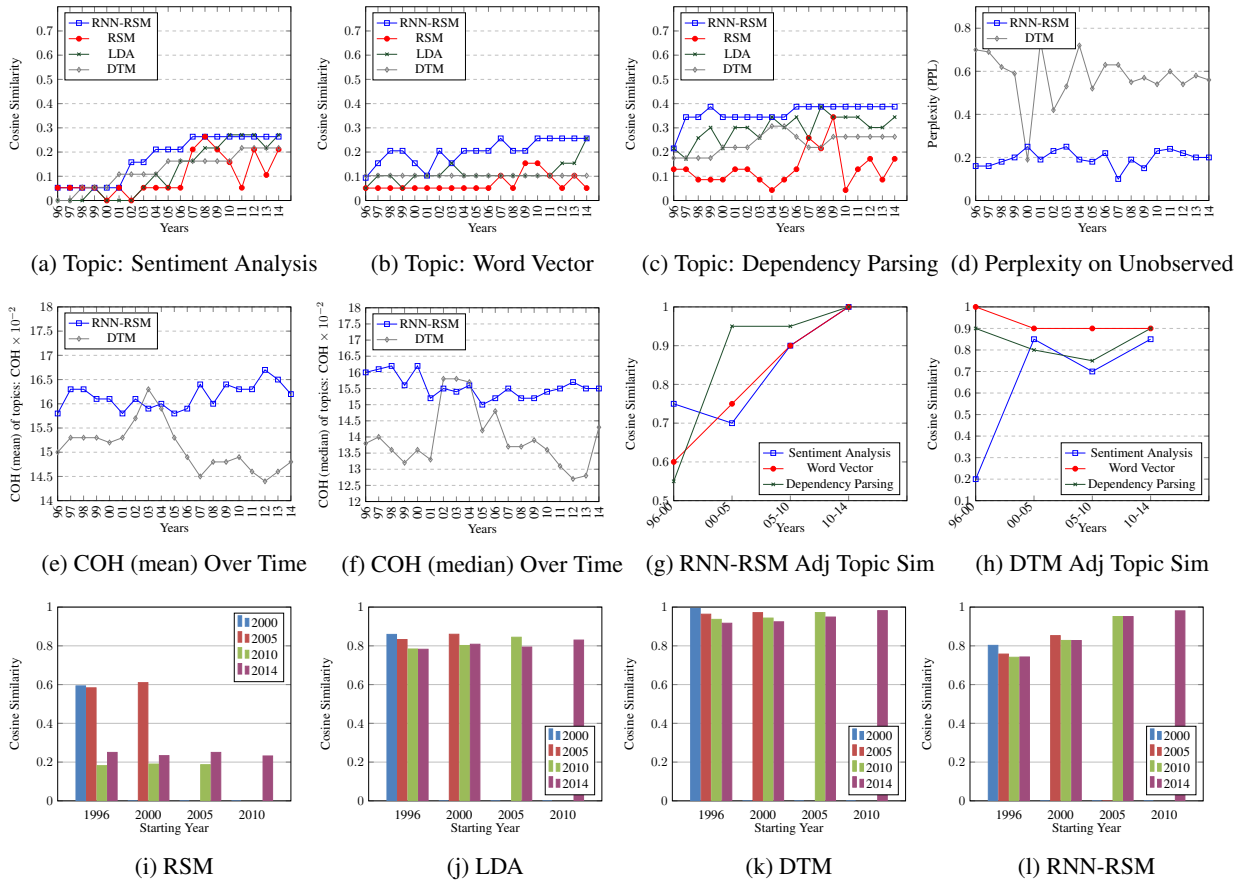
3.3 TSD, TED: Topic Evolution over Time

Topic Detection: To extract topics from each RSM, we compute posterior $P(\widehat{\mathbf{V}}^{(t)}|h_j = 1)$ by activating a hidden unit and deactivating the rest in a hidden layer. We extract the top 20 terms for every 30 topic set from 1996-2014, resulting in $|Q|_{max} = 19 \times 30 \times 20$ possible topic terms.

Topic Popularity: To determine topic *popularity*, we selected three popular topics (*Sentiment Analysis*, *Word Vector* and *Dependency Parsing*) in NLP research and create a set[3] of key-terms (including unigrams and bigrams) for each topic. We compute cosine similarity of the key-terms defined for each selected topic and topics discovered by the topic models over the years. We consider the discovered topic that is the most similar to the key-terms in the target topic and plot the similarity values in Figure 3a, 3b and 3b. Observe that RNN-RSM shows better topic evolution for the three emerging topics. LDA and RSM show

[2] https://radimrehurek.com/gensim/models/dtmmodel.html

[3] topic-terms to be released with code



(a) Topic: Sentiment Analysis (b) Topic: Word Vector (c) Topic: Dependency Parsing (d) Perplexity on Unobserved

(e) COH (mean) Over Time (f) COH (median) Over Time (g) RNN-RSM Adj Topic Sim (h) DTM Adj Topic Sim

(i) RSM (j) LDA (k) DTM (l) RNN-RSM

Figure 3: (a, b, c): Topic popularity by LDA, RSM, DTM and RNN-RSM over time (d): Perplexity on the unobserved document collections over time (e, f): Mean and Median Topic Coherence (g, h): Topic Evolution (i,j,k,l): Topic focus change over time. Adj- Adjacent; Sim- Similarity

topical locality in Figure 3c attributed to no correlation in topic dynamics over time, while in Figure 3b, DTM does not capture evolution of topic *Word Vector*.

Topic Drift (Focus Change): To compute the topic *focus* change over the years, we first split the time period 1996-2014 into five parts:{1996, 2000, 2005, 2010, 2014}. The cosine similarity scores are computed between the topic sets discovered in a particular year and the years preceding it in the above set, for example the similarity scores between the topic-terms in (1996, 2000), (1996, 2005), (1996, 2010) and (1996, 2014), respectively. Figure 3i, 3j, 3k and 3l demonstrate that RNN-RSM shows higher *convergence* in topic focus over the years, compared to LDA and RSM. In RNN-RSM, the topic similarity is gradually increased over time, however not in DTM. The higher similarities in the topic sets indicate that new/existing topics and words do not appear/disappear over time.

We compute topic-term drift (TTD) to show the changing topics from initial to final year, as

$$TTD = 1.0 - cosineSimilarity(\mathbf{Q}^{(t)}, \mathbf{Q}^{(t')})$$

where $\mathbf{Q}$ is the set of all topic-terms for time step t. Table 3 shows that TTD (where t=1996 and t'=2014) are 0.268 and 0.084 for RNN-RSM and DTM, respectively. It suggests that the higher number of new topic-terms evolved in RNN-RSM, compared to DTM. Qualitatively, the Table 4 shows the topics observed with the highest and lowest cosine drifts in DTM and RNN-RSM.

In Figure 3g and 3h, we also illustrate the temporal *evolution* (drift) in the selected topics by computing cosine similarity on their adjacent topic vectors over time. The topic vectors are selected similarly as in computing topic popularity. We observe better TED in RNN-RSM than DTM for the three emerging topics in NLP research. For instance, for the selected topic *Word Vector*, the red line in DTM (Fig 3h) shows no drift (for x-axis 00-05, 05-10 and 10-14), suggesting the topic-terms in the adjacent years are similar and does not evolve.

Drift	Model (year)	Topic Terms
0.20	**DTM** (1996)	document, retrieval, query, documents, information, search, information retrieval, queries, terms, words, system, results, performance, method, approach
	DTM (2014)	document, query, search, documents, queries, information, retrieval, method, results, information retrieval, research, terms, other, approach, knowledge
0.53	**DTM** (1996)	semantic, lexical, structure, syntactic, argument, frame, example, lexicon, information, approach, source, function, figure, verbs, semantic representation
	DTM (2014)	semantic, argument, frame, sentence, syntactic, semantic parsing, structure, semantic role, example, role labeling, language, learning, logical form, system, lexicon
0.20	**RNN-RSM** (1996)	reordering, statistical machine, translation model, translations, arabic, word align, translation probability, word alignment, translation system, source word, ibm model, source sentence, english translation, target language, word segmentation
	RNN-RSM (2014)	reordering, statistical machine, translation model, translations, arabic, word align, translation probability, word alignment, translation system, source word, reordering model, bleu score, smt system, english translation, target language
0.53	**RNN-RSM** (1996)	input, inference, semantic representation, distributional models, logical forms, space model, clustering algorithm, space models, similar word, frequent word, meaning representation, lexical acquisition, new algorithm, same context, multiple words
	RNN-RSM (2014)	input, inference, word vector, word vectors, vector representation, semantic representation, distributional models, semantic space, space model, semantic parser, vector representations, neural language, logical forms, cosine similarity, clustering algorithm

Table 4: Topics (top 15 words) with the highest and lowest drifts (cosine) observed in DTM and RNN-RSM

3.4 Topic Interpretability

Beyond perplexities, we also compute topic coherence (Chang et al., 2009; Newman et al., 2009; Das et al., 2015) to determine the meaningful topics captured. We use the coherence measure proposed by Aletras and Stevenson (2013) that retrieves co-occurrence counts for the set of topic words using Wikipedia as a reference corpus to identify context features (window=5) for each topic word. Relatedness between topic words and context features is measured using normalized pointwise mutual information (NPMI), resulting in a single vector for every topic word. The coherence (*COH*) score is computed as the arithmetic mean of the cosine similarities between all word pairs. Higher scores imply more coherent topics. We use Palmetto[4] library to estimate coherence.

Quantitative: We compute mean and median coherence scores for each time step using the corresponding topics, as shown in Fig 3e and 3f. Table 3 shows *mean-COH* and *median-COH* scores, computed by mean and median of scores from Fig 3e and 3f, respectively. Observe that RNN-RSM captures topics with higher coherence.

Qualitative: Table 5 shows topics (top-10 words) with the highest and lowest coherence scores.

DTM (2001)	**RNN-RSM** (2001)	**DTM** (2012)	**RNN-RSM** (1997)
semantic	words	discourse	parse
frame	models	relation	cluster
argument	grammar	relations	clustering
syntactic	trees	structure	results
structure	dependency parsing	sentence	query
lexical	parsers	class	pos tag
example	dependency trees	lexical	queries
information	parsing	argument	retrieval
annotation	parse trees	corpus	coreference
lexicon	dependency parse	other	logical form
COH: 0.268	0.284	0.064	0.071

Table 5: Topics with the highest and lowest coherence

3.5 TTC: Trending Keywords over time

We demonstrate the capability of RNN-RSM to capture word evolution (usage) in topics over time. We define: *keyword-trend* and SPAN. The *keyword-trend* is the appearance/disappearance of the keyword in topic-terms detected over time, while SPAN is the length of the longest sequence of the keyword appearance in its keyword trend.

Let $\widehat{\mathbf{Q}}_{model} = \{\mathbf{Q}^{(t)}_{model}\}_{t=1}^{T}$ be a set of sets[5] of topic-terms discovered by the $model$ (LDA, RSM, DTM and RNN-RSM) over different time steps. Let $\mathbf{Q}^{(t)} \in \widehat{\mathbf{Q}}_{model}$ be the topic-terms at time step t. The keyword-trend for a keyword k is a time-ordered sequence of 0s and 1s, as

$$\text{trend}_k(\widehat{\mathbf{Q}}) = [\text{find}(k, \mathbf{Q}^{(t)})]_{t=1}^{T}$$

$$\text{where;} \quad \text{find}(k, \mathbf{Q}^{(t)}) = \begin{cases} 1 & \text{if } k \in \mathbf{Q}^{(t)} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

And the SPAN (S_k) for the kth keyword is-

$$S_k(\widehat{\mathbf{Q}}) = \text{length}\big(\text{longestOnesSeq}(\text{trend}_k(\widehat{\mathbf{Q}})\big)$$

We compute keyword-trend and SPAN for each term from the set of some popular terms. We define average-SPAN for all the topic-terms appearing in the topics discovered over the years,

$$\text{avg-SPAN}(\widehat{\mathbf{Q}}) = \frac{1}{||\widehat{\mathbf{Q}}||} \sum_{\{k|\mathbf{Q}^{(t)} \in \widehat{\mathbf{Q}} \wedge k \in \mathbf{Q}^{(t)}\}} \frac{S_k(\widehat{\mathbf{Q}})}{\hat{v}^k}$$

$$= \frac{1}{||\widehat{\mathbf{Q}}||} \sum_{\{k|\mathbf{Q}^{(t)} \in \widehat{\mathbf{Q}} \wedge k \in \mathbf{Q}^{(t)}\}} S_k^{dict}(\widehat{\mathbf{Q}})$$

[4] github.com/earthquakesan/palmetto-py

[5] a set by **bold** and set of sets by $\widehat{\textbf{bold}}$

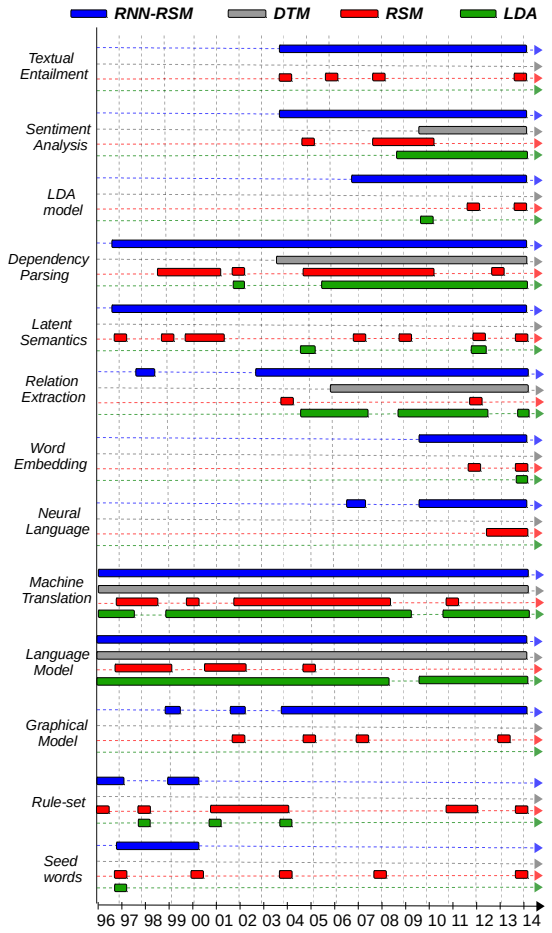


Figure 4: Keyword-trend by RNN-RSM, DTM, RSM, LDA. Bar: Keyword presence in topics for the year

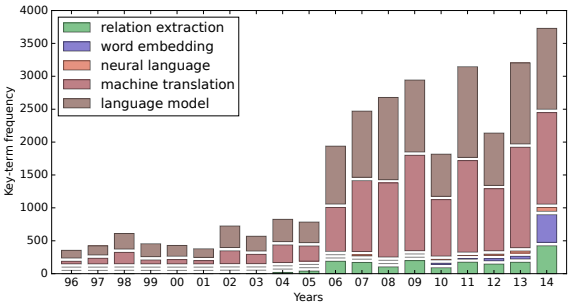


Figure 5: Key-term frequency in the input over years

Term	$\hat{v^k}$	LDA		RSM		DTM		RNN-RSM	
		S_k	S_k^{dict}	S_k	S_k^{dict}	S_k	S_k^{dict}	S_k	S_k^{dict}
Textual entailment	918	0	.000	1	.001	0	.000	**11**	.011
Sentiment analysis	1543	6	.004	3	.002	5	.0032	**11**	0.007
Lda model	392	1	.003	1	.002	0	.000	**8**	.020
Dependency parsing	3409	9	.003	5	.001	11	.0032	**18**	.005
Latent semantic	974	1	.001	2	.002	0	.000	**18**	.018
Relation extraction	1734	4	.002	1	.001	9	.0052	**12**	.007
Word embedding	534	1	.002	1	.002	0	.000	**5**	.009
Neural language	121	0	.000	3	.025	0	.000	**5**	.041
Machine translation	11741	11	.001	7	.001	**19**	.0016	**19**	.002
Language model	11768	13	.001	3	.000	**19**	.0016	**19**	.002
Graphical model	680	0	.000	1	.001	0	.000	**11**	.016
Rule set	589	1	.0017	**4**	.0068	0	.000	2	.0034
Seed words	396	1	.0025	1	.0025	0	.000	**4**	.0101
avg-SPAN($\widehat{\mathbf{Q}}$)			.002		.007		.003		**.018**
$\|\|\widehat{\mathbf{Q}}_{model}\|\|$		926		2274		335		731	

Table 6: SPAN (S_k) for selected terms, avg-SPAN and set $||\widehat{\mathbf{Q}}||$ by LDA, RSM, DTM and RNN-RSM

where $||\widehat{\mathbf{Q}}|| = |\{k|\mathbf{Q}^{(t)} \in \widehat{\mathbf{Q}} \wedge k \in \mathbf{Q}^{(t)}\}|$ is the count of unique topic-terms and $\hat{v^k} = \sum_{t=1}^{T}\sum_{j=1}^{D_t} v_{j,t}^k$ denotes the count of k^{th} keyword.

In Figure 4, the keyword-trends indicate emergence (appearance/disappearance) of the selected popular terms in topics discovered in ACL and EMNLP papers over time. Observe that RNN-RSM captures longer SPANs for popular keywords and better word usage in NLP research. For example: *Word Embedding* is one of the top keywords, appeared locally (Figure 5) in the recent years. RNN-RSM detects it in the topics from 2010 to 2014, however DTM does not. Similarly, for *Neural Language*. However, *Machine Translation* and *Language Model* are globally appeared in the input document collections over time and captured in the topics by RNN-RSM and DTM. We also show keywords (*Rule-set* and *Seed Words*) that disappeared in topics over time.

Higher SPAN suggests that the model is capable in capturing trending keywords. Table 6 shows corresponding comparison of SPANs for the 13 selected keywords. The SPAN S_k for each keyword is computed from Figure 4. Observe that $||\widehat{\mathbf{Q}}||_{DTM} < ||\widehat{\mathbf{Q}}||_{RNN-RSM}$ suggests new topics and words emerged over time in RNN-RSM, while higher SPAN values in RNN-RSM suggest better trends. Figure 6 shows how the word usage, captured by DTM and RNN-RSM for the topic *Word Vector*, changes over 19 years in NLP research. RNN-RSM captures popular terms *Word Embedding* and *Word Representation* emerged in it.

4 Discussion: RNN-RSM vs DTM

Architecture: RNN-RSM treats document's stream as high dimensional sequences over time and models the complex conditional probability distribution i.e. *heteroscedasticity* in document collections and topics over time by a temporal stack of RSMs (undirected graphical model), conditioned on time-feedback connections using RNN (Rumelhart et al., 1985). It has two hidden layers: **h** (stochastic binary) to capture topical information, while **u** (deterministic) to convey temporal information via BPTT that models the topic dependence at a time step t on *all* the previous steps $\tau < t$. In contrast, DTM is built upon

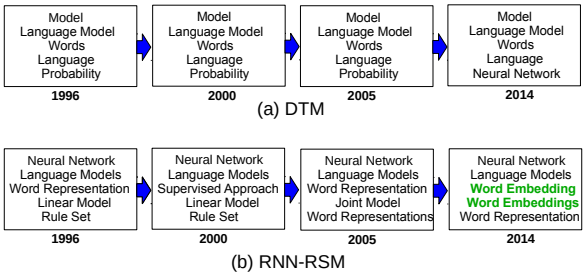


Figure 6: Word usage for emerging topic *Word Vector* over time, captured by DTM and RNN-RSM

LDA (directed model), where Dirichlet distribution on words is not amenable to sequential modeling, therefore its natural parameters (topic and topic proportion distributions) for each topic are chained, instead of latent topics that results in intractable inference in topic detection and chaining.

Topic Dynamics: The introduction of explicit connection in latent topics in RNN-RSM allow new topics and words for the underlying topics to appear or disappear over time by the dynamics of topic correlations. As discussed, the distinction of $\mathbf{h}$ and $\mathbf{u}$ permits the latent topic $\mathbf{h}^{(t)}$ to capture new topics, that may not be captured by $\mathbf{h}^{(t-1)}$.

DTM assumes a fixed number of global topics and models their distribution over time. However, there is no such assumption in RNN-RSM. We fixed the topic count in RNN-RSM at each time step, since $\mathbf{W_{vh}}$ is fixed over time and RSM biases turn off/on terms in each topic. However, this is fundamentally different for DTM. E.g. a unique label be assigned to each of the 30 topics at any time steps t and t'. DTM follows the sets of topic labels: $\{TopicLabels^{(t)}\}_{k=1}^{30} = \{TopicLabels^{(t')}\}_{k=1}^{30}$, due to eq (1) in Blei and Lafferty (2006) (discussed in section 5) that limits DTM to capture new (or local) topics or words appeared over time. It corresponds to the keyword-trends (section 3.5).

Optimization: The RNN-RSM is based on Gibbs sampling and BPTT for inference while DTM employs complex variational methods, since applying Gibbs sampling is difficult due to the nonconjugacy of the Gaussian and multinomial distributions. Thus, easier learning in RNN-RSM.

For all models, approximations are solely used to compute the likelihood, either using variational approaches or contrastive divergence; perplexity was then computed based on the approximated likelihood. More specifically, we use variational approximations to compute the likelihood for DTM (Blei and Lafferty, 2006). For RSM and RNN-RSM, the respective likelihoods are approximated using the standard Contrastive Divergence (CD). While there are substantial differences between variational approaches and CD, and thus in the manner the likelihood for different models is estimated - both approximations work well for the respective family of models in terms of approximating the true likelihood. Consequently, perplexities computed based on these approximated likelihoods are indeed comparable.

5 Conclusion and Future Work

We have proposed a neural temporal topic model which we name as RNN-RSM, based on probabilistic undirected graphical topic model RSM with time-feedback connections via deterministic RNN, to capture temporal relationships in historical documents. The model is the first of its kind that learns topic dynamics in collections of different-sized documents over time, within the generative and neural network framework. The experimental results have demonstrated that RNN-RSM shows better generalization (perplexity and time stamp prediction), topic interpretation (coherence) and evolution (popularity and drift) in scientific articles over time. We also introduced SPAN to illustrate topic characterization.

In future work, we forsee to investigate learning dynamics in variable number of topics over time. It would also be an interesting direction to investigate the effect of the skewness in the distribution of papers over all years. Further, we see a potential application of the proposed model in learning the time-aware i.e. dynamic word embeddings (Aitchison, 2001; Basile et al., 2014; Bamler and Mandt, 2017; Rudolph and Blei, 2018; Yao et al., 2018) in order to capture language evolution over time, instead of document topics.

Acknowledgments

We thank Sujatha Das Gollapalli for providing us with the data sets used in the experiments. We express appreciation for our colleagues Florian Buettner, Mark Buckley, Stefan Langer, Ulli Waltinger and Usama Yaseen, and anonymous reviewers for their in-depth review comments. This research was supported by Bundeswirtschaftsministerium (bmwi.de), grant 01MD15010A (Smart Data Web) at Siemens AG- CT Machine Intelligence, Munich Germany.

References

Jean Aitchison. 2001. *Language change: progress or decay?*. Cambridge University Press.

Nikolaos Aletras and Mark Stevenson. 2013. Evaluating topic coherence using distributional semantics. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS)*. Potsdam, Germany, pages 13–22.

James Allan. 2002. Introduction to topic detection and tracking. In *Topic detection and tracking*, Springer, pages 1–16.

James Allan, Jaime G Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. 1998. Topic detection and tracking pilot study final report. In *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*. Virginia, US, pages 194–218.

Robert Bamler and Stephan Mandt. 2017. Dynamic word embeddings. In *Proceedings of the 34th International Conference on Machine Learning*. Sydney, Australia, pages 380–389.

Pierpaolo Basile, Annalina Caputo, and Giovanni Semeraro. 2014. Analysing word meaning over time by exploiting temporal random indexing. In *Proceedings of the 1st Italian Conference on Computational Linguistics (CLiC-it)*. Pisa University Press, Pisa, Italy.

David M. Blei and John D. Lafferty. 2006. Dynamic topic models. In *Proceedings of the 23rd International Conference on Machine Learning*. Association for Computing Machinery, Pittsburgh, Pennsylvania USA, pages 113–120.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Proceedings of Machine Learning Research* 3(Jan):993–1022.

Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. 2012. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *Proceedings of the 29th International Conference on Machine Learning*. Edinburgh, Scotland UK.

Jonathan Chang, Sean Gerrish, Chong Wang, Jordan L. Boyd-Graber, and David M. Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., Vancouver, Canada, pages 288–296.

Rajarshi Das, Manzil Zaheer, and Chris Dyer. 2015. Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, Beijing, China, volume 1, pages 795–804.

Peter V. Gehler, Alex D. Holub, and Max Welling. 2006. The rate adapting poisson model for information retrieval and object recognition. In *Proceedings of the 23rd International Conference on Machine Learning*. Association for Computing Machinery, Pittsburgh, Pennsylvania USA, pages 337–344.

Sujatha Das Gollapalli and Xiaoli Li. 2015. Emnlp versus acl: Analyzing nlp research over time. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 2002–2006.

Pankaj Gupta, Thomas Runkler, Heike Adel, Bernt Andrassy, Hans-Georg Zimmermann, and Hinrich Schütze. 2015a. Deep learning methods for the extraction of relations in natural language text. Technical report, Technical University of Munich, Germany.

Pankaj Gupta, Thomas Runkler, and Bernt Andrassy. 2015b. Keyword learning for classifying requirements in tender documents. Technical report, Technical University of Munich, Germany.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. Osaka, Japan, pages 2537–2547.

Pankaj Gupta, Udhayaraj Sivalingam, Sebastian Pölsterl, and Nassir Navab. 2015c. Identifying patients with diabetes using discriminative restricted boltzmann machines. Technical report, Technical University of Munich, Germany.

David Hall, Daniel Jurafsky, and Christopher D. Manning. 2008. Studying the history of ideas using topic models. In *Proceedings of the conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Honolulu, Hawaii, pages 363–371.

Geoffrey Hinton and Ruslan Salakhutdinov. 2009. Replicated softmax: an undirected topic model. In *Advances in Neural Information Processing Systems 22*. Curran Associates, Inc., Vancouver, Canada, pages 1607–1614.

Geoffrey E. Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural Computation* 14(8):1771–1800.

Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. 2006. A fast learning algorithm for deep belief nets. *Neural Computation* 18(7):1527–1554.

Tomoharu Iwata, Takeshi Yamada, Yasushi Sakurai, and Naonori Ueda. 2010. Online multiscale dynamic topic models. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, Washington DC, USA, pages 663–672.

David Newman, Sarvnaz Karimi, and Lawrence Cavedon. 2009. External evaluation of topic models. In *Proceedings of the 14th Australasian Document Computing Symposium*. Citeseer, Sydney, Australia.

Iulian Pruteanu-Malinici, Lu Ren, John Paisley, Eric Wang, and Lawrence Carin. 2010. Hierarchical bayesian modeling of topics in time-stamped documents. *IEEE transactions on pattern analysis and machine intelligence* 32(6):996–1011.

Maja Rudolph and David Blei. 2018. Dynamic bernoulli embeddings for language evolution. In *Proceedings of the 27th International Conference on World Wide Web Companion*. Lyon, France.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. 1985. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science.

Ankan Saha and Vikas Sindhwani. 2012. Learning evolving and emerging topics in social media: a dynamic nmf approach with temporal regularization. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining*. Association for Computing Machinery, Seattle, Washington USA, pages 693–702.

Aaron Schein, Hanna Wallach, and Mingyuan Zhou. 2016. Poisson-gamma dynamical systems. In *Advances in Neural Information Processing Systems 29*, Curran Associates, Inc., Barcelona, Spain, pages 5005–5013.

Paul Smolensky. 1986. Information processing in dynamical systems: Foundations of harmony theory. Technical report, Colorado University at Boulder Department of Computer Science.

Ilya Sutskever and Geoffrey Hinton. 2007. Learning multilevel distributed representations for high-dimensional sequences. In *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*. San Juan, Puerto Rico, pages 548–555.

Ilya Sutskever, Geoffrey E. Hinton, and Graham W. Taylor. 2009. The recurrent temporal restricted boltzmann machine. In *Advances in Neural Information Processing Systems 22*. Curran Associates, Inc., Vancouver, Canada, pages 1601–1608.

Graham W. Taylor, Geoffrey E. Hinton, and Sam T. Roweis. 2007. Modeling human motion using binary latent variables. In *Advances in Neural Information Processing Systems 20*. Curran Associates, Inc., Vancouver, Canada, pages 1345–1352.

Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schütze. 2016a. Combining recurrent and convolutional neural networks for relation classification. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, San Diego, California USA, pages 534–539.

Ngoc Thang Vu, Pankaj Gupta, Heike Adel, and Hinrich Schütze. 2016b. Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *Proceedings of the Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Shanghai, China, pages 6060–6064.

Xiaojun Wan and Jianguo Xiao. 2008. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd National Conference on Artificial Intelligence*. Chicago, Illinois USA, volume 8, pages 855–860.

Xuerui Wang and Andrew McCallum. 2006. Topics over time: a non-markov continuous-time model of topical trends. In *Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, Philadelphia, Pennsylvania USA, pages 424–433.

Eric P. Xing, Rong Yan, and Alexander G. Hauptmann. 2005. Mining associated text and images with dual-wing harmoniums. In *Proceedings of the 21st Conference on Uncertainty in Artificial Intelligence*. AUAI Press, Edinburgh, Scotland UK.

Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. 2018. Dynamic word embeddings for evolving semantic discovery. In *Proceedings of the 11th ACM International Conference on Web Search and Data Mining (WSDM)*. Association for Computing Machinery, Los Angeles, California USA, pages 673–681.

Chapter 6

Document Informed Neural Autoregressive Topic Models with Distributional Prior

Document Informed Neural Autoregressive Topic Models with Distributional Prior

Pankaj Gupta[1,2], Yatin Chaudhary[1], Florian Buettner[1], Hinrich Schütze[2]
[1]Corporate Technology, Machine-Intelligence (MIC-DE), Siemens AG Munich, Germany
[2]CIS, University of Munich (LMU) Munich, Germany
{pankaj.gupta, yatin.chaudhary, buettner.florian}@siemens.com
pankaj.gupta@campus.lmu.de | inquiries@cislmu.org

Abstract

We address two challenges in topic models: (1) Context information around words helps in determining their actual meaning, e.g., "networks" used in the contexts *artificial neural networks* vs. *biological neuron networks*. Generative topic models infer topic-word distributions, taking no or only little context into account. Here, we extend a neural autoregressive topic model to exploit the full context information around words in a document in a language modeling fashion. The proposed model is named as *iDocNADE*. (2) Due to the small number of word occurrences (i.e., lack of context) in short text and data sparsity in a corpus of few documents, the application of topic models is challenging on such texts. Therefore, we propose a simple and efficient way of incorporating external knowledge into neural autoregressive topic models: we use embeddings as a distributional prior. The proposed variants are named as *DocNADEe* and *iDocNADEe*.

We present novel neural autoregressive topic model variants that consistently outperform state-of-the-art generative topic models in terms of generalization, interpretability (topic coherence) and applicability (retrieval and classification) over 7 long-text and 8 short-text datasets from diverse domains.

Introduction

Probabilistic topic models, such as LDA (Blei, Ng, and Jordan 2003), Replicated Softmax (RSM) (Salakhutdinov and Hinton 2009) and Document Autoregressive Neural Distribution Estimator (DocNADE) (Larochelle and Lauly 2012) are often used to extract topics from text collections and learn document representations to perform NLP tasks such as information retrieval (IR), document classification or summarization.

To motivate our first task of *incorporating full contextual information*, assume that we conduct topic analysis on a collection of research papers from NIPS conference, where one of the popular terms is "networks". However, without context information (nearby and/or distant words), its actual meaning is ambiguous since it can refer to such different concepts as *artificial neural networks* in *computer science* or *biological neural networks* in *neuroscience* or *Computer/data networks* in *telecommunications*. Given the context, one can determine the actual meaning of "networks", for instance, "Extracting rules from artificial neural networks with distributed representations", or "Spikes from the presynaptic neurons and postsynaptic neurons in small networks" or "Studies of neurons or networks under noise in artificial neural networks" or "Packet Routing in Dynamically Changing Networks".

Generative topic models such as LDA or DocNADE infer topic-word distributions that can be used to estimate a document likelihood. While basic models such as LDA do not account for context information when inferring these distributions, more recent approaches such as DocNADE achieve *amplified word and document likelihoods* by accounting for words preceding a word of interest in a document. More specifically, DocNADE (Larochelle and Lauly 2012; Zheng, Zhang, and Larochelle 2016) (Figure 1, Left) is a probabilistic graphical model that learns topics over sequences of words, corresponding to a language model (Manning and Schütze 1999; Bengio et al. 2003) that can be interpreted as a neural network with several parallel hidden layers. To predict the word v_i, each hidden layer $\mathbf{h}_i$ takes as input the sequence of preceding words $\mathbf{v}_{<i}$. However, it does *not* take into account the following words $\mathbf{v}_{>i}$ in the sequence. Inspired by bidirectional language models (Mousa and Schuller 2017) and recurrent neural networks (Elman 1990; Gupta, Schütze, and Andrassy 2016; Vu et al. 2016b; 2016a), trained to predict a word (or label) depending on its full left and right contexts, we extend DocNADE and incorporate full contextual information (all words around v_i) at each hidden layer $\mathbf{h}_i$ when predicting the word v_i in a language modeling fashion with neural topic modeling.

While this is a powerful approach for incorporating contextual information in particular for long texts and corpora with many documents, learning contextual information remains challenging in topic models with short texts and few documents, due to (1) limited word co-occurrences or little context and (2) significant word non-overlap in such short texts. However, distributional word representations (i.e. word embeddings) have shown to capture both the semantic and syntactic relatedness in words and demonstrated impressive performance in natural language processing (NLP) tasks. For example, assume that we conduct topic analysis over the two short text fragments: "*Goldman shares drop sharply downgrade*" and "*Falling market homes*

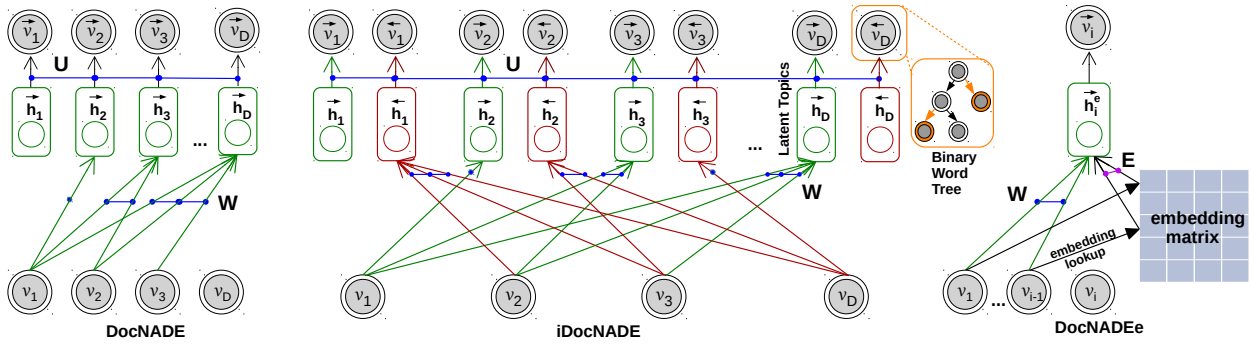


Figure 1: *DocNADE* (left), *iDocNADE* (middle) and DocNADEe (right) models. Blue colored lines signify the connections that share parameters. The observations (*double circle*) for each word v_i are multinomial. Hidden vectors in *green* and *red* colors identify the forward and backward network layers, respectively. Symbols $\overrightarrow{v}_i$ and $\overleftarrow{v}_i$ represent the autoregressive conditionals $p(v_i|\mathbf{v}_{<i})$ and $p(v_i|\mathbf{v}_{>i})$, respectively. Connections between each v_i and hidden units are shared, and each conditional $\overrightarrow{v}_i$ (or $\overleftarrow{v}_i$) is decomposed into a tree of binary logistic regressions, i.e. hierarchical softmax.

weaken economy". Traditional topic models will not be able to infer relatedness between word pairs across sentences such as (*economy*, *shares*) due to the lack of word-overlap between sentences. However, in embedding space, the word pairs (*economy*, *shares*), (*market*, *shares*) and (*falling*, *drop*) have cosine similarities of 0.65, 0.56 and 0.54.

Therefore, we *incorporate word embeddings* as fixed prior in neural topic models in order to introduce complementary information. The proposed neural architectures learn task specific word vectors in association with static embedding priors leading to better text representation for topic extraction, information retrieval, classification, etc.

The multi-fold **contributions** in this work are: **(1)** We propose an advancement in neural autoregressive topic model by incorporating full contextual information around words in a document to boost the likelihood of each word (and document). This enables learning better (*informed*) document representations that we quantify via *generalization* (perplexity), *interpretability* (topic coherence) and *applicability* (document retrieval and classification). We name the proposed topic model as *Document Informed Neural Autoregressive Distribution Estimator* (**iDocNADE**). **(2)** We propose a further extension of DocNADE-like models by incorporating complementary information via word embeddings, along with the standard sparse word representations (e.g., one-hot encoding). The resulting two DocNADE variants are named as *Document Neural Autoregressive Distribution Estimator with Embeddings* (**DocNADEe**) and *Document Informed Neural Autoregressive Distribution Estimator with Embeddings* (**iDocNADEe**). **(3)** We also investigate the two contributions above in the deep versions of topic models. **(4)** We apply our modeling approaches to 8 short-text and 7 long-text datasets from diverse domains. With the learned representations, we show a gain of 5.2% (404 vs 426) in perplexity, 11.1% (.60 vs .54) in precision at retrieval fraction 0.02 and 5.2% (.664 vs .631) in $F1$ for text categorization, compared to the DocNADE model (on average over 15 datasets). *Code* and *supplementary material* are available at `https://github.com/pgcool/iDocNADEe`.

Neural Autoregressive Topic Models

RSM (Salakhutdinov and Hinton 2009), a probabilistic undirected topic model, is a generalization of the energy-based Restricted Boltzmann Machines RBM (Hinton 2002) that can be used to model word counts. NADE (Larochelle and Murray 2011) decomposes the joint distribution of observations into autoregressive conditional distributions, modeled using non-linear functions. Unlike for RBM/RSM, this leads to tractable gradients of the data negative log-likelihood but can only be used to model binary observations.

DocNADE (Figure 1, Left) is a generative neural autoregressive topic model to account for word counts, inspired by RSM and NADE. For a document $\mathbf{v} = [v_1, ..., v_D]$ of size D, it models the joint distribution $p(\mathbf{v})$ of all words v_i, where $v_i \in \{1, ..., K\}$ is the index of the ith word in the dictionary of vocabulary size K. This is achieved by decomposing it as a product of conditional distributions i.e. $p(\mathbf{v}) = \prod_{i=1}^{D} p(v_i|\mathbf{v}_{<i})$ and computing each autoregressive conditional $p(v_i|\mathbf{v}_{<i})$ via a feed-forward neural network for $i \in \{1, ...D\}$,

$$\overrightarrow{\mathbf{h}}_i(\mathbf{v}_{<i}) = g(\mathbf{c} + \textstyle\sum_{k<i} \mathbf{W}_{:,v_k}) \quad (1)$$

$$p(v_i = w|\mathbf{v}_{<i}) = \frac{\exp(b_w + \mathbf{U}_{w,:}\, \overrightarrow{\mathbf{h}}_i(\mathbf{v}_{<i}))}{\sum_{w'} \exp(b_{w'} + \mathbf{U}_{w',:}\, \overrightarrow{\mathbf{h}}_i(\mathbf{v}_{<i}))}$$

where $\mathbf{v}_{<i} \in \{v_1, ..., v_{i-1}\}$. $g(\cdot)$ is a non-linear activation function, $\mathbf{W} \in \mathbb{R}^{H \times K}$ and $\mathbf{U} \in \mathbb{R}^{K \times H}$ are weight matrices, $\mathbf{c} \in \mathbb{R}^H$ and $\mathbf{b} \in \mathbb{R}^K$ are bias parameter vectors. H is the number of hidden units (topics). $\mathbf{W}_{:,<i}$ is a matrix made of the $i-1$ first columns of $\mathbf{W}$. The probability of the word v_i is thus computed using a position-dependent hidden layer $\overrightarrow{\mathbf{h}}_i(\mathbf{v}_{<i})$ that learns a representation based on all previous words $\mathbf{v}_{<i}$; however it does *not* incorporate the following words $\mathbf{v}_{>i}$. Taken together, the log-likelihood of any document $\mathbf{v}$ of arbitrary length can be computed as:

$$\mathcal{L}^{DocNADE}(\mathbf{v}) = \sum_{i=1}^{D} \log p(v_i|\mathbf{v}_{<i}) \quad (2)$$

iDocNADE (Figure 1, Right), our *proposed* model, accounts for the full context information (both previous $\mathbf{v}_{<i}$

and following $\mathbf{v}_{>i}$ words) around each word v_i for a document $\mathbf{v}$. Therefore, the log-likelihood $\mathcal{L}^{iDocNADE}$ for a document $\mathbf{v}$ in *iDocNADE* is computed using forward and backward language models as:

$$\log p(\mathbf{v}) = \frac{1}{2}\sum_{i=1}^{D} \underbrace{\log p(v_i|\mathbf{v}_{<i})}_{\text{forward}} \quad + \underbrace{\log p(v_i|\mathbf{v}_{>i})}_{\text{backward}} \tag{3}$$

i.e., the mean of the forward ($\overrightarrow{\mathcal{L}}$) and backward ($\overleftarrow{\mathcal{L}}$) log-likelihoods. This is achieved in a bi-directional language modeling and feed-forward fashion by computing position dependent *forward* ($\overrightarrow{\mathbf{h}}_i$) and *backward* ($\overleftarrow{\mathbf{h}}_i$) hidden layers for each word i, as:

$$\overrightarrow{\mathbf{h}}_i(\mathbf{v}_{<i}) = g(\overrightarrow{\mathbf{c}} + \sum_{k<i} \mathbf{W}_{:,v_k}) \tag{4}$$

$$\overleftarrow{\mathbf{h}}_i(\mathbf{v}_{>i}) = g(\overleftarrow{\mathbf{c}} + \sum_{k>i} \mathbf{W}_{:,v_k}) \tag{5}$$

where $\overrightarrow{\mathbf{c}} \in \mathbb{R}^H$ and $\overleftarrow{\mathbf{c}} \in \mathbb{R}^H$ are bias parameters in forward and backward passes, respectively. H is the number of hidden units (topics).

Two autoregressive conditionals are computed for each ith word using the forward and backward hidden vectors,

$$p(v_i = w|\mathbf{v}_{<i}) = \frac{\exp(\overrightarrow{b}_w + \mathbf{U}_{w,:}\overrightarrow{\mathbf{h}}_i(\mathbf{v}_{<i}))}{\sum_{w'}\exp(\overrightarrow{b}_{w'} + \mathbf{U}_{w',:}\overrightarrow{\mathbf{h}}_i(\mathbf{v}_{<i}))} \tag{6}$$

$$p(v_i = w|\mathbf{v}_{>i}) = \frac{\exp(\overleftarrow{b}_w + \mathbf{U}_{w,:}\overleftarrow{\mathbf{h}}_i(\mathbf{v}_{>i}))}{\sum_{w'}\exp(\overleftarrow{b}_{w'} + \mathbf{U}_{w',:}\overleftarrow{\mathbf{h}}_i(\mathbf{v}_{>i}))} \tag{7}$$

for $i \in [1, ..., D]$ where $\overrightarrow{\mathbf{b}} \in \mathbb{R}^K$ and $\overleftarrow{\mathbf{b}} \in \mathbb{R}^K$ are biases in forward and backward passes, respectively. Note that the parameters $\mathbf{W}$ and $\mathbf{U}$ are shared between the two networks.

DocNADEe and iDocNADEe with Embedding priors: We introduce additional semantic information for each word into DocNADE-like models via its pre-trained embedding vector, thereby enabling better textual representations and semantically more coherent topic distributions, in particular for short texts. In its simplest form, we extend DocNADE with word embedding aggregation at each autoregressive step k to generate a complementary textual representation, i.e., $\sum_{k<i}\mathbf{E}_{:,v_k}$. This mechanism utilizes prior knowledge encoded in a pre-trained embedding matrix $\mathbf{E} \in \mathbb{R}^{H\times K}$ when learning task-specific matrices $\mathbf{W}$ and latent representations in DocNADE-like models. The position dependent forward $\overrightarrow{\mathbf{h}}^e_i(\mathbf{v}_{<i})$ and (only in iDocNADEe) backward $\overleftarrow{\mathbf{h}}^e_i(\mathbf{v}_{>i})$ hidden layers for each word i now depend on $\mathbf{E}$ as:

$$\overrightarrow{\mathbf{h}}^e_i(\mathbf{v}_{<i}) = g(\overrightarrow{\mathbf{c}} + \sum_{k<i}\mathbf{W}_{:,v_k} + \lambda\sum_{k<i}\mathbf{E}_{:,v_k}) \tag{8}$$

$$\overleftarrow{\mathbf{h}}^e_i(\mathbf{v}_{>i}) = g(\overleftarrow{\mathbf{c}} + \sum_{k>i}\mathbf{W}_{:,v_k} + \lambda\sum_{k>i}\mathbf{E}_{:,v_k}) \tag{9}$$

where, λ is a mixture coefficient, determined using validation set. As in equations 6 and 7, the forward and backward autoregressive conditionals are computed via hidden vectors $\overrightarrow{\mathbf{h}}^e_i(\mathbf{v}_{<i})$ and $\overleftarrow{\mathbf{h}}^e_i(\mathbf{v}_{>i})$, respectively.

Deep DocNADEs with/without Embedding Priors: DocNADE can be extended to a deep, multiple hidden layer architecture by adding new hidden layers as in a regular deep feed-forward neural network, allowing for improved performance (Lauly et al. 2017). In this deep version of DocNADE variants, the first hidden layers are computed in an analogous fashion to iDocNADE (eq. 4 and 5). Subsequent hidden layers are computed as:

$$\overrightarrow{\mathbf{h}}^{(d)}_i(\mathbf{v}_{<i}) = g(\overrightarrow{\mathbf{c}}^{(d)} + \mathbf{W}^{(d)} \cdot \overrightarrow{\mathbf{h}}^{(d-1)}_i(\mathbf{v}_{<i}))$$

and similarly, $\overleftarrow{\mathbf{h}}^{(d)}_i(\mathbf{v}_{>i})$ for $d = 2, ..., n$, where n is the total number of hidden layers. The exponent "(d)" is used as an index over the hidden layers and parameters in the deep feed-forward network. Forward and/or backward conditionals for each word i are modeled using the forward and backward hidden vectors at the last layer n. The deep DocNADE or iDocNADE variants without or with embeddings are named as *DeepDNE*, *iDeepDNE*, *DeepDNEe* and *iDeepDNEe*, respectively where $\mathbf{W}^{(1)}$ is the word representation matrix. However in *DeepDNEe* (or *iDeepDNEe*), we introduce embedding prior $\mathbf{E}$ in the first hidden layer, i.e.,

$$\overrightarrow{\mathbf{h}}^{e,(1)}_i = g(\overrightarrow{\mathbf{c}}^{(1)} + \sum_{k<i}\mathbf{W}^{(1)}_{:,v_k} + \lambda\sum_{k<i}\mathbf{E}_{:,v_k})$$

for each word i via embedding aggregation of its context $\mathbf{v}_{<i}$ (and $\mathbf{v}_{>i}$). Similarly, we compute $\overleftarrow{\mathbf{h}}^{e,(1)}_i$.

Learning: Similar to DocNADE, the conditionals $p(v_i = w|\mathbf{v}_{<i})$ and $p(v_i = w|\mathbf{v}_{>i})$ in DocNADEe, iDocNADE or iDocNADEe are computed by a neural network for each

Algorithm 1 Computation of $\log p(\mathbf{v})$ in *iDocNADE* or *iDocNADEe* using *tree-softmax* or *full-softmax*

Input: A training document vector $\mathbf{v}$, Embedding matrix $\mathbf{E}$
Parameters: $\{\overrightarrow{\mathbf{b}}, \overleftarrow{\mathbf{b}}, \overrightarrow{\mathbf{c}}, \overleftarrow{\mathbf{c}}, \mathbf{W}, \mathbf{U}\}$
Output: $\log p(\mathbf{v})$

$\overrightarrow{\mathbf{a}} \leftarrow \overrightarrow{\mathbf{c}}$
if `iDocNADE` **then**
  $\overleftarrow{\mathbf{a}} \leftarrow \overleftarrow{\mathbf{c}} + \sum_{i>1}\mathbf{W}_{:,v_i}$
if `iDocNADEe` **then**
  $\overleftarrow{\mathbf{a}} \leftarrow \overleftarrow{\mathbf{c}} + \sum_{i>1}\mathbf{W}_{:,v_i} + \lambda\sum_{i>1}\mathbf{E}_{:,v_i}$
$q(\mathbf{v}) = 1$
for i from 1 to D **do**
  $\overrightarrow{\mathbf{h}}_i \leftarrow g(\overrightarrow{\mathbf{a}})$; $\overleftarrow{\mathbf{h}}_i \leftarrow g(\overleftarrow{\mathbf{a}})$
  if `tree-softmax` **then**
    $p(v_i|\mathbf{v}_{<i}) = 1$; $p(v_i|\mathbf{v}_{>i}) = 1$
    for m from 1 to $|\pi(v_i)|$ **do**
      $p(v_i|\mathbf{v}_{<i}) \leftarrow p(v_i|\mathbf{v}_{<i})p(\pi(v_i)_m|\mathbf{v}_{<i})$
      $p(v_i|\mathbf{v}_{>i}) \leftarrow p(v_i|\mathbf{v}_{>i})p(\pi(v_i)_m|\mathbf{v}_{>i})$
  if `full-softmax` **then**
    compute $p(v_i|\mathbf{v}_{<i})$ using equation 6
    compute $p(v_i|\mathbf{v}_{>i})$ using equation 7
  $q(\mathbf{v}) \leftarrow q(\mathbf{v})p(v_i|\mathbf{v}_{<i})p(v_i|\mathbf{v}_{>i})$
  if `iDocNADE` **then**
    $\overrightarrow{\mathbf{a}} \leftarrow \overrightarrow{\mathbf{a}} + \mathbf{W}_{:,v_i}$; $\overleftarrow{\mathbf{a}} \leftarrow \overleftarrow{\mathbf{a}} - \mathbf{W}_{:,v_i}$
  if `iDocNADEe` **then**
    $\overrightarrow{\mathbf{a}} \leftarrow \overrightarrow{\mathbf{a}} + \mathbf{W}_{:,v_i} + \lambda\,\mathbf{E}_{:,v_i}$
    $\overleftarrow{\mathbf{a}} \leftarrow \overleftarrow{\mathbf{a}} - \mathbf{W}_{:,v_i} - \lambda\,\mathbf{E}_{:,v_i}$
$\log p(\mathbf{v}) \leftarrow \frac{1}{2}\log q(\mathbf{v})$

word v_i, allowing efficient learning of *informed* representations $\overrightarrow{\mathbf{h}}_i$ and $\overleftarrow{\mathbf{h}}_i$ (or $\overrightarrow{\mathbf{h}}_i^e(\mathbf{v}_{<i})$ and $\overleftarrow{\mathbf{h}}_i^e(\mathbf{v}_{>i})$), as it consists simply of a linear transformation followed by a non-linearity. Observe that the weight $\mathbf{W}$ (or prior embedding matrix $\mathbf{E}$) is the same across all conditionals and ties contextual observables (blue colored lines in Figure 1) by computing each $\overrightarrow{\mathbf{h}}_i$ or $\overleftarrow{\mathbf{h}}_i$ (or $\overrightarrow{\mathbf{h}}_i^e(\mathbf{v}_{<i})$ and $\overleftarrow{\mathbf{h}}_i^e(\mathbf{v}_{>i})$).

Binary word tree (`tree-softmax`) to compute conditionals: To compute the likelihood of a document, the autoregressive conditionals $p(v_i = w|\mathbf{v}_{<i})$ and $p(v_i = w|\mathbf{v}_{>i})$ have to be computed for each word $i \in [1, 2, ...D]$, requiring time linear in vocabulary size K. To reduce computational cost and achieve a complexity logarithmic in K we follow Larochelle and Lauly (2012) and decompose the computation of the conditionals using a probabilistic tree. All words in the documents are randomly assigned to a different leaf in a binary tree and the probability of a word is computed as the probability of reaching its associated leaf from the root. Each left/right transition probability is modeled using a binary logistic regressor with the hidden layer $\overrightarrow{\mathbf{h}}_i$ or $\overleftarrow{\mathbf{h}}_i$ ($\overrightarrow{\mathbf{h}}_i^e$ or $\overleftarrow{\mathbf{h}}_i^e$) as its input. In the binary tree, the probability of a given word is computed by multiplying each of the left/right transition probabilities along the tree path.

Algorithm 1 shows the computation of $\log p(\mathbf{v})$ using *iDocNADE* (or *iDocNADEe*) structure, where the autogressive conditionals (lines 14 and 15) for each word v_i are obtained from the forward and backward networks and modeled into a binary word tree, where $\pi(v_i)$ denotes the sequence of binary left/right choices at the internal nodes along the tree path and $\mathbf{l}(v_i)$ the sequence of tree nodes on that tree path. For instance, $l(v_i)_1$ will always be the root of the binary tree and $\pi(v_i)_1$ will be 0 if the word leaf v_i is in the left subtree or 1 otherwise. Therefore, each of the forward and backward conditionals are computed as:

$$p(v_i = w|\mathbf{v}_{<i}) = \prod_{m=1}^{|\pi(v_i)|} p(\pi(v_i)_m|\mathbf{v}_{<i})$$

$$p(v_i = w|\mathbf{v}_{>i}) = \prod_{m=1}^{|\pi(v_i)|} p(\pi(v_i)_m|\mathbf{v}_{>i})$$

$$p(\pi(v_i)_m|\mathbf{v}_{<i}) = g(\overrightarrow{b}_{l(v_i)_m} + \mathbf{U}_{l(v_i)_m,:}\overrightarrow{\mathbf{h}}(\mathbf{v}_{<i}))$$

$$p(\pi(v_i)_m|\mathbf{v}_{>i}) = g(\overleftarrow{b}_{l(v_i)_m} + \mathbf{U}_{l(v_i)_m,:}\overleftarrow{\mathbf{h}}(\mathbf{v}_{>i}))$$

where $\mathbf{U} \in \mathbb{R}^{T\times H}$ is the matrix of logistic regressions weights, T is the number of internal nodes in binary tree, and $\overrightarrow{\mathbf{b}}$ and $\overleftarrow{\mathbf{b}}$ are bias vectors.

Each of the forward and backward conditionals $p(v_i = w|\mathbf{v}_{<i})$ or $p(v_i = w|\mathbf{v}_{>i})$ requires the computation of its own hidden layers $\overrightarrow{\mathbf{h}}_i(\mathbf{v}_{<i})$ and $\overleftarrow{\mathbf{h}}_i(\mathbf{v}_{>i})$ (or $\overrightarrow{\mathbf{h}}_i^e(\mathbf{v}_{<i})$ and $\overleftarrow{\mathbf{h}}_i^e(\mathbf{v}_{>i})$), respectively. With H being the size of each hidden layer and D the number of words in $\mathbf{v}$, computing a single layer requires $O(HD)$, and since there are D hidden layers to compute, a naive approach for computing all hidden layers would be in $O(D^2H)$. However, since the weights in the matrix $\mathbf{W}$ are tied, the linear activations $\overrightarrow{\mathbf{a}}$ and $\overleftarrow{\mathbf{a}}$ (algorithm 1) can be re-used in every hidden layer and computational complexity reduces to $O(HD)$.

Algorithm 2 Computing gradients of $-\log p(\mathbf{v})$ in *iDocNADE* or *iDocNADEe* using *tree-softmax*

Input: A training document vector $\mathbf{v}$
Parameters: $\{\overrightarrow{\mathbf{b}}, \overleftarrow{\mathbf{b}}, \overrightarrow{\mathbf{c}}, \overleftarrow{\mathbf{c}}, \mathbf{W}, \mathbf{U}\}$
Output: $\delta\overrightarrow{\mathbf{b}}, \delta\overleftarrow{\mathbf{b}}, \delta\overrightarrow{\mathbf{c}}, \delta\overleftarrow{\mathbf{c}}, \delta\mathbf{W}, \delta\mathbf{U}$

1: $\overrightarrow{\mathbf{a}} \leftarrow 0$; $\overleftarrow{\mathbf{a}} \leftarrow 0$; $\overrightarrow{\mathbf{c}} \leftarrow 0$; $\overleftarrow{\mathbf{c}} \leftarrow 0$; $\overrightarrow{\mathbf{b}} \leftarrow 0$; $\overleftarrow{\mathbf{b}} \leftarrow 0$
2: **for** i from D to 1 **do**
3: $\quad\delta\overrightarrow{\mathbf{h}}_i \leftarrow 0$; $\delta\overleftarrow{\mathbf{h}}_i \leftarrow 0$
4: $\quad$**for** m from 1 to $|\pi(v_i)|$ **do**
5: $\quad\quad\overrightarrow{b}_{l(v_i)_m} \leftarrow \overrightarrow{b}_{l(v_i)_m} + (p(\pi(v_i)_m|\mathbf{v}_{<i}) - \pi(v_i)_m)$
6: $\quad\quad\overleftarrow{b}_{l(v_i)_m} \leftarrow \overleftarrow{b}_{l(v_i)_m} + (p(\pi(v_i)_m|\mathbf{v}_{>i}) - \pi(v_i)_m)$
7: $\quad\quad\delta\overrightarrow{\mathbf{h}}_i \leftarrow \delta\overrightarrow{\mathbf{h}}_i + (p(\pi(v_i)_m|\mathbf{v}_{<i}) - \pi(v_i)_m)\mathbf{U}_{l(v_i)_m,:}$
8: $\quad\quad\delta\overleftarrow{\mathbf{h}}_i \leftarrow \delta\overleftarrow{\mathbf{h}}_i + (p(\pi(v_i)_m|\mathbf{v}_{>i}) - \pi(v_i)_m)\mathbf{U}_{l(v_i)_m,:}$
9: $\quad\quad\delta\mathbf{U}_{l(v_i)_m} \leftarrow \delta\mathbf{U}_{l(v_i)_m} + (p(\pi(v_i)_m|\mathbf{v}_{<i}) - \pi(v_i)_m)\overrightarrow{\mathbf{h}}_i^T + (p(\pi(v_i)_m|\mathbf{v}_{>i}) - \pi(v_i)_m)\overleftarrow{\mathbf{h}}_i^T$
10: $\quad\delta\overrightarrow{\mathbf{g}} \leftarrow \overrightarrow{\mathbf{h}}_i \circ (1 - \overrightarrow{\mathbf{h}}_i)$ # for sigmoid activation
11: $\quad\delta\overleftarrow{\mathbf{g}} \leftarrow \overleftarrow{\mathbf{h}}_i \circ (1 - \overleftarrow{\mathbf{h}}_i)$ # for sigmoid activation
12: $\quad\delta\overrightarrow{\mathbf{c}} \leftarrow \delta\overrightarrow{\mathbf{c}} + \delta\overrightarrow{\mathbf{h}}_i \circ \delta\overrightarrow{\mathbf{g}}$; $\delta\overleftarrow{\mathbf{c}} \leftarrow \delta\overleftarrow{\mathbf{c}} + \delta\overleftarrow{\mathbf{h}}_i \circ \delta\overleftarrow{\mathbf{g}}$
13: $\quad\delta\mathbf{W}_{:,v_i} \leftarrow \delta\mathbf{W}_{:,v_i} + \delta\overrightarrow{\mathbf{a}} + \delta\overleftarrow{\mathbf{a}}$
14: $\quad\delta\overrightarrow{\mathbf{a}} \leftarrow \delta\overrightarrow{\mathbf{a}} + \delta\overrightarrow{\mathbf{h}}_i \circ \delta\overrightarrow{\mathbf{g}}$; $\delta\overleftarrow{\mathbf{a}} \leftarrow \delta\overleftarrow{\mathbf{a}} + \delta\overleftarrow{\mathbf{h}}_i \circ \delta\overleftarrow{\mathbf{g}}$

With the trained *iDocNADEe* (or *DocNADE* variants), the representation ($\overleftrightarrow{\mathbf{h}}^e \in \mathbb{R}^H$) for a new document $\mathbf{v}^*$ of size D^* is extracted by summing the hidden representations from the forward and backward networks to account for the context information around each word in the words' sequence, as

$$\overrightarrow{\mathbf{h}}^e(\mathbf{v}^*) = g(\overrightarrow{\mathbf{c}} + \sum_{k\leqslant D^*}\mathbf{W}_{:,v_k^*} + \lambda\sum_{k\leqslant D^*}\mathbf{E}_{:,v_k^*}) \quad (10)$$

$$\overleftarrow{\mathbf{h}}^e(\mathbf{v}^*) = g(\overleftarrow{\mathbf{c}} + \sum_{k\geqslant 1}\mathbf{W}_{:,v_k^*} + \lambda\sum_{k\geqslant 1}\mathbf{E}_{:,v_k^*}) \quad (11)$$

$$\text{Therefore; } \overleftrightarrow{\mathbf{h}}^e = \overrightarrow{\mathbf{h}}^e(\mathbf{v}^*) + \overleftarrow{\mathbf{h}}^e(\mathbf{v}^*) \quad (12)$$

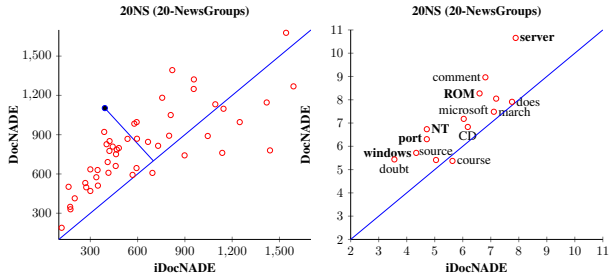
The DocNADE variants without embeddings compute the representation $\overleftrightarrow{\mathbf{h}}$ excluding the embedding term $\mathbf{E}$. Parameters $\{\overrightarrow{\mathbf{b}}, \overleftarrow{\mathbf{b}}, \overrightarrow{\mathbf{c}}, \overleftarrow{\mathbf{c}}, \mathbf{W}, \mathbf{U}\}$ are learned by minimizing the average negative log-likelihood of the training documents using stochastic gradient descent (algorithm 2). In our proposed formulation of iDocNADE or its variants (Figure 1), we perform inference by computing $\mathcal{L}^{iDocNADE}(\mathbf{v})$ (Eq.3).

Evaluation

We perform evaluations on 15 (8 short-text and 7 long-text) datasets of varying size with single/multi-class labeled documents from public as well as industrial corpora. See the *supplementary material* for the data description, hyper-parameters and grid-search results for generalization and IR tasks. Table 1 shows the data statistics, where 20NS: 20NewsGroups and R21578: Reuters21578. Since, Gupta et al. (2018a) have shown that DocNADE outperforms gaussian-LDA (Das, Zaheer, and Dyer 2015), glove-LDA and glove-DMM (Nguyen et al. 2015) in terms of topic coherence, text retrieval and classification, therefore we adopt DocNADE as the strong *baseline*. We use the development (dev) sets of each of the datasets to perform a grid-search on mixture weights, $\lambda = [0.1, 0.5, 1.0]$.

Data	Train	Val	Test	K	L	C	Domain	Tree-Softmax(TS)				Full-Softmax (FS)							
								DocNADE		iDocNADE		DocNADE		iDocNADE		DocNADEe		iDocNADEe	
								PPL	IR	PPL	IR	PPL	IR	PPL	IR	PPL	IR	PPL	IR
20NSshort	1.3k	0.1k	0.5k	2k	13.5	20	News	894	.23	880	.30	646	.25	639	.26	638	.28	633	**.28**
TREC6	5.5k	0.5k	0.5k	2k	9.8	6	Q&A	42	.48	39	.55	64	.54	61	.56	62	.56	60	**.57**
R21578title†	7.3k	0.5k	3.0k	2k	7.3	90	News	298	.61	239	.63	193	.61	181	.62	179	.65	176	**.66**
Subjectivity	8.0k	.05k	2.0k	2k	23.1	2	Senti	303	.78	287	.81	371	.77	365	.80	362	.80	361	**.81**
Polarity	8.5k	.05k	2.1k	2k	21.0	2	Senti	311	.51	292	.54	358	.54	345	.56	341	.56	340	**.57**
TMNtitle	22.8k	2.0k	7.8k	2k	4.9	7	News	863	.57	823	.59	711	.44	670	.46	668	.54	664	**.55**
TMN	22.8k	2.0k	7.8k	2k	19	7	News	548	.64	536	.66	592	.60	560	.64	563	.64	561	**.66**
AGnewstitle	118k	2.0k	7.6k	5k	6.8	4	News	811	.59	793	.65	545	.62	516	.64	516	.66	514	**.68**
Avg (short)								509	.55	486	.59	435	.54	417	.57	416	.58	413	**.60**
20NSsmall	0.4k	0.2k	0.2k	2k	187	20	News	-	-	-	-	628	.30	592	.32	607	**.33**	590	**.33**
Reuters8	5.0k	0.5k	2.2k	2k	102	8	News	172	.88	152	.89	184	.83	178	.88	178	**.87**	178	**.87**
20NS	8.9k	2.2k	7.4k	2k	229	20	News	830	.27	812	.33	474	.20	463	.24	464	**.25**	463	**.25**
R21578†	7.3k	0.5k	3.0k	2k	128	90	News	215	.70	179	.74	297	.70	285	**.73**	286	.71	285	.72
RCV1V2†	23.0k	.05k	10.0k	2k	123	103	News	381	.81	364	.86	479	.86	463	**.89**	465	.87	462	.88
SiROBs†	27.0k	1.0k	10.5k	3k	39	22	Industry	398	.31	351	.35	399	.34	340	.34	343	**.37**	340	.36
AGNews	118k	2.0k	7.6k	5k	38	4	News	471	.72	441	.77	451	.71	439	.78	433	.76	438	**.79**
Avg (long)								417	.61	383	.65	416	.56	394	**.60**	396	**.60**	393	**.60**
Avg (all)								469	.57	442	.62	426	.54	406	.58	407	.59	404	**.60**

Table 1: *Data statistics* of short and long texts as well as small and large corpora from various domains. *State-of-the-art* comparison in terms of PPL and IR (i.e, IR-precision) for **short** and **long** text datasets. The symbols are- L: average text length in number of words, K:dictionary size, C: number of classes, Senti: Sentiment, Avg: average, 'k':thousand and †: multi-label data. PPL and IR (IR-precision) are computed over 200 ($T200$) topics at retrieval fraction = 0.02. For short-text, $L < 25$. The *underline* and **bold** numbers indicate the best scores in PPL and retrieval task, respectively in FS setting. See Larochelle and Lauly (2012) for LDA (Blei, Ng, and Jordan 2003) performance in terms of PPL, where DocNADE outperforms LDA.



(a) PPL: 20NS (b) NLL of 14 Words

Figure 2: (a) PPL (T200) by iDocNADE and DocNADE for each of the 50 held-out documents of 20NS. The *filled circle* points to the document for which *PPL* differs by maximum. (b) NLL of each of the words in the document marked by the *filled circle* in (a), due to iDocNADE and DocNADE.

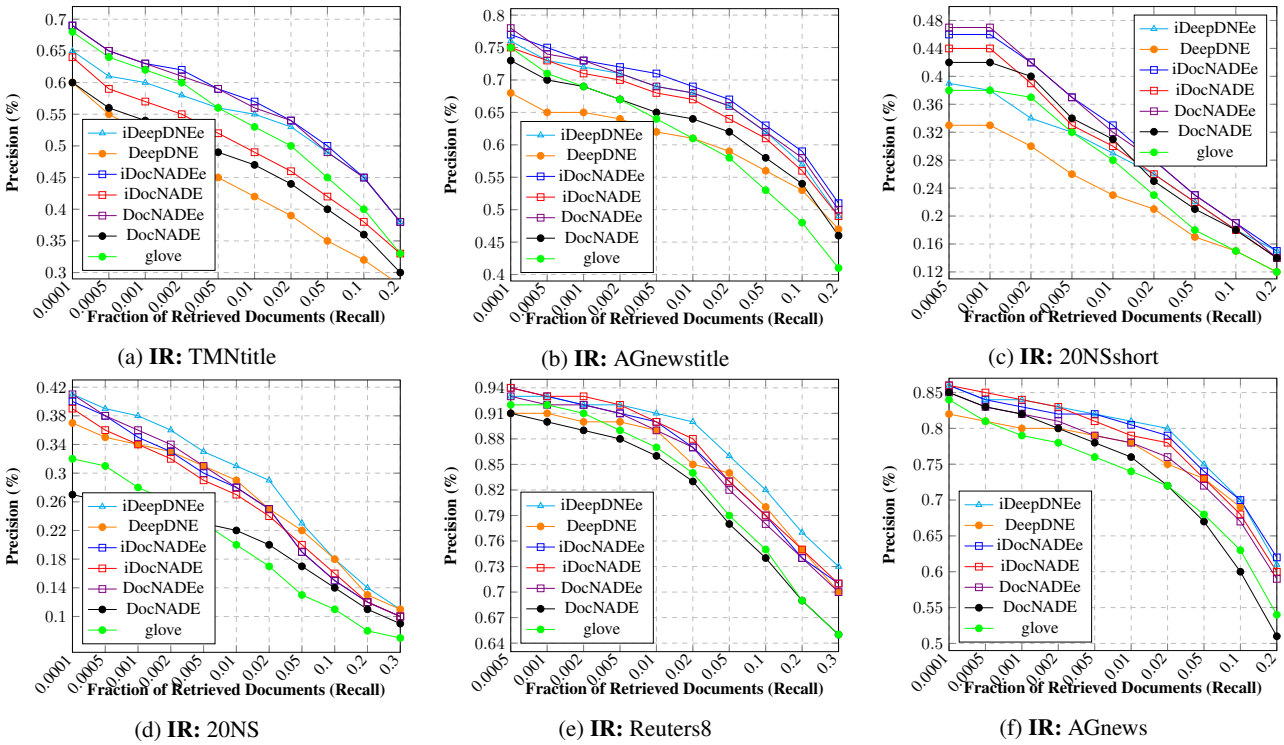
Generalization (Perplexity, PPL) We evaluate the topic models' generative performance as a generative model of documents by estimating log-probability for the test documents. During training, we initialize the proposed DocNADE extensions with DocNADE, i.e., **W** matrix. A comparison is made with the *baselines* (DocNADE and DeepDNE) and proposed variants (iDocNADE, DocNADEe, iDocNADEe, iDeepDNE, DeepDNEe and iDeepDNEe) using 50 (in *supplementary*) and 200 (T200) topics, set by the hidden layer size H.

Quantitative: Table 1 shows the average held-out perplexity (PPL) per word as, $PPL = \exp\Big(-\frac{1}{N}\sum_{t=1}^{N}\frac{1}{|\mathbf{v}^t|}\log p(\mathbf{v}^t)\Big)$ where N and $|\mathbf{v}^t|$ are the total number of documents and words in a document $\mathbf{v}^t$. To compute PPL, the log-likelihood of the document $\mathbf{v}^t$, i.e., $\log p(\mathbf{v}^t)$, is obtained by $\mathcal{L}^{DocNADE}$ (eqn. 2) in the DocNADE (forward only) variants, while we average PPL scores from the forward and backward networks of the iDocNADE variants.

Table 1 shows that the proposed models achieve lower perplexity for both the short-text (413 vs 435) and long-text (393 vs 416) datasets than *baseline* DocNADE with full-softmax (or tree-softmax). In total, we show a gain of 5.2% (404 vs 426) in PPL score on an average over the 15 datasets.

Table 2 illustrates the generalization performance of deep variants, where the proposed extensions outperform the DeepDNE for both short-text and long-text datasets. We report a gain of 10.7% (402 vs 450) in PPL due to iDeepDNEe over the baseline DeepDNE, on an average over 11 datasets.

Inspection: We quantify the use of context information in learning informed document representations. For 20NS dataset, we randomly select 50 held-out documents from its test set and compare (Figure 2a) the *PPL* for each of the held-out documents under the learned 200-dimensional DocNADE and iDocNADE. Observe that iDocNADE achieves lower *PPL* for the majority of the documents. The *filled* circle(s) points to the document for which *PPL* differs by a maximum between iDocNADE and DocNADE. We select the corresponding document and compute the negative log-likelihood (*NLL*) for every word. Figure 2b shows that the *NLL* for the majority of the words is lower (better) in iDocNADE than DocNADE. See the *supplementary material* for the raw text of the selected documents.



(a) **IR:** TMNtitle (b) **IR:** AGnewstitle (c) **IR:** 20NSshort

(d) **IR:** 20NS (e) **IR:** Reuters8 (f) **IR:** AGnews

Figure 3: Document retrieval performance (IR-precision) on 3 short-text and 3 long-text datasets at different retrieval fractions

Interpretability (Topic Coherence) Beyond PPL, we compute topic coherence (Chang et al. 2009; Newman, Karimi, and Cavedon 2009; Das, Zaheer, and Dyer 2015; Gupta et al. 2018b) to assess the meaningfulness of the underlying topics captured. We choose the coherence measure proposed by Röder, Both, and Hinneburg (2015) that identifies context features for each topic word using a sliding window over the reference corpus. The higher scores imply more coherent topics.

Quantitative: We use gensim module (*coherence type* = c_v) to estimate coherence for each of the 200 topics (top 10 and 20 words). Table 3 shows average coherence over 200 topics using short-text and long-text datasets, where the high scores for long-text in iDocNADE (.636 vs .602) suggest that the contextual information helps in generating more coherent topics than DocNADE. On top, the introduction of embeddings, i.e., iDocNADEe for short-text boosts (.847 vs .839) topic coherence. **Qualitative:** Table 5 illustrates example topics each with a coherence score.

Applicability (Document Retrieval) To evaluate the quality of the learned representations, we perform a document retrieval task using the 15 datasets and their label information. We use the experimental setup similar to Lauly et al. (2017), where all test documents are treated as queries to retrieve a fraction of the closest documents in the original training set using cosine similarity measure between their representations (eqn. 12 in iDocNADE and $\overrightarrow{\mathbf{h}}_D$ in DocNADE). To compute retrieval precision for each fraction (e.g., 0.0001, 0.005, 0.01, 0.02, 0.05, 0.1, 0.2, etc.), we average the number of retrieved training documents with the same label as the query. For multi-label datasets, we average the precision scores over multiple labels for each query. Since Salakhutdinov and Hinton (2009) and Lauly et al. (2017) showed that RSM and DocNADE strictly outperform LDA on this task, we only compare DocNADE and its proposed extensions.

Table 1 shows the IR-precision scores at retrieval fraction 0.02. Observe that the introduction of both pre-trained

data	DeepDNE		iDeepDNE		DeepDNEe		iDeepDNEe	
	PPL	IR	PPL	IR	PPL	IR	PPL	IR
20NSshort	917	.21	841	.22	827	.25	830	**.26**
TREC6	114	.50	69	.52	69	**.55**	68	**.55**
R21578title	253	.50	231	.52	236	**.63**	230	.61
Subjectivity	428	.77	393	.77	392	.81	392	**.82**
Polarity	408	.51	385	.51	383	**.55**	387	.53
TMN	681	.60	624	.62	627	.63	623	**.66**
Avg (short)	467	.51	424	.53	422	**.57**	421	**.57**
Reuters8	216	.85	192	.89	191	.88	191	**.90**
20NS	551	.25	504	.28	504	**.29**	506	**.29**
R21578	318	.71	299	**.73**	297	.72	298	**.73**
AGNews	572	.75	441	.77	441	.75	440	**.80**
RCV1V2	489	.86	464	.88	466	**.89**	462	**.89**
Avg (long)	429	.68	380	.71	379	.71	379	**.72**
Avg (all)	450	.59	404	.61	403	.63	402	**.64**

Table 2: Deep Variants (+ Full-softmax) with T200: PPL and IR (i.e, IR-precision) for **short** and **long** text datasets.

model	DocNADE		iDocNADE		DocNADEe		iDocNADEe	
	W10	W20	W10	W20	W10	W20	W10	W20
20NSshort	.744	.849	.748	.852	.747	.851	.744	.849
TREC6	.746	.860	.748	.864	.753	.858	.752	.866
R21578title	.742	.845	.748	.855	.749	.859	.746	.856
Polarity	.730	.833	.732	.837	.734	.839	.738	.841
TMNtitle	.738	.840	.744	.848	.746	.850	.746	.850
TMN	.709	.811	.713	.814	.717	.818	.721	.822
Avg (short)	.734	.839	.739	.845	**.742**	.846	.741	**.847**
20NSsmall	.515	.629	.564	.669	.533	.641	.549	.661
Reuters8	.578	.665	.564	.657	.574	.655	.554	.641
20NS	.417	.496	.453	.531	.385	.458	.417	.490
R21578	.540	.570	.548	.640	.542	.596	.551	.663
AGnews	.718	.828	.721	.840	.677	.739	.696	.760
RCV1V2	.383	.426	.428	.480	.364	.392	.420	.463
Avg (long)	.525	.602	**.546**	**.636**	.513	.580	.531	.613

Table 3: Topic coherence with the top 10 (W10) and 20 (W20) words from topic models (T200). Since, (Gupta et al. 2018a) have shown that DocNADE outperforms both glove-DMM and glove-LDA, therefore DocNADE as the baseline.

data	glove		doc2vec		DocNADE		DocNADEe		iDocNADE		iDocNADEe	
	F1	acc	F1	acc	F1	acc	F1	acc	F1	acc	F1	acc
20NSshort	.493	.520	.413	.457	.428	.474	.473	.529	.456	.491	.518	.535
TREC6	.798	.810	.400	.512	.804	.822	.854	.856	.808	.812	.842	.844
R21578title	.356	.695	.176	.505	.318	.653	.352	.693	.302	.665	.335	.700
Subjectivity	.882	.882	.763	.763	.872	.872	.886	.886	.871	.871	.886	.886
Polarity	.715	.715	.624	.624	.693	.693	.712	.712	.688	.688	.714	.714
TMNtitle	.693	.727	.582	.617	.624	.667	.697	.732	.632	.675	.696	.731
TMN	.736	.755	.720	.751	.740	.778	.765	.801	.751	.790	.771	.805
AGnewstitle	.814	.815	.513	.515	.812	.812	.829	.828	.819	.818	.829	.828
Avg (short)	.685	.739	.523	.593	.661	.721	.696	.755	.666	.726	**.700**	**.756**
Reuters8	.830	.950	.937	.852	.753	.931	.848	.956	.836	.957	.860	.960
20NS	.509	.525	.396	.409	.512	.535	.514	.540	.524	.548	.523	.544
R21578	.316	.703	.215	.622	.324	.716	.322	.721	.350	.710	.300	.722
AGnews	.870	.871	.713	.711	.873	.876	.880	.880	.880	.880	.886	.886
RCV1V2	.442	.368	.442	.341	.461	.438	.460	.457	.463	.452	.465	.454
Avg (long)	.593	.683	.540	.587	.584	.699	.605	.711	**.611**	.710	.607	**.713**
Avg (all)	.650	.718	.530	.590	.631	.712	.661	.738	.645	.720	**.664**	**.740**

Table 4: Text classification for short and long texts with T200 or word embedding dimension (Topic models with FS)

embedding priors and contextual information leads to improved performance on the IR task for short-text and long-text datasets. We report a gain of 11.1% (.60 vs .54) in precision on an average over the 15 datasets, compared to DocNADE. On top, the deep variant i.e. iDeepDNEe (Table 2) demonstrates a gain of 8.5% (.64 vs .59) in precision over the 11 datasets, compared to DeepDNE. Figures (3a, 3b, 3c) and (3d, 3e and 3f) illustrate the average precision for the retrieval task on short-text and long-text datasets, respectively.

Applicability (Text Categorization) Beyond the document retrieval, we perform text categorization to measure the quality of word vectors learned in the topic models. We consider the same experimental setup as in the document retrieval task and extract the document representation (latent vector) of 200 dimension for each document (or text), learned during the training of DocNADE variants. To perform document categorization, we employ a logistic regression classifier with $L2$ regularization. We also compute document representations from pre-trained glove (Pennington, Socher, and Manning 2014) embedding matrix by summing the word vectors and compute classification performance. On top, we also extract document representation from doc2vec (Le and Mikolov 2014).

Table 4 shows that *glove* leads DocNADE in classification performance, suggesting a need for distributional priors. For short-text dataset, iDocNADEe (and DocNADEe) outperforms *glove* (.700 vs .685) and DocNADE (.700 vs .661) in F1. Overall, we report a gain of 5.2% (.664 vs .631) in F1 due to iDocNADEe over DocNADE for classification on an average over 13 datasets.

Inspection of Learned Representations: To analyze the meaningful semantics captured, we perform a qualitative inspection of the learned representations by the topic models. Table 5 shows topics for 20NS dataset that could be interpreted as *religion*, which are (sub)categories in the data, confirming that meaningful topics are captured. Observe that DocNADEe extracts a more coherent topic.

For word level inspection, we extract *word representations* using the columns $W_{:,v_i}$ as the vector (200 dimension) representation of each word v_i, learned by iDocNADE using 20NS dataset. Table 6 shows the five nearest neighbors of some selected words in this space and their corresponding similarity scores. We also compare similarity in word vectors from iDocNADE and glove embeddings, confirming that meaningful word representations are learned.

DocNADE	iDocNADE	DocNADEe
beliefs, muslims, forward, alt, islam, towards, atheism, christianity, hands, opinions	scripture, atheists, sin, religions, christianity, lord, bible, msg, heaven, jesus	atheists, christianity, belief, eternal, atheism, catholic, bible, arguments, islam, religions
0.44	0.46	0.52

Table 5: Topics (top 10 words) of 20NS with coherence

book			jesus			windows			gun		
neighbors	s_i	s_g	*neighbors*	s_i	s_g	*neighbors*	s_i	s_g	*neighbors*	s_i	s_g
books	.61	.84	christ	.86	.83	dos	.74	.34	guns	.72	.79
reference	.52	.51	god	.78	.63	files	.63	.36	firearms	.63	.63
published	.46	.74	christians	.74	.49	version	.59	.43	criminal	.63	.33
reading	.45	.54	faith	.71	.51	file	.59	.36	crime	.62	.42
author	.44	.77	bible	.71	.51	unix	.52	.47	police	.61	.43

Table 6: 20NS dataset: The five nearest neighbors by iDocNADE. s_i: Cosine similarity between the word vectors from iDocNADE, for instance vectors of *jesus* and *god*. s_g: Cosine similarity in embedding vectors from glove.

Conclusion

We show that leveraging contextual information and introducing distributional priors via pre-trained word embeddings in our proposed topic models result in learning better word/document representation for short and long documents, and improve generalization, interpretability of topics and their applicability in text retrieval and classification.

References

Bengio, Y.; Ducharme, R.; Vincent, P.; and Jauvin, C. 2003. A neural probabilistic language model. In *Journal of Machine Learning Research 3*, 1137–1155.

Blei, D.; Ng, A.; and Jordan, M. 2003. Latent dirichlet allocation. 993–1022.

Chang, J.; Boyd-Graber, J.; Wang, C.; Gerrish, S.; and Blei., D. M. 2009. Reading tea leaves: How humans interpret topic models. In *In Neural Information Processing Systems (NIPS)*.

Das, R.; Zaheer, M.; and Dyer, C. 2015. Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics.

Elman, J. L. 1990. Finding structure in time. *Cognitive science* 14(2):179–211.

Gupta, P.; Chaudhary, Y.; Buettner, F.; and Schütze, H. 2018a. texttovec: Deep contextualized neural autoregressive models of language with distributed compositional prior. In *Preprint arxiv*.

Gupta, P.; Rajaram, S.; Schütze, H.; and Andrassy, B. 2018b. Deep temporal-recurrent-replicated-softmax for topical trends over time. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, 1079–1089. New Orleans, USA: Association of Computational Linguistics.

Gupta, P.; Schütze, H.; and Andrassy, B. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2537–2547.

Hinton, G. E. 2002. Training products of experts by minimizing contrastive divergence. In *Neural Computation*, 1771–1800.

Larochelle, H., and Lauly, S. 2012. A neural autoregressive topic model. In *Proceedings of the Advances in Neural Information Processing Systems 25 (NIPS 2012)*. NIPS.

Larochelle, H., and Murray, I. 2011. The neural autoregressive distribution estimato. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS 2011)*, 29–37. JMLR.

Lauly, S.; Zheng, Y.; Allauzen, A.; and Larochelle, H. 2017. Document neural autoregressive distribution estimation. *Journal of Machine Learning Research* 18(113):1–24.

Le, Q. V., and Mikolov, T. 2014. Distributed representations of sentences and documents. 1188–1196.

Manning, C. D., and Schütze, H. 1999. Foundations of statistical natural language processing. Cambridge MA: The MIT Press.

Mousa, A. E.-D., and Schuller, B. 2017. Contextual bidirectional long short-term memory recurrent neural network language models: A generative approach to sentiment analysis. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, 1023–1032. Association for Computational Linguistics.

Newman, D.; Karimi, S.; and Cavedon, L. 2009. External evaluation of topic models. In *Proceedings of the 14th Australasian Document Computing Symposium*.

Nguyen, D. Q.; Billingsley, R.; Du, L.; and Johnson, M. 2015. Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics* 3:299–313.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, 1532–1543.

Röder, M.; Both, A.; and Hinneburg, A. 2015. Exploring the space of topic coherence measures. In *Proceedings of the WSDM*. ACM.

Salakhutdinov, R., and Hinton, G. 2009. Replicated softmax: an undirected topic model. In *Proceedings of the Advances in Neural Information Processing Systems 22 (NIPS 2009)*, 1607–1614. NIPS.

Vu, N. T.; Adel, H.; Gupta, P.; and Schütze, H. 2016a. Combining recurrent and convolutional neural networks for relation classification. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 534–539. San Diego, California USA: Association for Computational Linguistics.

Vu, N. T.; Gupta, P.; Adel, H.; and Schütze, H. 2016b. Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *Proceedings of IEEE/ACM Trans. on Audio, Speech, and Language Processing (ICASSP)*. IEEE.

Zheng, Y.; Zhang, Y.-J.; and Larochelle, H. 2016. A deep and autoregressive approach for topic modeling of multimodal data. In *IEEE transactions on pattern analysis and machine intelligence*, 1056–1069. IEEE.

Chapter 7

Multi-view and Multi-source Transfers in Neural Topic Modeling

Multi-view and Multi-source Transfers in Neural Topic Modeling

Pankaj Gupta[1,2], Yatin Chaudhary[1], Hinrich Schütze[2]
[1]Corporate Technology, Machine-Intelligence (MIC-DE), Siemens AG Munich, Germany
[2]CIS, University of Munich (LMU) Munich, Germany
{pankaj.gupta, yatin.chaudhary}@siemens.com
pankaj.gupta@campus.lmu.de | inquiries@cislmu.org

Abstract

Though word embeddings and topics are complementary representations, several past works have used word embeddings in (neural) topic modeling to address data sparsity problem in short text or small collection of documents. In this paper, we propose an approach to jointly transfer the two representations (or views) in neural topic modeling to better deal with polysemy and data sparsity issues. Moreover, we identify multiple relevant source domains and take advantage of word and topic features to guide meaningful learning in the sparse target domain. We quantify the quality of topic and document representations via generalization (perplexity), interpretability (topic coherence) and information retrieval.

1 Introduction

Probabilistic topic models, such as LDA (Blei et al., 2003), Replicated Softmax (RSM) (Salakhutdinov and Hinton, 2009) and Document Neural Autoregressive Distribution Estimator (DocNADE) (Larochelle and Lauly, 2012) are often used to extract topics from text collections and learn latent document representations to perform natural language processing tasks, such as information retrieval (IR). Though they have been shown to be powerful in modeling large text corpora, the topic modeling (TM) still remains challenging especially in the sparse-data setting, e.g., on short text or a corpus of few documents.

Though word embeddings (Pennington et al., 2014) and topics are complementary in how they represent the meaning, they are distinctive in how they learn from word occurrences observed in text corpora. Word embeddings have *local* context (*view*) in the sense that they are learned based on local collocation pattern in a text corpus, where the representation of each word either depends on a local context window (Mikolov et al., 2013) or is a function of its sentence(s) (Peters et al., 2018). Consequently, the word occurrences are modeled in a *fine-granularity*. On other hand, a topic (Blei et al., 2003) has a *global* word context (*view*): TM infers topic distributions across documents in the corpus and assigns a topic to each word occurrence, where the assignment is equally dependent on all other words appearing in the same document. Therefore, it learns from word occurrences across documents and encodes a *coarse-granularity* description. Unlike topics, the word embeddings can not capture the thematic structures (topical semantics) in the underlying corpus.

Consider the following topics (Z_1-Z_4), where Z_1-Z_3 are respectively obtained from different (large) source ($\mathcal{S}^1$-$\mathcal{S}^3$) domains whereas Z_4 from the target domain $\mathcal{T}$ in the sparse-data setting:

Z_1 ($\mathcal{S}^1$): *profit, growth, stocks,* **apple**, *consumer, buy, billion, shares* $\rightarrow$ *Marketing/Trading*

Z_2($\mathcal{S}^2$): *smartphone, ipad,* **apple**, *app, iphone, devices, phone, tablet* $\rightarrow$ *Product Line*

Z_3 ($\mathcal{S}^3$): *microsoft, mac, linux, ibm, ios,* **apple**, *xp, windows* $\rightarrow$ *Operating System/Company*

Z_4 ($\mathcal{T}$): **apple**, *talk, computers, shares, disease, driver, electronics, profit, ios* $\rightarrow$?

Usually, top words associated with topics learned on a large corpus are semantically coherent, e.g., *Marketing, Product Line*, etc. However in sparse-data setting, topics (e.g., Z_4) are incoherent (*noisy*) and therefore, it is difficult to infer meaningful semantics. Additionally, notice that the word *apple* is topically/thematically contextualized (word-topic combination) in different semantics in $\mathcal{S}^1$-$\mathcal{S}^3$ and referring to a *company*. Unlike topics, the top-5 nearest neighbors (NN) of *apple* (below) in the embeddings (Mikolov et al., 2013) space suggest that it refers to a *fruit*.

apple $\overset{NN}{\Longrightarrow}$ *apples, pear, fruit, berry, pears, strawberry*

Motivation (1): Das et al. (2015); Nguyen et al. (2015); Gupta et al. (2019) have shown that TM can be improved by using external knowledge,

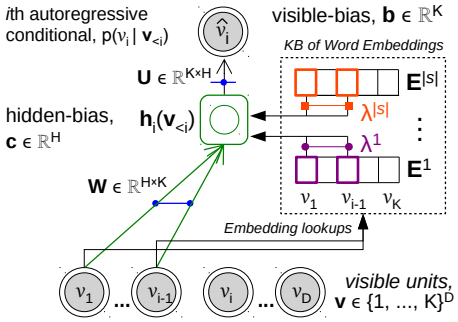


Figure 1: DocNADE+LVT: Introducing multi-source word embeddings in DocNADE at each autoregressive step i. Double circle → multinomial (softmax) unit

e.g., word embeddings especially for short text or small collections to alleviate sparsity issues. Since the word embeddings ignore the thematically/topically contextualized structures and therefore, can not deal with ambiguity. Additionally, knowledge transfer via word embeddings is vulnerable to negative transfer (Cao et al., 2010) on the target domain when domains are shifted.

To illustrate, consider a short-text document $\mathbf{v}$: [Apple gained its US market shares] in the target domain $\mathcal{T}$. Here[1], the word *apple* refers to a *company* and hence, the word vector of *apple* is an irrelevant source of knowledge transfer for both $\mathbf{v}$ and topic Z_4. In contrast, one can better model $\mathbf{v}$ and amend Z_4 for coherence, given meaningful representations Z_1-Z_3 via latent topic features.

Motivation (2): There are usually several word-topic associations in different domains, e.g., in topics Z_1-Z_3. Given a noisy topic Z_4 in $\mathcal{T}$ and meaningful topics Z_1-Z_3, we want to identify multiple relevant domains and take advantage of the representations (word and topic features) of $\mathcal{S}^1$-$\mathcal{S}^3$ to empower meaningful learning in $\mathcal{T}$.

Contribution(s): To better deal with polysemy and alleviate data-sparsity issues, we introduce an approach to transfer latent topic features (thematically contextualized) instead using word embeddings exclusively. Moreover, we learn word and topic representations on multiple source domains and then perform *multi-view* and *multi-source* knowledge transfers within neural topic modeling by jointly using the complementary representations. To do so, we guide the generative process of learning hidden topics of the target domain by word and latent topic features from a source domain(s) such that the hidden topics on the target get meaningful. *Code in supplementary*.

[1]TM ignores punctuation, capitalization, stop words, etc.

Algorithm 1 Computation of $\log p(\mathbf{v})$ and Loss $\mathcal{L}(\mathbf{v})$

Input: A target training document $\mathbf{v}$, $|\mathcal{S}|$ source domains
Input: KB of latent topics $\{\mathbf{Z}^1, ..., \mathbf{Z}^{|\mathcal{S}|}\}$
Input: KB of word embedding matrices $\{\mathbf{E}^1, ..., \mathbf{E}^{|\mathcal{S}|}\}$
Parameters: $\mathbf{\Theta} = \{\mathbf{b}, \mathbf{c}, \mathbf{W}, \mathbf{U}, \mathbf{A}^1, ..., \mathbf{A}^{|\mathcal{S}|}\}$
hyper-parameters: $\theta = \{\lambda^1, ..., \lambda^{|\mathcal{S}|}, \gamma^1, ..., \gamma^{|\mathcal{S}|}, H\}$
Initialize $\mathbf{a} \leftarrow \mathbf{c}$ and $p(\mathbf{v}) \leftarrow 1$
for i from 1 to D **do**
  $\mathbf{h}_i(\mathbf{v}_{<i}) \leftarrow g(\mathbf{a})$, where g = {sigmoid, tanh}
  $p(v_i = w|\mathbf{v}_{<i}) \leftarrow \frac{\exp(b_w + \mathbf{U}_{w,:}\mathbf{h}_i(\mathbf{v}_{<i}))}{\sum_{w'} \exp(b_{w'} + \mathbf{U}_{w',:}\mathbf{h}_i(\mathbf{v}_{<i}))}$
  $p(\mathbf{v}) \leftarrow p(\mathbf{v})p(v_i|\mathbf{v}_{<i})$
  compute pre-activation at step, i: $\mathbf{a} \leftarrow \mathbf{a} + \mathbf{W}_{:,v_i}$
  if LVT **then**
    get word embedding for v_i from source domain(s)
    $\mathbf{a} \leftarrow \mathbf{a} + \sum_{k=1}^{|\mathcal{S}|} \lambda^k \mathbf{E}^k_{:,v_i}$
$\mathcal{L}(\mathbf{v}) \leftarrow -\log p(\mathbf{v})$
if GVT **then**
  $\mathcal{L}(\mathbf{v}) \leftarrow \mathcal{L}(\mathbf{v}) + \sum_{k=1}^{|\mathcal{S}|} \gamma^k \sum_{j=1}^{H} ||\mathbf{A}^k_{j,:}\mathbf{W} - \mathbf{Z}^k_{j,:}||_2^2$

2 Knowledge Transfer in Topic Modeling

Consider a sparse target domain $\mathcal{T}$ and a set of $|\mathcal{S}|$ source domains $\mathcal{S}$, we first prepare two knowledge bases (KBs) of representations from each of the sources: (1) word embeddings matrices $\{\mathbf{E}^1, ..., \mathbf{E}^{|\mathcal{S}|}\}$, where $\mathbf{E}^k \in \mathbb{R}^{E \times K}$ and (2) latent topic features $\{\mathbf{Z}^1, ..., \mathbf{Z}^{|\mathcal{S}|}\}$, where $\mathbf{Z}^k \in \mathbb{R}^{H \times K}$ encodes a distribution over a vocabulary of K words. E and H are word embedding and latent topic dimensions, respectively. While TM on $\mathcal{T}$, we introduce two types of knowledge transfers: *Local* (LVT) and *Global* (GVT) *View Transfer* using the two KBs, respectively. *Notice* that a superscript indicates a source.

Neural Autoregressive Topic Model: Since DocNADE (Larochelle and Lauly, 2012; Gupta et al., 2019), a neural-network based topic model has shown to outperform traditional models, therefore we adopt it to perform knowledge transfer.

For a document $\mathbf{v} = (v_1, ..., v_D)$ of size D, each word index v_i takes value in $\{1, ..., K\}$ of vocabulary size K. DocNADE learns topics in a language modeling fashion (Bengio et al., 2003) and decomposes the joint probability distribution $p(\mathbf{v}) = \prod_{i=1}^{D} p(v_i|\mathbf{v}_{<i})$ such that each autoregressive conditional $p(v_i|\mathbf{v}_{<i})$ is modeled by a feed-forward neural network using preceding words $\mathbf{v}_{<i}$ in the sequence. For DocNADE, Figure 1 and Algorithm 1 (LVT and GVT set to *False*) demonstrate the computation of $\log p(\mathbf{v})$ and negative log-likelihood $\mathcal{L}(\mathbf{v})$ that is minimized using gradient descent. Importantly, we exploit properties of $\mathbf{W}$ in DocNADE that the column vector $\mathbf{W}_{:,v_i}$ corresponds to embedding of the word v_i, whereas

KBs from Source Corpus	Model/ Transfer Type	Scores on Target Corpus (*in sparse-data setting*)											
		20NSshort			TMNtitle			R21578title			20NSsmall		
		PPL	*COH*	*IR*	*PPL*	*COH*	*IR*	*PPL*	*COH*	*IR*	*PPL*	*COH*	*IR*
baselines	glove-DMM	-	.512	.183	-	.633	.445	-	.364	.273	-	.578	.090
	doc2vec	-	-	.090	-	-	.190	-	-	.518	-	-	.200
	DocNADE	646	.667	.290	706	.709	.521	192	.713	.657	594	.462	.270
	DocNADEe	**629**	.674	.294	680	.719	.541	187	.721	.663	**590**	.455	.274
20NS	LVT	630	.673	.298	705	.709	.523	194	.708	.656	594	.455	.288
	GVT	646	.690	.303	718	.720	.527	**184**	.698	.660	594	.500	.310
	MVT	638	.690	**.314**	714	.718	.528	188	.715	.655	600	.499	.311
	+ Glove	630	.700	.298	690	.733	.539	186	.724	.664	601	.499	.306
TMN	LVT	649	.668	.296	**655**	.731	.548	187	.703	.659	593	.460	.273
	GVT	661	.692	.294	689	.728	.555	191	.709	.660	596	.521	.276
	MVT	658	.687	.297	663	.747	.553	195	.720	.660	599	.507	.292
	+ Glove	640	.689	.295	673	.750	.542	186	.716	.662	599	.517	.261
R21578	LVT	656	.667	.292	704	.715	.522	186	.715	.676	593	.458	.267
	GVT	654	.672	.293	716	.719	.526	194	.706	.672	595	.485	.279
	MVT	650	.670	.296	716	.720	.528	194	.724	.676	599	.490	.280
	+ Glove	633	.691	.295	689	.734	.540	188	.734	.676	598	.485	.255
AGnews	LVT	650	.677	.297	682	.723	.533	185	.710	.659	593	.458	.260
	GVT	667	.695	.300	728	.735	.534	190	.717	.663	598	.563	.282
	MVT	659	.696	.290	718	.740	.533	189	.727	.659	599	.566	.279
	+ Glove	642	.707	.291	706	.745	.540	190	.734	.664	600	.573	.284
MST	LVT	640	.678	.308	663	.732	.547	186	.712	.673	596	.442	.277
	GVT	658	.705	.305	704	.746	.550	192	.727	.673	599	.585	**.326**
	MVT	656	**.721**	**.314**	680	**.752**	**.556**	188	**.737**	**.678**	600	**.600**	.285
	+ Glove	644	.719	.293	687	**.752**	.538	189	.732	.674	609	.586	.282

Table 1: State-of-the-art comparisons: Perplexity (PPL), topic coherence (COH) and precision (IR) at retrieval fraction 0.02. + Glove: MVT+Glove embeddings. *Please read column-wise*. **Bold**: best in column.

the row vector $\mathbf{W}_{j,:}$ encodes latent features for jth topic. Therefore, we use DocNADE to prepare KBs of $\mathbf{E}$ and $\mathbf{Z}$ using source domains $\mathcal{S}$.

Multi View (MVT) and Multi Source Transfers (MST): Illustrated in Figure 1 and Algorithm 1 with `LVT` = *True*, we perform knowledge transfer to $\mathcal{T}$ using word embeddings $\{\mathbf{E}^1, ..., \mathbf{E}^{|\mathcal{S}|}\}$ from several sources $\mathcal{S}$. Notice that λ^k is a weight for $\mathbf{E}^k$ that controls the amount of knowledge transferred in $\mathcal{T}$. Recently, DocNADEe (Gupta et al., 2019) has incorporated word embeddings in extending DocNADE though a single source.

Next, we perform knowledge transfer exclusively using latent topic features of $\mathcal{S}$ (Algorithm 1 when `GVT` = *True*). In doing so, we add a regularization term to the loss function $\mathcal{L}(\mathbf{v})$ and require DocNADE to minimize the overall loss in a way that the (latent) topic features in $\mathbf{W}$ simultaneously inherit relevant topical features from each of the source domains, and generate meaningful representations for the target $\mathcal{T}$. Consequently, the generative process of learning topic features in $\mathbf{W}$ is guided by relevant features in $\{\mathbf{Z}\}_1^{|\mathcal{S}|}$ to address data-sparsity. Here, $\mathbf{A}^k \in \mathbb{R}^{H \times H}$ aligns latent topics in $\mathcal{T}$ and kth source, and γ^k governs the degree of imitation of topic features $\mathbf{Z}^k$ by $\mathbf{W}$ in $\mathcal{T}$.

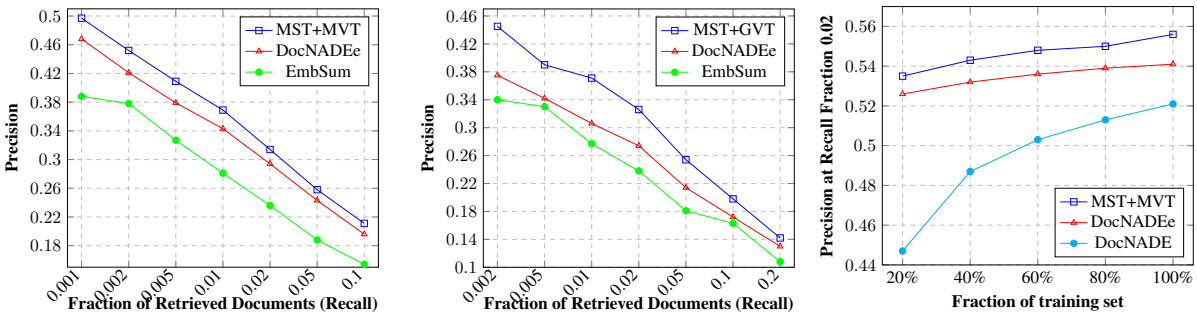
When `LVT` and `GVT` are *True* for many sources, the two complementary representations are jointly used in knowledge transfer and therefore, the name *multi-view* and *multi-source* transfers.

3 Evaluation and Analysis

Datasets: Our target domain $\mathcal{T}$ consists of 3 short-text (`20NSshort`, `TMNtitle` and `R21578title`) and a corpus (`20NSsmall`) of few documents. However in source $\mathcal{S}$, we use 4 large corpora (`20NS`, `TMN`, `R21578` and `AGnews`) in different label spaces. See the data description in *supplementary*.

Baselines: We consider topic models, e.g., (1) glove-DMM (Nguyen et al., 2015): LDA-based with word embedding (2) DocNADE: Neural network-based, and (3) DocNADEe (Gupta et al., 2019): DocNADE+Glove embeddings (Pennington et al., 2014). To quantify the quality of document representations, we employ doc2vec (Le and Mikolov, 2014) and EmbSum (to represent a document by summing the embedding vectors of its words using Glove). Using DocNADE, we first learn word embeddings and latent topics on each of the sources and then use them in knowledge transfer to $\mathcal{T}$. See the experimental setup and hyper-parameter configurations in *supplementary*.

Generalization via Perplexity (PPL): To evaluate the generative performance in TM, we estimate the log-probabilities for the test documents and compute the average held-out perplexity per



(a) **IR:** 20NSshort (b) **IR:** 20NSsmall (c) **IR:** TMNtitle

Figure 2: (a, b) Retrieval performance (precision) on 20NSshort and 20NSsmall datasets. (c) Precision at recall fraction 0.02, each for a fraction (20%, 40%, 60%, 80%, 100%) of the training set of TMNtitle. See *supplementary*.

word as, $PPL = \exp\big(-\frac{1}{N}\sum_{t=1}^{N}\frac{1}{|\mathbf{v}_t|}\log p(\mathbf{v}_t)\big)$, where N and $|\mathbf{v}_t|$ are the total number of documents and words in a document $\mathbf{v}_t$, respectively. Table 1 quantitatively shows PPL scores on the four target corpora, each with H=200 topics determined using the development set. Using TMN in LVT and MVT, we see improved (reduced) scores on `TMNtitle`: (655 vs 680) and (663 vs 680) respectively in comparison to DocNADEe. It suggests a positive knowledge transfer and domain overlap in `TMNtitle` and `TMN`. Also, MST+LVT boosts (663 vs 680) generalization in `TMNtitle`.

Interpretabilty via Topic Coherence (COH): To estimate meaningfulness of words in the topics captured, we follow Röder et al. (2015); Gupta et al. (2019) and compute COH with top 10 words in each topic. Higher scores imply more coherent topics. Table 1 (under COH column) demonstrates that our proposed knowledge transfer approaches show noticeable gains in COH, e.g., using `AGnews` as a source alone in GVT configuration for `20NSsmall` datatset, we observe COH of (.563 vs .455) compared to DocNADEe. On top, MST+MVT boosts COH for all the four targets compared to the baselines, suggesting the need for two complementary (word and topics) representations and knowledge transfers from several domains. Qualitatively, Table 2 illustrates example topics from target domains, where GVT using a corresponding source shows more coherent topics.

Applicability via Information Retrieval (IR): To evaluate document representations, we perform a document retrieval task on the target datasets and use their label information to compute precision. We follow the experimental setup similar to Lauly et al. (2017); Gupta et al. (2019), where all test documents are treated as queries to retrieve a fraction of the closest documents in the original training set using cosine similarity measure between their document vectors. To compute retrieval precision for each fraction (e.g., 0.001, 0.005, etc.), we average the number of retrieved training documents with the same label as the query.

Target	Source	Feature	Topic-words (top 5) on Target data
`20NSshort`	`20NS`	DocNADE	sale, price, monitor, site, setup
		GVT	shipping, sale, price, expensive, subscribe
`20NSshort`	`AGnews`	DocNADE	apple, modem, side, baud, perform
		GVT	microsoft, software, desktop, computer, apple
`TMNtitle`	`AGnews`	DocNADE	strike, jackson, kill, earthquake, injures
		GVT	earthquake, radiation, explosion, wildfire

Table 2: Topics on Target with/without transfers

Table 1 depicts precision scores at retrieval fraction 0.02, where the configuration MST+MVT outperforms DocNADEe in IR on *all* the four target datasets, e.g., (.314 vs .294) for `20NSshort`. We also see a large gain (.326 vs .274) due to MST+GVT for `20NSsmall`. Additionally, Figures 2a and 2b illustrate the precision on `20NSshort` (in MST+MVT) and `20NSsmall` (in MST+GVT), respectively, where they consistently outperform both DocNADEe and EmbSum at all fractions.

Moreover, we split the training data of TMNtitle into several sets: 20%, 40%, 60%, 80% of the training set and then retrain DocNADE, DocNADEe and DocNADE+MST+MVT. We demonstrate the impact of knowledge transfers via word and topic features in learning representations on the sparse target domain. Figure 2c plots precision at retrieval (recall) fraction 0.02 and demonstrates that the proposed modeling consistently reports a gain over DocNADE(e) at each of the splits.

Conclusion: Within neural topic modeling, we have demonstrated an approach to jointly transfer word embedding and latent topic features from many sources that better deals with data sparsity.

References

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. 2003. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Bin Cao, Sinno Jialin Pan, Yu Zhang, Dit-Yan Yeung, and Qiang Yang. 2010. Adaptive transfer learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*. AAAI Press.

Rajarshi Das, Manzil Zaheer, and Chris Dyer. 2015. Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 795–804. Association for Computational Linguistics.

Pankaj Gupta, Yatin Chaudhary, Florian Buettner, and Hinrich Schütze. 2019. Document informed neural autoregressive topic models with distributional prior. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*.

Hugo Larochelle and Stanislas Lauly. 2012. A neural autoregressive topic model. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems*, pages 2717–2725.

Stanislas Lauly, Yin Zheng, Alexandre Allauzen, and Hugo Larochelle. 2017. Document neural autoregressive distribution estimation. *Journal of Machine Learning Research*, 18:113:1–113:24.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1188–1196. JMLR.org.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems*, pages 3111–3119.

Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. 2015. Improving topic models with latent feature word representations. *TACL*, 3:299–313.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Michael Röder, Andreas Both, and Alexander Hinneburg. 2015. Exploring the space of topic coherence measures. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM 2015, Shanghai, China, February 2-6, 2015*, pages 399–408. ACM.

Ruslan Salakhutdinov and Geoffrey E. Hinton. 2009. Replicated softmax: an undirected topic model. In *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems*, pages 1607–1614. Curran Associates, Inc.

Chapter 8

textTOvec: Deep Contextualized Neural Autoregressive Topic Models of Language with Distributed Compositional Prior

textTOvec: DEEP CONTEXTUALIZED NEURAL AUTOREGRESSIVE TOPIC MODELS OF LANGUAGE WITH DISTRIBUTED COMPOSITIONAL PRIOR

Pankaj Gupta[1,2], Yatin Chaudhary[1], Florian Buettner[1], Hinrich Schütze[2]
[1]Corporate Technology, Machine-Intelligence (MIC-DE), Siemens AG Munich, Germany
[2]CIS, University of Munich (LMU) Munich, Germany
{pankaj.gupta, yatin.chaudhary, buettner.florian}@siemens.com

ABSTRACT

We address two challenges of probabilistic topic modelling in order to better estimate the probability of a word in a given context, i.e., $P(\text{word}|\text{context})$: (1) *No language structure in context*: Probabilistic topic models ignore word order by summarizing a given context as a "bag-of-word" and consequently the semantics of words in the context is lost. In this work, we incorporate language structure by combining a neural autoregressive topic model (TM) (e.g., DocNADE) with a LSTM based language model (LSTM-LM) in a single probabilistic framework. The LSTM-LM learns a vector-space representation of each word by accounting for word order in local collocation patterns, while the TM simultaneously learns a latent representation from the entire document. In addition, the LSTM-LM models complex characteristics of language (e.g., syntax and semantics), while the TM discovers the underlying thematic structure in a collection of documents. We unite two complementary paradigms of learning the meaning of word occurrences by combining a topic model and a language model in a unified probabilistic framework, named as ctx-DocNADE. (2) *Limited context and/or smaller training corpus of documents*: In settings with a small number of word occurrences (i.e., lack of context) in short text or data sparsity in a corpus of few documents, the application of TMs is challenging. We address this challenge by incorporating external knowledge into neural autoregressive topic models via a language modelling approach: we use word embeddings as input of a LSTM-LM with the aim to improve the word-topic mapping on a smaller and/or short-text corpus. The proposed DocNADE extension is named as ctx-DocNADEe.

We present novel neural autoregressive topic model variants coupled with neural language models and embeddings priors that consistently outperform state-of-the-art generative topic models in terms of generalization (perplexity), interpretability (topic coherence) and applicability (retrieval and classification) over 7 long-text and 8 short-text datasets from diverse domains.

1 INTRODUCTION

Probabilistic topic models, such as LDA (Blei et al., 2003), Replicated Softmax (RSM) (Salakhutdinov & Hinton, 2009) and Document Neural Autoregressive Distribution Estimator (DocNADE) (Larochelle & Lauly, 2012; Zheng et al., 2016; Lauly et al., 2017) are often used to extract topics from text collections, and predict the probabilities of each word in a given document belonging to each topic. Subsequently, they learn latent document representations that can be used to perform natural language processing (NLP) tasks such as information retrieval (IR), document classification or summarization. However, such probabilistic topic models ignore word order and represent a given context as a bag of its words, thereby disregarding semantic information.

To motivate our first task of extending probabilistic topic models to incorporate word order and language structure, assume that we conduct topic analysis on the following two sentences:

`Bear falls into market territory` and `Market falls into bear territory`

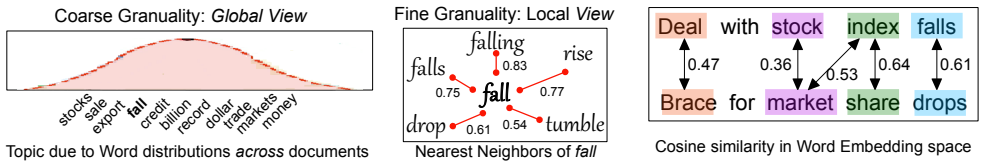


Figure 1: (left): A topic-word distribution due to global exposure, obtained from the matrix $\mathbf{W}$ as row-vector. (middle): Nearest neighbors in semantics space, represented by $\mathbf{W}$ in its column vectors. (right): BoW and cosine similarity illustration in distributed embedding space.

When estimating the probability of a word in a given context (here: $P(\text{“}bear\text{”}|\text{context})$), traditional topic models do not account for language structure since they ignore word order within the context and are based on “bag-of-words” (BoWs) only. In this particular setting, the two sentences have the same unigram statistics, but are about different topics. On deciding which topic generated the word “bear” in the second sentence, the preceding words “market falls” make it more likely that it was generated by a topic that assigns a high probability to words related to *stock market trading*, where “bear territory” is a colloquial expression in the domain. In addition, the language structure (e.g., syntax and semantics) is also ignored. For instance, the word “bear” in the first sentence is a proper noun and subject while it is an object in the second. In practice, topic models also ignore functional words such as “into”, which may not be appropriate in some scenarios.

Recently, Peters et al. (2018) have shown that a deep contextualized LSTM-based language model (LSTM-LM) is able to capture different language concepts in a layer-wise fashion, e.g., the lowest layer captures language syntax and topmost layer captures semantics. However, in LSTM-LMs the probability of a word is a function of its sentence only and word occurrences are modeled in a *fine granularity*. Consequently, LSTM-LMs do not capture semantics at a document level. To this end, recent studies such as TDLM (Lau et al., 2017), Topic-RNN (Dieng et al., 2016) and TCNLM (Wang et al., 2018) have integrated the merits of latent topic and neural language models (LMs); however, they have focused on improving LMs with global (semantics) dependencies using latent topics.

Similarly, while bi-gram LDA based topic models (Wallach, 2006; Wang et al., 2007) and n-gram based topic learning (Lauly et al., 2017) can capture word order in short contexts, they are unable to capture long term dependencies and language concepts. In contrast, DocNADE (Larochelle & Lauly, 2012) learns word occurrences across documents i.e., *coarse granularity* (in the sense that the topic assigned to a given word occurrence equally depends on all the other words appearing in the same document); however since it is based on the BoW assumption all language structure is ignored. In language modeling, Mikolov et al. (2010) have shown that recurrent neural networks result in a significant reduction of perplexity over standard n-gram models.

Contribution 1: We *introduce language structure* into neural autoregressive topic models via a LSTM-LM, thereby accounting for word ordering (or semantic regularities), language concepts and long-range dependencies. This allows for the accurate prediction of words, where the probability of each word is a function of global and local (semantics) contexts, modeled via DocNADE and LSTM-LM, respectively. The proposed neural topic model is named as *contextualized-Document Neural Autoregressive Distribution Estimator* (*ctx-DocNADE*) and offers learning complementary semantics by combining joint word and latent topic learning in a unified neural autoregressive framework. For instance, Figure 1 (left and middle) shows the complementary topic and word semantics, based on TM and LM representations of the term “fall”. Observe that the topic captures the usage of “fall” in the context of *stock market trading*, attributed to the global (semantic) view.

While this is a powerful approach for incorporating language structure and word order in particular for long texts and corpora with many documents, learning from contextual information remains challenging in settings with short texts and few documents, since (1) limited word co-occurrences or little context (2) significant word non-overlap in such short texts and (3) small training corpus of documents lead to little evidence for learning word co-occurrences. However, distributional word representations (i.e. word embeddings) (Pennington et al., 2014) have shown to capture both the semantic and syntactic relatedness in words and demonstrated impressive performance in NLP tasks.

For example, assume that we conduct topic analysis over the two short text fragments: `Deal with stock index falls` and `Brace for market share drops`. Traditional topic models

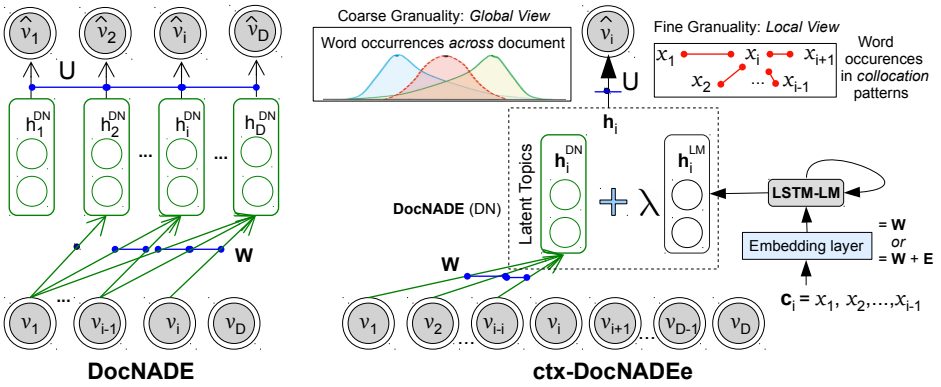


Figure 2: (left): DocNADE for the document $\mathbf{v}$. (right): ctx-DocNADEe for the observable corresponding to $v_i \in \mathbf{v}$. Blue colored lines signify the connections that share parameters. The observations (double circle) for each word v_i are multinomial, where v_i is the index in the vocabulary of the ith word of the document. $\mathbf{h}_i^{DN}$ and $\mathbf{h}_i^{LM}$ are hidden vectors from DocNADE and LSTM models, respectively for the target word v_i. Connections between each input v_i and hidden units $\mathbf{h}_i^{DN}$ are shared. The symbol $\hat{v}_i$ represents the autoregressive conditionals $p(v_i|\mathbf{v}_{<i})$, computed using $\mathbf{h}_i$ which is a weighted sum of $\mathbf{h}_i^{DN}$ and $\mathbf{h}_i^{LM}$ in ctx-DocNADEe.

with "BoW" assumption will not be able to infer relatedness between word pairs such as (*falls*, *drops*) due to the lack of word-overlap and small context in the two phrases. However, in the distributed embedding space, the word pairs are semantically related as shown in Figure 1 (left).

Related work such as Sahami & Heilman (2006) employed web search results to improve the information in short texts and Petterson et al. (2010) introduced word similarity via thesauri and dictionaries into LDA. Das et al. (2015) and Nguyen et al. (2015) integrated word embeddings into LDA and Dirichlet Multinomial Mixture (DMM) (Nigam et al., 2000) models. However, these works are based on LDA-based models without considering language structure, e.g. word order. In addition, DocNADE outperforms LDA and RSM topic models in terms of perplexity and IR.

Contribution 2: We incorporate *distributed compositional priors* in DocNADE: we use pre-trained word embeddings via LSTM-LM to supplement the multinomial topic model (i.e., DocNADE) in learning latent topic and textual representations on a smaller corpus and/or short texts. Knowing similarities in a distributed space and integrating this complementary information via a LSTM-LM, a topic representation is much more likely and coherent.

Taken together, we combine the advantages of complementary learning and external knowledge, and couple topic- and language models with pre-trained word embeddings to model short and long text documents in a unified neural autoregressive framework, named as *ctx-DocNADEe*. Our approach learns better textual representations, which we quantify via generalizability (e.g., perplexity), interpretability (e.g., topic extraction and coherence) and applicability (e.g., IR and classification).

To illustrate our two *contributions*, we apply our modeling approaches to 7 long-text and 8 short-text datasets from diverse domains and demonstrate that our approach consistently outperforms state-of-the-art generative topic models. Our learned representations, result in a gain of: (1) 4.6% (.790 vs .755) in topic coherence, (2) 6.5% (.615 vs .577) in precision at retrieval fraction 0.02, and (3) 4.4% (.662 vs .634) in $F1$ for text classification, averaged over 6 long-text and 8 short-text datasets.

When applied to short-text and long-text documents, our proposed modeling approaches generate *contextualized topic vectors*, which we name *textTOvec*. The *code* is available at `https://github.com/pgcool/textTOvec`.

2 Neural Autoregressive Topic Models

Generative models are based on estimating the probability distribution of multidimensional data, implicitly requiring modeling complex dependencies. Restricted Boltzmann Machine (RBM) (Hinton et al., 2006) and its variants (Larochelle & Bengio, 2008) are probabilistic undirected models of binary data. RSM (Salakhutdinov & Hinton, 2009) and its variants (Gupta et al., 2018) are gen-

eralization of the RBM, that are used to model word counts. However, estimating the complex probability distribution of the underlying high-dimensional observations is intractable. To address this challenge, NADE (Larochelle & Murray, 2011) decomposes the joint distribution of binary observations into autoregressive conditional distributions, each modeled using a feed-forward network. Unlike for RBM/RSM, this leads to tractable gradients of the data negative log-likelihood.

2.1 DOCUMENT NEURAL AUTOREGRESSIVE TOPIC MODEL (DOCNADE)

An extension of NADE and RSM, DocNADE (Larochelle & Lauly, 2012) models collections of documents as orderless bags of words (BoW approach), thereby disregarding any language structure. In other words, it is trained to learn word representations reflecting the underlying topics of the documents only, ignoring syntactical and semantic features as those encoded in word embeddings (Bengio et al., 2003; Mikolov et al., 2013; Pennington et al., 2014; Peters et al., 2018).

DocNADE (Lauly et al., 2017) represents a document by transforming its BoWs into a sequence $\mathbf{v} = [v_1, ..., v_D]$ of size D, where each element $v_i \in \{1, 2, ..., K\}$ corresponds to a multinomial observation (representing a word from a vocabulary of size K). Thus, v_i is the index in the vocabulary of the ith word of the document $\mathbf{v}$. DocNADE models the joint distribution $p(\mathbf{v})$ of all words v_i by decomposing it as $p(\mathbf{v}) = \prod_{i=1}^{D} p(v_i|\mathbf{v}_{<i})$, where each autoregressive conditional $p(v_i|\mathbf{v}_{<i})$ for the word observation v_i is computed using the preceding observations $\mathbf{v}_{<i} \in \{v_1, ..., v_{i-1}\}$ in a feed-forward neural network for $i \in \{1, ...D\}$,

$$\mathbf{h}_i^{DN}(\mathbf{v}_{<i}) = g(\mathbf{e} + \sum_{k<i} \mathbf{W}_{:,v_k}) \text{ and } p(v_i = w|\mathbf{v}_{<i}) = \frac{\exp(b_w + \mathbf{U}_{w,:}\mathbf{h}_i^{DN}(\mathbf{v}_{<i}))}{\sum_{w'} \exp(b_{w'} + \mathbf{U}_{w',:}\mathbf{h}_i^{DN}(\mathbf{v}_{<i}))} \tag{1}$$

where $g(\cdot)$ is an activation function, $\mathbf{U} \in \mathbb{R}^{K \times H}$ is a weight matrix connecting hidden to output, $\mathbf{e} \in \mathbb{R}^{H}$ and $\mathbf{b} \in \mathbb{R}^{K}$ are bias vectors, $\mathbf{W} \in \mathbb{R}^{H \times K}$ is a word representation matrix in which a column $\mathbf{W}_{:,v_i}$ is a vector representation of the word v_i in the vocabulary, and H is the number of hidden units (topics). The log-likelihood of any document $\mathbf{v}$ of any arbitrary length is given by: $\mathcal{L}^{DN}(\mathbf{v}) = \sum_{i=1}^{D} \log p(v_i|\mathbf{v}_{<i})$. Note that the past word observations $\mathbf{v}_{<i}$ are orderless due to BoWs, and may not correspond to the words preceding the ith word in the document itself.

Algorithm 1 Computation of $\log p(\mathbf{v})$

Input: A training document $\mathbf{v}$
Input: Word embedding matrix $\mathbf{E}$
Output: $\log p(\mathbf{v})$

$\mathbf{a} \leftarrow \mathbf{e}$
$q(\mathbf{v}) = 1$
for i from 1 to D **do**
  compute $\mathbf{h}_i$ and $p(v_i|\mathbf{v}_{<i})$
  $q(\mathbf{v}) \leftarrow q(\mathbf{v})p(v_i|\mathbf{v}_{<i})$
  $\mathbf{a} \leftarrow \mathbf{a} + \mathbf{W}_{:,v_i}$
$\log p(\mathbf{v}) \leftarrow \log q(\mathbf{v})$

model	$\mathbf{h}_i$	$p(v_i\|\mathbf{v}_{<i})$
DocNADE	$\mathbf{h}_i^{DN} \leftarrow g(\mathbf{a})$ $\mathbf{h}_i \leftarrow \mathbf{h}_i^{DN}$	equation 1
ctx-DocNADE	$\mathbf{h}_i^{LM} \leftarrow \text{LSTM}(\mathbf{c}_i, \text{embedding} = \mathbf{W})$ $\mathbf{h}_i \leftarrow \mathbf{h}_i^{DN} + \lambda\, \mathbf{h}_i^{LM}$	equation 2
ctx-DocNADEe	$\mathbf{h}_i^{LM} \leftarrow \text{LSTM}(\mathbf{c}_i, \text{embedding} = \mathbf{W} + \mathbf{E})$ $\mathbf{h}_i \leftarrow \mathbf{h}_i^{DN} + \lambda\, \mathbf{h}_i^{LM}$	equation 2

Table 1: Computation of $\mathbf{h}_i$ and $p(v_i|\mathbf{v}_{<i})$ in DocNADE, ctx-DocNADE and ctx-DocNADEe models, correspondingly used in estimating $\log p(\mathbf{v})$ (Algorithm 1).

2.2 DEEP CONTEXTUALIZED DOCNADE WITH DISTRIBUTIONAL SEMANTICS

We propose two extensions of the DocNADE model: (1) *ctx-DocNADE*: introducing language structure via LSTM-LM and (2) *ctx-DocNADEe*: incorporating external knowledge via pre-trained word embeddings $\mathbf{E}$, to model short and long texts. The unified network(s) account for the ordering of words, syntactical and semantic structures in a language, long and short term dependencies, as well as external knowledge, thereby circumventing the major drawbacks of BoW-based representations.

Similar to DocNADE, ctx-DocNADE models each document $\mathbf{v}$ as a sequence of multinomial observations. Let $[x_1, x_2, ..., x_N]$ be a sequence of N words in a given document, where x_i is represented by an embedding vector of dimension, dim. Further, for each element $v_i \in \mathbf{v}$, let $\mathbf{c}_i = [x_1, x_2, ..., x_{i-1}]$ be the context (preceding words) of ith word in the document. Unlike in DocNADE, the conditional probability of the word v_i in ctx-DocNADE (or ctx-DocNADEe) is a function of two hidden vectors: $\mathbf{h}_i^{DN}(\mathbf{v}_{<i})$ and $\mathbf{h}_i^{LM}(\mathbf{c}_i)$, stemming from the DocNADE-based and

short-text									long-text							
Data	**Train**	**Val**	**Test**	**|RV|**	**|FV|**	**L**	**C**	**Domain**	**Data**	**Train**	**Val**	**Test**	**|RV|**	**|FV|**	**L**	**C**	**Domain**
20NSshort	1.3k	0.1k	0.5k	1.4k	1.4k	13.5	20	News	20NSsmall	0.4k	0.2k	0.2k	2k	4555	187.5	20	News
TREC6	5.5k	0.5k	0.5k	2k	2295	9.8	6	Q&A	Reuters8	5.0k	0.5k	2.2k	2k	7654	102	8	News
R21578title†	7.3k	0.5k	3.0k	2k	2721	7.3	90	News	20NS	7.9k	1.6k	5.2k	2k	33770	107.5	20	News
Subjectivity	8.0k	.05k	2.0k	2k	7965	23.1	2	Senti	R21578†	7.3k	0.5k	3.0k	2k	11396	128	90	News
Polarity	8.5k	.05k	2.1k	2k	7157	21.0	2	Senti	BNC	15.0k	1.0k	1.0k	9.7k	41370	1189	-	News
TMNtitle	22.8k	2.0k	7.8k	2k	6240	4.9	7	News	SiROBs†	27.0k	1.0k	10.5k	3k	9113	39	22	Indus
TMN	22.8k	2.0k	7.8k	2k	12867	19	7	News	AGNews	118k	2.0k	7.6k	5k	34071	38	4	News
AGnewstitle	118k	2.0k	7.6k	5k	17125	6.8	4	News									

Table 2: Data statistics: Short/long texts and/or small/large corpora from diverse domains. Symbols- Avg: average, L: avg text length (#words), $|RV|$ and $|FV|$: size of reduced (RV) and full vocabulary (FV), C: number of classes, Senti: Sentiment, Indus: Industrial, 'k':thousand and †: multi-label. For short-text, L<25.

LSTM-based components of ctx-DocNADE, respectively:

$$\mathbf{h}_i(\mathbf{v}_{<i}) = \mathbf{h}_i^{DN}(\mathbf{v}_{<i}) + \lambda\ \mathbf{h}_i^{LM}(\mathbf{c}_i) \text{ and } p(v_i = w|\mathbf{v}_{<i}) = \frac{\exp(b_w + \mathbf{U}_{w,:}\mathbf{h}_i(\mathbf{v}_{<i}))}{\sum_{w'}\exp(b_{w'} + \mathbf{U}_{w',:}\mathbf{h}_i(\mathbf{v}_{<i}))} \tag{2}$$

where $\mathbf{h}_i^{DN}(\mathbf{v}_{<i})$ is computed as in DocNADE (equation 1) and λ is the mixture weight of the LM component, which can be optimized during training (e.g., based on the validation set). The second term $\mathbf{h}_i^{LM}$ is a context-dependent representation and output of an LSTM layer at position $i-1$ over input sequence $\mathbf{c}_i$, trained to predict the next word v_i. The LSTM offers history for the ith word via modeling temporal dependencies in the input sequence, $\mathbf{c}_i$. The conditional distribution for each word v_i is estimated by equation 2, where the unified network of DocNADE and LM combines global and context-dependent representations. Our model is jointly optimized to maximize the pseudo log likelihood, $\log p(\mathbf{v}) \approx \sum_{i=1}^{D}\log p(v_i|\mathbf{v}_{<i})$ with stochastic gradient descent. See Larochelle & Lauly (2012) for more details on training from bag of word counts.

In the weight matrix $\mathbf{W}$ of DocNADE (Larochelle & Lauly, 2012), each row vector $\mathbf{W}_{j,:}$ encodes topic information for the jth hidden topic feature and each column vector $\mathbf{W}_{:,v_i}$ is a vector for the word v_i. To obtain complementary semantics, we exploit this property and expose $\mathbf{W}$ to both global and local influences by sharing $\mathbf{W}$ in the DocNADE and LSTM-LM componenents. Thus, the embedding layer of LSTM-LM component represents the column vectors.

ctx-DocNADE, in this realization of the unified network the embedding layer in the LSTM component is randomly initialized. This extends DocNADE by accounting for the ordering of words and language concepts via context-dependent representations for each word in the document.

ctx-DocNADEe, the second version extends ctx-DocNADE with distributional priors, where the embedding layer in the LSTM component is initialized by the sum of a pre-trained embedding matrix $\mathbf{E}$ and the weight matrix $\mathbf{W}$. Note that $\mathbf{W}$ is a model parameter; however $\mathbf{E}$ is a static prior.

Algorithm 1 and Table 1 show the $\log p(\mathbf{v})$ for a document $\mathbf{v}$ in three different settings: *DocNADE*, *ctx-DocNADE* and *ctx-DocNADEe*. In the DocNADE component, since the weights in the matrix $\mathbf{W}$ are tied, the linear activation $\mathbf{a}$ can be re-used in every hidden layer and computational complexity reduces to $O(HD)$, where H is the size of each hidden layer. In every epoch, we run an LSTM over the sequence of words in the document and extract hidden vectors h_i^{LM}, corresponding to $\mathbf{c}_i$ for every target word v_i. Therefore, the computational complexity in ctx-DocNADE or ctx-DocNADEe is $O(HD + \mathfrak{N})$, where $\mathfrak{N}$ is the total number of edges in the LSTM network (Hochreiter & Schmidhuber, 1997; Sak et al., 2014). The trained models can be used to extract a *textTOvec* representation, i.e., $\mathbf{h}(\mathbf{v}^*) = \mathbf{h}^{DN}(\mathbf{v}^*) + \lambda\,\mathbf{h}^{LM}(\mathbf{c}^*_{N+1})$ for the text $\mathbf{v}^*$ of length $\mathbf{D}^*$, where $\mathbf{h}^{DN}(\mathbf{v}^*) = g(\mathbf{e} + \sum_{k\leq\mathbf{D}^*}\mathbf{W}_{:,v_k})$ and $\mathbf{h}^{LM}(\mathbf{c}^*_{N+1}) = \text{LSTM}(\mathbf{c}^*_{N+1}, \text{embedding} = \mathbf{W} \text{ or } (\mathbf{W}+\mathbf{E}))$.

ctx-DeepDNEe: DocNADE and LSTM can be extended to a deep, multiple hidden layer architecture by adding new hidden layers as in a regular deep feed-forward neural network, allowing for improved performance. In the deep version, the first hidden layer is computed in an analogous fashion to DocNADE variants (equation 1 or 2). Subsequent hidden layers are computed as:

$$\mathbf{h}_{i,d}^{DN}(\mathbf{v}_{<i}) = g(\mathbf{e}_d + \mathbf{W}_{i,d}\cdot\mathbf{h}_{i,d-1}(\mathbf{v}_{<i}))\ \text{ or }\ \mathbf{h}_{i,d}^{LM}(\mathbf{c}_i) = deepLSTM(\mathbf{c}_i, \text{depth} = d)$$

for $d = 2, ...n$, where n is the total number of hidden layers (i.e., depth) in the deep feed-forward and LSTM networks. For d=1, the hidden vectors $\mathbf{h}_{i,1}^{DN}$ and $\mathbf{h}_{i,1}^{LM}$ correspond to equations 1 and 2. The conditional $p(v_i = w|\mathbf{v}_{<i})$ is computed using the last layer n, i.e., $\mathbf{h}_{i,n} = \mathbf{h}_{i,n}^{DN} + \lambda\,\mathbf{h}_{i,n}^{LM}$.

Model	20NSshort		TREC6		R21578title		Subjectivity		Polarity		TMNtitle		TMN		AGnewstitle		Avg	
	IR	$F1$	IR	$F1$	IR	$F1$	IR	$F1$	IR	$F1$	IR	$F1$	IR	$F1$	IR	$F1$	IR	$F1$
glove(RV)	.236	.493	.480	.798	.587	.356	.754	.882	.543	.715	.513	.693	.638	.736	.588	814	.542	.685
glove(FV)	.236	.488	.480	.785	.595	.356	.775	.901	.553	.728	.545	.736	.643	.813	.612	.830	.554	.704
doc2vec	.090	.413	.260	.400	.518	.176	.571	.763	.510	.624	.190	.582	.220	.720	.265	.600	.328	.534
Gauss-LDA	.080	.118	.325	.202	.367	.012	.558	.676	.505	.511	.408	.472	.713	.692	.516	.752	.434	.429
glove-DMM	.183	.213	.370	.454	.273	.011	.738	.834	.515	.585	.445	.590	.551	.666	.540	.652	.451	.500
glove-LDA	.160	.320	.300	.600	.387	.052	.610	.805	.517	.607	.260	.412	.428	.627	.547	.687	.401	.513
TDLM	.219	.308	.521	.671	.563	.174	.839	.885	.520	.599	.535	.657	.672	.767	.534	.722	.550	.586
DocNADE(RV)	.290	.440	.550	.804	.657	.313	.820	.889	.560	.699	.524	.664	.652	.759	.656	.819	.588	.673
DocNADE(FV)	.290	.440	.546	.791	.654	.302	.848	.907	.576	.724	.525	.688	.687	.796	.678	.821	.600	.683
DeepDNE	.100	.080	.479	.629	.630	.221	.865	.909	.503	.531	.536	.661	.671	.783	.682	.825	.558	.560
ctx-DocNADE	.296	.440	.595	.817	.641	.300	.874	.910	.591	.725	.560	.687	.692	.793	.691	.826	.617	.688
ctx-DocNADEe	**.306**	.490	.599	.824	**.656**	.308	.874	.917	**.605**	.740	**.595**	.726	**.698**	.806	**.703**	.828	**.630**	.705
ctx-DeepDNEe	.278	.416	**.606**	.804	.647	.244	**.878**	.920	.591	.723	.576	.694	.687	.796	.689	.826	.620	.688

Table 3: State-of-the-art comparison: IR (i.e, IR-precision at 0.02 fraction) and classification $F1$ for *short* texts, where Avg: average over the row values, the **bold** and underline: the maximum for IR and F1, respectively.

3 EVALUATION

We apply our modeling approaches (in improving topic models, i.e, DocNADE using language concepts from LSTM-LM) to 8 short-text and 7 long-text datasets of varying size with single/multi-class labeled documents from public as well as industrial corpora. We present four quantitative measures in evaluating topic models: generalization (perplexity), topic coherence, text retrieval and categorization. See the *appendices* for the data description and example texts. Table 2 shows the data statistics, where 20NS and R21578 signify 20NewsGroups and Reuters21578, respectively.

Baselines: While, we evaluate our multi-fold contributions on four tasks: generalization (perplexity), topic coherence, text retrieval and categorization, we compare performance of our proposed models `ctx-DocNADE` and `ctx-DocNADEe` with related baselines based on: **(1)** word representation: `glove` (Pennington et al., 2014), where a document is represented by summing the embedding vectors of it's words, **(2)** document representation: `doc2vec` (Le & Mikolov, 2014), **(3)** LDA based BoW TMs: `ProdLDA` (Srivastava & Sutton, 2017) and `SCHOLAR`[1] (Card et al., 2017) **(4)** neural BoW TMs: `DocNADE` and `NTM` (Cao et al., 2015) and , **(5)** TMs, including pre-trained word embeddings: `Gauss-LDA` (`GaussianLDA`) (Das et al., 2015), and `glove-DMM`, `glove-LDA` (Nguyen et al., 2015). **(6)** jointly[2] trained topic and language models: `TDLM` (Lau et al., 2017), `Topic-RNN` (Dieng et al., 2016) and `TCNLM` (Wang et al., 2018).

Model	20NSsmall		Reuters8		20NS		R21578		SiROBs		AGnews		Avg	
	IR	$F1$	IR	$F1$	IR	$F1$	IR	$F1$	IR	$F1$	IR	$F1$	IR	$F1$
glove(RV)	.214	.442	.845	.830	.200	.608	.644	.316	.273	.202	.725	.870	.483	.544
glove(FV)	.238	.494	.837	.880	.253	.632	.659	.340	.285	.217	.737	.890	.501	.575
doc2vec	.200	.450	.586	.852	.216	.691	.524	.215	.282	.226	.387	.713	.365	.524
Gauss-LDA	.090	.080	.712	.557	.142	.340	.539	.114	.232	.070	.456	.818	.361	.329
glove-DMM	.060	.134	.623	.453	.092	.187	.501	.023	.226	.050	-	-	-	-
DocNADE(RV)	.270	.530	**.884**	.890	.366	.644	**.723**	.336	.374	.298	.787	.882	.567	.596
DocNADE(FV)	.299	.509	.879	.907	.427	.727	.715	.340	.382	.308	.794	.888	.582	.613
ctx-DocNADE	.313	.526	.880	.898	.472	.732	.714	.315	.386	.309	.791	.890	.592	.611
ctx-DocNADEe	**.327**	.524	.883	.900	**.486**	.745	.721	.332	**.390**	.311	**.796**	.894	**.601**	.618

Table 4: IR-precision at fraction 0.02 and classification $F1$ for *long* texts

	Model	PPL		Model	PPL
Subjectivity	`DocNADE`	980	AGnewstitle	`DocNADE`	846
	`ctx-DocNADE`	968		`ctx-DocNADE`	822
	`ctx-DocNADEe`	966		`ctx-DocNADEe`	820
Reuters8	`DocNADE`	283	20NS	`DocNADE`	1375
	`ctx-DocNADE`	276		`ctx-DocNADE`	1358
	`ctx-DocNADEe`	272		`ctx-DocNADEe`	1361
TMNtitle	`DocNADE`	1437	20NSshort	`DocNADE`	646
	`ctx-DocNADE`	1430		`ctx-DocNADE`	656
	`ctx-DocNADEe`	1427		`ctx-DocNADEe`	648

Table 5: Generalization: PPL

Experimental Setup: DocNADE is often trained on a reduced vocabulary (RV) after pre-processing (e.g., ignoring functional words, etc.); however, we also investigate training it on full text/vocabulary (FV) (Table 2) and compute document representations to perform different evaluation tasks. The FV setting preserves the language structure, required by LSTM-LM, and allows a fair comparison of DocNADE+FV and ctx-DocNADE variants. We use the glove embedding of 200 dimensions. All the baselines and proposed models (*ctx-DocNADE*, *ctx-DocNADEe* and *ctx-DeepDNEe*) were run in the FV setting over 200 topics to quantify the quality of the learned representations. To better initialize the complementary learning in ctx-DocNADEs, we perform a pre-training for 10 epochs with λ set to 0. See the *appendices* for the experimental setup and hyperparameters for the following tasks, including the ablation over λ on validation set.

[1] focuses on incorporating meta-data (author, date, etc.) into TMs; SCHOLAR w/o meta-data $\equiv$ ProdLDA

[2] though focused on improving language models using topic models, different to our motivation

Data	glove-DMM		glove-LDA		DocNADE		ctx-DNE		ctx-DNEe		Data	glove-DMM		DocNADE		ctx-DNE		ctx-DNEe	
	W10	W20	W10	W20	W10	W20	W10	W20	W10	W20		W10	W20	W10	W20	W10	W20	W10	W20
20NSshort	.512	.575	.616	.767	.669	.779	.682	.794	**.696**	**.801**	*Subjectivity*	.538	.433	.613	.749	.629	.767	**.634**	**.771**
TREC6	.410	.475	.551	.736	.699	.818	**.714**	**.810**	.713	.809	*AGnewstitle*	.584	.678	.731	.757	.739	.858	**.746**	**.865**
R21578title	.364	.458	.478	.677	.701	.812	.713	.802	**.723**	**.834**	*20NSsmall*	**.578**	.548	.508	.628	.546	.667	.565	**.692**
Polarity	.637	.363	.375	.468	.610	.742	.611	.756	**.650**	**.779**	*Reuters8*	.372	.302	.583	.710	.584	.710	**.592**	**.714**
TMNtitle	.633	.778	.651	.798	.712	.822	.716	.831	**.735**	**.845**	*20NS*	.458	.374	.606	.729	.615	.746	**.631**	**.759**
TMN	.705	.444	.550	.683	.642	.762	.639	.759	**.709**	**.825**	Avg (all)	.527	..452	.643	.755	.654	.772	**.672**	**.790**

Table 6: Average coherence for *short* and *long* texts over 200 topics in FV setting, where *DocNADE* $\leftrightarrow$ *DNE*

We run TDLM[3] (Lau et al., 2017) for all the short-text datasets to evaluate the quality of representations learned in the spare data setting. For a fair comparison, we set 200 topics and hidden size, and initialize with the same pre-trained word embeddings (i.e., glove) as used in the ctx-DocNADEe.

3.1 GENERALIZATION: PERPLEXITY (PPL)

To evaluate the generative performance of the topic models, we estimate the log-probabilities for the test documents and compute the average held-out perplexity (PPL) per word as, $PPL = \exp\big(-\frac{1}{z}\sum_{t=1}^{z}\frac{1}{|\mathbf{v}^t|}\log p(\mathbf{v}^t)\big)$, where z and $|\mathbf{v}^t|$ are the total number of documents and words in a document $\mathbf{v}^t$. For DocNADE, the log-probability $\log p(\mathbf{v}^t)\big)$ is computed using $\mathcal{L}^{DN}(\mathbf{v})$; however, we ignore the mixture coefficient, i.e., λ=0 (equation 2) to compute the exact log-likelihood in ctx-DocNADE versions. The optimal λ is determined based on the validation set. Table 5 quantitatively shows the PPL scores, where the complementary learning with $\lambda = 0.01$ (optimal) in ctx-DocNADE achieves lower perplexity than the baseline DocNADE for both short and long texts, e.g., (822 vs 846) and (1358 vs 1375) on *AGnewstitle* and *20NS* [4] datasets, respectively in the FV setting.

3.2 INTERPRETABILITY: TOPIC COHERENCE

We compute topic coherence (Chang et al., 2009; Newman et al., 2009; Gupta et al., 2018) to assess the meaningfulness of the underlying topics captured. We choose the coherence measure proposed by Röder et al. (2015) , which identifies context features for each topic word using a sliding window over the reference corpus. Higher scores imply more coherent topics.

We use the gensim module (*radimrehurek.com/gensim/models/coherencemodel.html*, *coherence type* = c_v) to estimate coherence for each of the 200 topics (top 10 and 20 words). Table 6 shows average coherence over 200 topics, where the higher scores in ctx-DocNADE compared to DocNADE (.772 vs .755) suggest that the contextual information and language structure help in generating more coherent topics. The introduction of embeddings in ctx-DocNADEe boosts the topic coherence, leading to a gain of 4.6% (.790 vs .755) on average over 11 datasets. Note that the proposed models also outperform the baselines methods glove-DMM and glove-LDA. Qualitatively, Table 8 illustrates an example topic from the 20NSshort text dataset for DocNADE, ctx-DocNADE and ctx-DocNADEe, where the inclusion of embeddings results in a more coherent topic.

Additional Baslines: We further compare our proposed models to other approaches that combining topic and language models, such as TDLM (Lau et al., 2017), Topic-RNN (Dieng et al., 2016) and TCNLM (Wang et al., 2018). However, the related studies focus on improving language models using topic models: in contrast, the focus of our work is on improving topic models for textual representations (short-text or long-text documents) by incorporating language concepts (e.g., word ordering, syntax, semantics, etc.) and external knowledge (e.g., word embeddings) via neural language models, as discussed in section 1.

To this end, we follow the experimental setup of the most recent work, TCNML and *quantitatively* compare the performance of our models (i.e., ctx-DocNADE and ctx-DocNADEe) in terms of topic coherence (NPMI) on BNC dataset. Table 7 (left) shows NPMI scores of different models, where the results suggest that our contribution (i.e., ctx-DocNADE) of introducing language concepts into BoW topic model (i.e., DocNADE) improves topic coherence[5]. The better performance for high val-

[3]https://github.com/jhlau/topically-driven-language-model

[4]PPL scores in (RV/FV) settings: DocNADE (665/1375) outperforms ProdLDA (1168/2097) on 200 topics

[5]NPMI over (50/200) topics learned on 20NS by: ProdLDA (.24/.19) and DocNADE (.15/.12) in RV setting

Model	Coherence (NPMI)		
	50	100	150
(*sliding window*=20)			
LDA#	.106	.119	.119
NTM#	.081	.070	.072
TDLM(s)#	.102	.106	.100
TDLM(l)#	.095	.101	.104
Topic-RNN(s)#	.102	.108	.102
Topic-RNN(l)#	.100	.105	.097
TCNLM(s)#	.114	.111	.107
TCNLM(l)#	.101	.104	.102
DocNADE	.097	.095	.097
ctx-DocNADE*(λ=0.2)	.102	.103	.102
ctx-DocNADE*(λ=0.8)	.106	.105	.104
ctx-DocNADEe*(λ=0.2)	.098	.101	-
ctx-DocNADEe*(λ=0.8)	.105	.104	-
(*sliding window*=110)			
DocNADE	.133	.131	.132
ctx-DocNADE*(λ=0.2)	.134	.141	.138
ctx-DocNADE*(λ=0.8)	.139	.142	.140
ctx-DocNADEe*(λ=0.2)	.133	.139	-
ctx-DocNADEe*(λ=0.8)	.135	.141	-

Topic	Model	Topic-words (ranked by their probabilities in topic)
environment	TCNLM#	pollution, emissions, nuclear, waste, environmental
	ctx-DocNADE*	ozone, pollution, emissions, warming, waste
	ctx-DocNADEe*	pollution, emissions, dioxide, warming, environmental
politics	TCNLM#	elections, economic, minister, political, democratic
	ctx-DocNADE*	elections, democracy, votes, democratic, communist
	ctx-DocNADEe*	democrat, candidate, voters, democrats, poll
art	TCNLM#	album, band, guitar, music, film
	ctx-DocNADE*	guitar, album, band, bass, tone
	ctx-DocNADEe*	guitar, album, pop, guitars, song
facilities	TCNLM#	bedrooms, hotel, garden, situated, rooms
	ctx-DocNADE*	bedrooms, queen, hotel, situated, furnished
	ctx-DocNADEe*	hotel, bedrooms, golf, resorts, relax
business	TCNLM#	corp, turnover, unix, net, profits
	ctx-DocNADE*	shares, dividend, shareholders, stock, profits
	ctx-DocNADEe*	profits, growing, net, earnings, turnover
expression	TCNLM#	eye, looked, hair, lips, stared
	ctx-DocNADE*	nodded, shook, looked, smiled, stared
	ctx-DocNADEe*	charming, smiled, nodded, dressed, eyes
education	TCNLM#	courses, training, students, medau, education
	ctx-DocNADE*	teachers, curriculum, workshops, learning, medau
	ctx-DocNADEe*	medau, pupils, teachers, schools, curriculum

Table 7: (Left): Topic coherence (NMPI) scores of different models for 50, 100 and 150 topics on BNC dataset. The *sliding window* is one of the hyper-parameters for computing topic coherence (Röder et al., 2015; Wang et al., 2018). A *sliding window* of 20 is used in TCNLM; in addition we also present results for a window of size 110. λ is the mixture weight of the LM component in the topic modeling process, and (s) and (l) indicate small and large model, respectively. The symbol '-' indicates no result, since word embeddings of 150 dimensions are not available from glove vectors. (Right): The top 5 words of seven learnt topics from our models and TCNLM. The asterisk (*) indicates our proposed models and (#) taken from TCNLM (Wang et al., 2018).

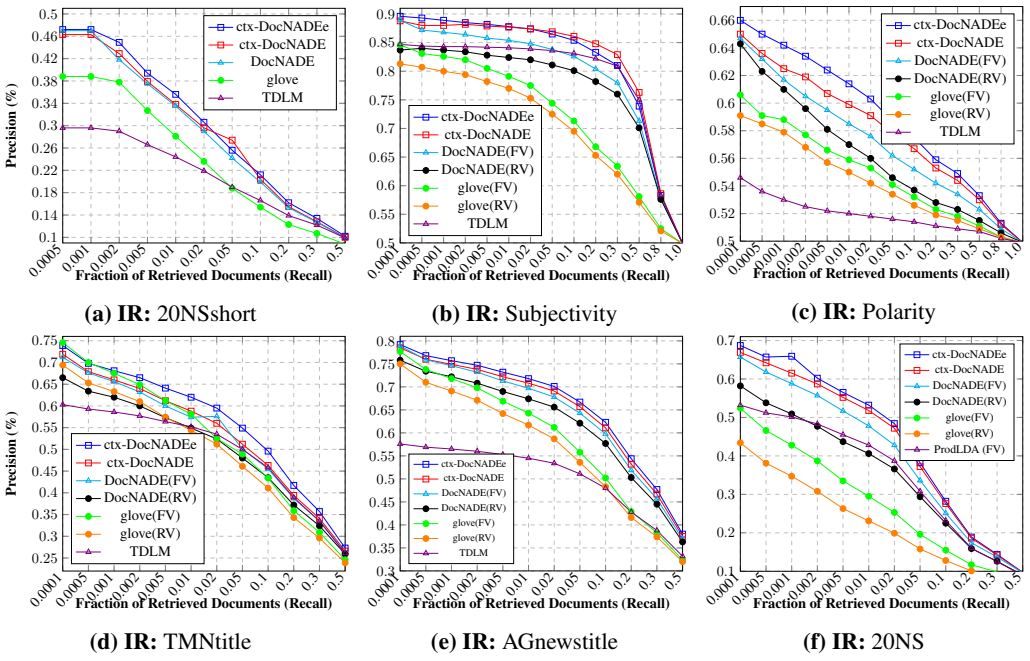
ues of λ illustrates the relevance of the LM component for topic coherence (DocNADE corresponds to λ=0). Similarly, the inclusion of word embeddings (i.e., ctx-DocNADEe) results in more coherent topics than the baseline DocNADE. Importantly, while ctx-DocNADEe is motivated by sparse data settings, the BNC dataset is neither a collection of short-text nor a corpus of few documents. Consequently, ctx-DocNADEe does not show improvements in topic coherence over ctx-DocNADE.

In Table 7 (right), we further *qualitatively* show the top 5 words of seven learnt topics (topic name summarized by Wang et al. (2018)) from our models (i.e., ctx-DocNADE and ctx-DocNADEe) and TCNLM. Since the BNC dataset is unlabeled, we are here restricted to comparing model performance in terms of topic coherence only.

3.3 Applicability: Text Retrieval and Categorization

Text Retrieval: We perform a document retrieval task using the short-text and long-text documents with label information. We follow the experimental setup similar to Lauly et al. (2017), where all test documents are treated as queries to retrieve a fraction of the closest documents in the original training set using cosine similarity measure between their `textTOvec` representations (section 2.2). To compute retrieval precision for each fraction (e.g., 0.0001, 0.005, 0.01, 0.02, 0.05, etc.), we average the number of retrieved training documents with the same label as the query. For multi-label datasets, we average the precision scores over multiple labels for each query. Since, Salakhutdinov & Hinton (2009) and Lauly et al. (2017) have shown that RSM and DocNADE strictly outperform LDA on this task, we solely compare DocNADE with our proposed extensions.

Table 3 and 4 show the retrieval precision scores for the short-text and long-text datasets, respectively at retrieval fraction 0.02. Observe that the introduction of both pre-trained embeddings and language/contextual information leads to improved performance on the IR task noticeably for short texts. We also investigate topic modeling without pre-processing and filtering certain words, i.e. the FV setting and find that the DocNADE(FV) or glove(FV) improves IR precision over the baseline RV setting. Therefore, we opt for the FV in the proposed extensions. On an average over the 8 short-text and 6 long-text datasets, *ctx-DocNADEe* reports a gain of 7.1% (.630 vs .588) (Table 3) 6.0% (.601 vs .567) (Table 4), respectively in precision compared to DocNADE(RV). To further compare with TDLM, our proposed models (ctx-DocNADE and ctx-DocNADEe) outperform it by a notable margin for all the short-text datasets, i.e., a gain of 14.5% (.630 vs .550: ctx-DocNADEe vs TDLM)



(a) IR: 20NSshort **(b) IR:** Subjectivity **(c) IR:** Polarity

(d) IR: TMNtitle **(e) IR:** AGnewstitle **(f) IR:** 20NS

Figure 3: Retrieval performance (IR-precision) on 6 datasets at different fractions

in IR-precision. In addition, the deep variant (d=3) with embeddings, i.e., ctx-DeepDNEe shows competitive performance on TREC6 and Subjectivity datasets.

Figures (3a, 3b, 3c, 3d, 3e and 3f) illustrate the average precision for the retrieval task on 6 datasets. Observe that the ctx-DocNADEe outperforms DocNADE(RV) at all the fractions and demonstrates a gain of 6.5% (.615 vs .577) in precision at fraction 0.02, averaged over 14 datasets. Additionally, our proposed models outperform TDLM and ProdLDA[6] (for 20NS) by noticeable margins.

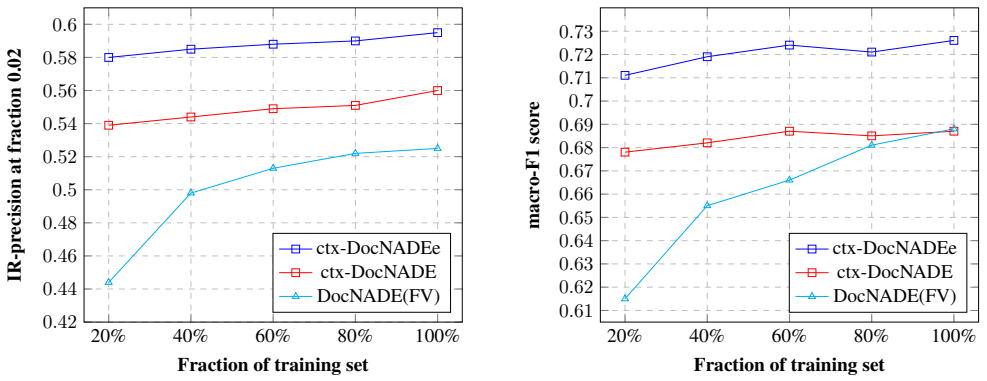
Text Categorization: We perform text categorization to measure the quality of our `textTovec` representations. We consider the same experimental setup as in the retrieval task and extract `textTOvec` of 200 dimension for each document, learned during the training of ctx-DocNADE variants. To perform text categorization, we employ a logistic regression classifier with $L2$ regularization. While, *ctx-DocNADEe* and *ctx-DeepDNEe* make use of glove embeddings, they are evaluated against the topic model baselines with embeddings. For the short texts (Table 3), the *glove* leads DocNADE in classification performance, suggesting a need for distributional priors in the topic model. Therefore, the ctx-DocNADEe reports a gain of 4.8% (.705 vs .673) and 3.6%(.618 vs .596) in $F1$, compared to DocNADE(RV) on an average over the short (Table 3) and long (Table 4) texts, respectively. In result, a gain of 4.4% (.662 vs .634) overall.

In terms of classification accuracy on 20NS dataset, the scores are: DocNADE (0.734), ctx-DocNADE (0.744), ctx-DocNADEe (0.751), NTM (0.72) and SCHOLAR (0.71). While, our proposed models, i.e., ctx-DocNADE and ctx-DocNADEe outperform both NTM (results taken from Cao et al. (2015), Figure 2) and SCHOLAR (results taken from Card et al. (2017), Table 2), the DocNADE establishes itself as a strong neural topic model baseline.

3.4 Inspection of Learned Representations

To further interpret the topic models, we analyze the meaningful semantics captured via topic extraction. Table 8 shows a topic extracted using 20NS dataset that could be interpreted as *computers*, which are (sub)categories in the data, confirming that meaningful topics are captured. Observe that

[6]IR-precision scores at 0.02 retrieval fraction on the short-text datasets by ProdLDA: *20NSshort* (.08), *TREC6* (.24), *R21578title* (.31), *Subjectivity* (.63) and *Polarity* (.51). Therefore, the *DocNADE, ctx-DocNADE and ctx-DocNADEe outperform ProdLDA in both the settings*: data sparsity and sufficient co-occurrences.



(a) Text Retrieval **(b)** Text classification

Figure 4: Evaluations at different fractions (20%, 40%, 60%, 80%, 100%) of the training set of TMNtitle

the ctx-DocNADEe extracts a more coherent topic due to embedding priors. To *qualitatively* inspect the contribution of word embeddings and `textTOvec` representations in topic models, we analyse the text retrieved for each query using the representations learned from DocNADE and ctx-DoocNADEe models. Table 9 illustrates the retrieval of the top 3 texts for an input query, selected from *TMNtitle* dataset, where #match is YES if the query and retrievals have the same class label. Observe that ctx-DocNADEe retrieves the top 3 texts, each with no unigram overlap with the query.

DocNADE	*ctx-DocNADE*	*ctx-DocNADEe*
vga, screen,	computer, color,	svga, graphics
computer, sell,	screen, offer,	bar, macintosh,
color, powerbook,	vga, card,	san, windows,
sold, cars,	terminal, forsale,	utility, monitor,
svga, offer	gov, vesa	computer, processor
.554	.624	.667

Table 8: A topic of 20NS dataset with coherence

	Query :: "emerging economies move ahead nuclear plans"	#match
ctx-DocNADEe	#IR1 :: imf sign lifting japan yen	YES
	#IR2 :: japan recovery takes hold debt downgrade looms	YES
	#IR3 :: japan ministers confident treasuries move	YES
DocNADE	#IR1 :: nuclear regulator back power plans	NO
	#IR2 :: defiant iran plans big rise nuclear	NO
	#IR3 :: japan banks billion nuclear operator sources	YES

Table 9: Illustration of the top-3 retrievals for an input query

Additionally, we show the quality of representations learned at different fractions (20%, 40%, 60%, 80%, 100%) of training set from TMNtitle data and use the same experimental setup for the IR and classification tasks, as in section 3.3. In Figure 4, we quantify the quality of representations learned and demonstrate improvements due to the proposed models, i.e., ctx-DocNADE and ctx-DocNADEe over DocNADE at different fractions of the training data. Observe that the gains in both the tasks are large for smaller fractions of the datasets. For instance, one of the proposed models, i.e., ctx-DocNADEe (vs DocNADE) reports: (1) a precision (at 0.02 fraction) of 0.580 vs 0.444 at 20% and 0.595 vs 0.525 at 100% of the training set, and (2) an F1 of 0.711 vs 0.615 at 20% and 0.726 vs 0.688 at 100% of the training set. Therefore, the findings conform to our second contribution of improving topic models with word embeddings, especially in the sparse data setting.

3.5 Conclusion

In this work, we have shown that accounting for language concepts such as word ordering, syntactic and semantic information in neural autoregressive topic models helps to better estimate the probability of a word in a given context. To this end, we have combined a neural autoregressive topic- (i.e., DocNADE) and a neural language (e.g., LSTM-LM) model in a single probabilistic framework with an aim to introduce language concepts in each of the autoregressive steps of the topic model. This facilitates learning a latent representation from the entire document whilst accounting for the local dynamics of the collocation patterns, encoded in the internal states of LSTM-LM. We further augment this complementary learning with external knowledge by introducing word embeddings. Our experimental results show that our proposed modeling approaches consistently outperform state-of-the-art generative topic models, quantified by generalization (perplexity), topic interpretability (coherence), and applicability (text retrieval and categorization) on 15 datasets.

REFERENCES

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. *Journal of machine learning research*, 3(Feb):1137–1155, 2003.

D. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. pp. 993–1022, 2003.

Ziqiang Cao, Sujian Li, Yang Liu, Wenjie Li, and Heng Ji. A novel neural topic model and its supervised extension. In *AAAI*, pp. 2210–2216, 2015.

Dallas Card, Chenhao Tan, and Noah A Smith. A neural framework for generalized topic models. *arXiv preprint arXiv:1705.09296*, 2017.

Jonathan Chang, Jordan Boyd-Graber, Chong Wang, Sean Gerrish, and David M. Blei. Reading tea leaves: How humans interpret topic models. In *In Neural Information Processing Systems (NIPS)*, 2009.

Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pp. 795–804, 2015.

Adji B. Dieng, Chong Wang, Jianfeng Gao, and John William Paisley. Topicrnn: A recurrent neural network with long-range semantic dependency. *CoRR*, abs/1611.01702, 2016.

Pankaj Gupta, Subburam Rajaram, Hinrich Schütze, and Bernt Andrassy. Deep temporal-recurrent-replicated-softmax for topical trends over time. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pp. 1079–1089, New Orleans, USA, 2018. Association of Computational Linguistics.

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted boltzmann machines. In *Proceedings of the 25th international conference on Machine learning*, pp. 536–543. ACM, 2008.

Hugo Larochelle and Stanislas Lauly. A neural autoregressive topic model. In *Proceedings of the Advances in Neural Information Processing Systems 25 (NIPS 2012)*. NIPS, 2012.

Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 29–37, 2011.

Jey Han Lau, Timothy Baldwin, and Trevor Cohn. Topically driven neural language model. In *ACL*, 2017.

Stanislas Lauly, Yin Zheng, Alexandre Allauzen, and Hugo Larochelle. Document neural autoregressive distribution estimation. *Journal of Machine Learning Research*, 18(113):1–24, 2017. URL `http://jmlr.org/papers/v18/16-017.html`.

Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. pp. 1188–1196, 2014.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*, 2010.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Workshop Track of the 1st International Conference on Learning Representations (ICLR 2013)*, 2013.

David Newman, Sarvnaz Karimi, and Lawrence Cavedon. External evaluation of topic models. In *Proceedings of the 14th Australasian Document Computing Symposium*, 2009.

Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics*, 3:299–313, 2015.

Kamal Nigam, Andrew Kachites McCallum, Sebastian Thrun, and Tom Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3):103–134, 2000.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1532–1543, 2014. URL http://aclweb.org/anthology/D/D14/D14-1162.pdf.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/N18-1202.

James Petterson, Wray Buntine, Shravan M Narayanamurthy, Tibério S Caetano, and Alex J Smola. Word features for latent dirichlet allocation. In *Advances in Neural Information Processing Systems*, pp. 1921–1929, 2010.

Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. In *Proceedings of the WSDM*. ACM, 2015.

Mehran Sahami and Timothy D Heilman. A web-based kernel function for measuring the similarity of short text snippets. In *Proceedings of the 15th international conference on World Wide Web*, pp. 377–386. AcM, 2006.

Haşim Sak, Andrew Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In *Fifteenth annual conference of the international speech communication association*, 2014.

Ruslan Salakhutdinov and Geoffrey Hinton. Replicated softmax: an undirected topic model. In *Proceedings of the Advances in Neural Information Processing Systems 22 (NIPS 2009)*, pp. 1607–1614. NIPS, 2009.

Akash Srivastava and Charles Sutton. Autoencoding variational inference for topic models. In *International Conference on Learning Representations (ICLR)*, 2017.

Hanna M Wallach. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd international conference on Machine learning*, pp. 977–984. ACM, 2006.

Wenlin Wang, Zhe Gan, Wenqi Wang, Dinghan Shen, Jiaji Huang, Wei Ping, Sanjeev Satheesh, and Lawrence Carin. Topic compositional neural language model. In *AISTATS*, 2018.

Xuerui Wang, Andrew McCallum, and Xing Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *icdm*, pp. 697–702. IEEE, 2007.

Yin Zheng, Yu-Jin Zhang, and Hugo Larochelle. A deep and autoregressive approach for topic modeling of multimodal data. In *IEEE transactions on pattern analysis and machine intelligence*, pp. 1056–1069. IEEE, 2016.

Label: *training*
Instructors shall have tertiary education and experience in the operation and maintenance of the equipment or sub-system of Plant. They shall be proficient in the use of the English language both written and oral. They shall be able to deliver instructions clearly and systematically. The curriculum vitae of the instructors shall be submitted for acceptance by the Engineer at least 8 weeks before the commencement of any training.
Label: *maintenance*
The Contractor shall provide experienced staff for 24 hours per Day, 7 Days per week, throughout the Year, for call out to carry out On-call Maintenance for the Signalling System.
Label: *cables*
Unless otherwise specified, this standard is applicable to all cables which include single and multi-core cables and wires, Local Area Network (LAN) cables and Fibre Optic (FO) cables.
Label: *installation*
The Contractor shall provide and permanently install the asset labels onto all equipment supplied under this Contract. The Contractor shall liaise and co-ordinate with the Engineer for the format and the content of the labels. The Contractor shall submit the final format and size of the labels as well as the installation layout of the labels on the respective equipment, to the Engineer for acceptance.
Label: *operations, interlocking*
It shall be possible to switch any station Interlocking capable of reversing the service into "Auto-Turnaround Operation". This facility once selected shall automatically route Trains into and out of these stations, independently of the ATS system. At stations where multiple platforms can be used to reverse the service it shall be possible to select one or both platforms for the service reversal.

Table 10: SiROBs data: Example Documents (Requirement Objects) with their types (label).

A DATA DESCRIPTION

We use 14 different datasets: (1) `20NSshort`: We take documents from 20NewsGroups data, with document size less (in terms of number of words) than 20. (2) `TREC6`: a set of questions (3) `Reuters21578title`: a collection of new stories from `nltk.corpus`. We take titles of the documents. (4) `Subjectivity`: sentiment analysis data. (5) `Polarity`: a collection of positive and negative snippets acquired from Rotten Tomatoes (6) `TMNtitle`: Titles of the Tag My News (TMN) news dataset. (7) `AGnewstitle`: Titles of the AGnews dataset. (8) `Reuters8`: a collection of news stories, processed and released by (9) `Reuters21578`: a collection of new stories from `nltk.corpus`. (10) `20NewsGroups`: a collection of news stories from `nltk.corpus`. (11) RCV1V2 (Reuters): `www.ai.mit.edu/projects/jmlr/papers/volume5/lewis04a/lyrl2004_rcv1v2_README.htm` (12) `20NSsmall`: We sample 20 document for training from each class of the 20NS dataset. For validation and test, 10 document for each class. (13) `TMN`: The Tag My News (TMN) news dataset. (14) `Sixxx Requirement OBjects (SiROBs)`: a collection of paragraphs extracted from industrial tender documents (our industrial corpus).

The SiROBs is our industrial corpus, extracted from industrial tender documents. The documents contain requirement specifications for an industrial project for example, *railway metro construction*. There are 22 types of requirements i.e. class labels (multi-class), where a requirement is a paragraph or collection of paragraphs within a document. We name the requirement as Requirement Objects (ROBs). Some of the requirement types are *project management*, *testing*, *legal*, *risk analysis*, *financial cost*, *technical requirement*, etc. We need to classify the requirements in the tender documents and assign each ROB to a relevant department(s). Therefore, we analyze such documents to automate decision making, tender comparison, similar tender as well as ROB retrieval and assigning ROBs to a relevant department(s) to optimize/expedite tender analysis. See some examples of ROBs from SiROBs corpus in Table 10.

Hyperparameter	Search Space
learning rate	[0.001]
hidden units	[200]
iterations	[2000]
activation function	sigmoid
λ	[1.0, 0.8, 0.5, 0.3, 0.1, 0.01, 0.001]

Table 11: Hyperparameters in Generalization in the DocNADE and ctx-DocNADE variants for 200 topics

Hyperparameter	Search Space
retrieval fraction	[0.02]
learning rate	[0.001]
hidden units	[200]
activation function	tanh
iterations	[2000]
λ	[1.0, 0.8, 0.5, 0.3, 0.1, 0.01, 0.001]

Table 12: Hyperparameters in the Document Retrieval task.

B EXPERIMENTAL SETUP

B.1 EXPERIMENTAL SETUP AND HYPERPARAMETERS FOR GENERALIZATION TASK

See Table 11 for hyperparameters used in generalization.

B.2 EXPERIMENTAL SETUP AND HYPERPARAMETERS FOR IR TASK

We set the maximum number of training passes to 1000, topics to 200 and the learning rate to 0.001 with $tanh$ hidden activation. For model selection, we used the validation set as the query set and used the average precision at 0.02 retrieved documents as the performance measure. Note that the labels are not used during training. The class labels are only used to check if the retrieved documents have the same class label as the query document. To perform document retrieval, we use the same train/development/test split of documents discussed in data statistics (experimental section) for all the datasets during learning.

See Table 12 for the hyperparameters in the document retrieval task.

B.3 EXPERIMENTAL SETUP FOR DOC2VEC MODEL

We used gensim (`https://github.com/RaRe-Technologies/gensim`) to train Doc2Vec models for 12 datasets. Models were trained with distributed bag of words, for 1000 iterations using a window size of 5 and a vector size of 500.

B.4 CLASSIFICATION TASK

We used the same split in training/development/test as for training the Doc2Vec models (also same split as in IR task) and trained a regularized logistic regression classifier on the inferred document vectors to predict class labels. In the case of multilabel datasets (`R21578`,`R21578title`, `RCV1V2`), we used a one-vs-all approach. Models were trained with a liblinear solver using L2 regularization and accuracy and macro-averaged F1 score were computed on the test set to quantify predictive power.

B.5 EXPERIMENTAL SETUP FOR GLOVE-DMM AND GLOVE-LDA MODELS

We used LFTM (`https://github.com/datquocnguyen/LFTM`) to train glove-DMM and glove-LDA models. Models were trained for 200 iterations with 2000 initial iterations using 200 topics. For short texts we set the hyperparameter beta to 0.1, for long texts to 0.01; the mixture parameter lambda was set to 0.6 for all datasets. The setup for the classification task was the same as

Dataset	Model	λ		
		1.0	0.1	0.01
20NSshort	ctx-DocNADE	899.04	829.5	842.1
	ctx-DocNADEe	890.3	828.8	832.4
Subjectivity	ctx-DocNADE	982.8	977.8	966.5
	ctx-DocNADEe	977.1	975.0	964.2
TMNtitle	ctx-DocNADE	1898.1	1482.7	1487.1
	ctx-DocNADEe	1877.7	1480.2	1484.7
AGnewstitle	ctx-DocNADE	1296.1	861.1	865
	ctx-DocNADEe	1279.2	853.3	862.9
Reuters-8	ctx-DocNADE	336.1	313.2	311.9
	ctx-DocNADEe	323.3	312.0	310.2
20NS	ctx-DocNADE	1282.1	1209.3	1207.2
	ctx-DocNADEe	1247.1	1211.6	1206.1

Table 13: Perplexity scores for different λ in Generalization task: Ablation over validation set

Dataset	Model	λ			
		1.0	0.8	0.5	0.3
20NSshort	ctx-DocNADE	0.264	0.265	0.265	0.265
	ctx-DocNADEe	0.277	0.277	0.278	0.276
Subjectivity	ctx-DocNADE	0.874	0.874	0.873	0.874
	ctx-DocNADEe	0.868	0.868	0.874	0.87
Polarity	ctx-DocNADE	0.587	0.588	0.591	0.587
	ctx-DocNADEe	0.602	0.603	0.601	0.599
TMNtitle	ctx-DocNADE	0.556	0.557	0.559	0.568
	ctx-DocNADEe	0.604	0.604	0.6	0.6
TMN	ctx-DocNADE	0.683	0.689	0.692	0.694
	ctx-DocNADEe	0.696	0.698	0.698	0.7
AGnewstitle	ctx-DocNADE	0.665	0.668	0.678	0.689
	ctx-DocNADEe	0.686	0.688	0.695	0.696
20NSsmall	ctx-DocNADE	0.352	0.356	0.366	0.37
	ctx-DocNADEe	0.381	0.381	0.375	0.353
Reuters-8	ctx-DocNADE	0.863	0.866	0.87	0.87
	ctx-DocNADEe	0.875	0.872	0.873	0.872
20NS	ctx-DocNADE	0.503	0.506	0.513	0.512
	ctx-DocNADEe	0.524	0.521	0.518	0.511
R21578	ctx-DocNADE	0.714	0.714	0.714	0.714
	ctx-DocNADEe	0.715	0.715	0.715	0.714
SiROBs	ctx-DocNADE	0.409	0.409	0.408	0.408
	ctx-DocNADEe	0.41	0.411	0.411	0.409
AGnews	ctx-DocNADE	0.786	0.789	0.792	0.797
	ctx-DocNADEe	0.795	0.796	0.8	0.799

Table 14: λ for IR task: Ablation over validation set at retrieval fraction 0.02

for doc2vec; classification was performed using relative topic proportions as input (i.e. we inferred the topic distribution of the training and test documents and used the relative distribution as input for the logistic regression classifier). Similarly, for the IR task, similarities were computed based on the inferred relative topic distribution.

B.6 Experimental Setup for ProdLDA

We run ProdLDA (`https://github.com/akashgit/autoencoding_vi_for_topic_models`) on the short-text datasets in the FV setting to generate document vectors for

IR-task. We use 200 topics for a fair comparison with other baselines used for the IR tasks. We infer topic distribution of the training and test documents and used the relative distribution as input for the IR task, similar to section 3.3.

To fairly compare PPL scores of ProdLDA and DocNADE in the RV setting, we take the pre-processed 20NS dataset released by ProdLDA and run DocNADE for 200 topics. To further compare them in the FV setting, we run ProdLDA (`https://github.com/akashgit/autoencoding_vi_for_topic_models`) on the processed 20NS dataset for 200 topics used in this paper.

C Ablation over the mixture weight λ

C.1 λ for Generalization task

See Table 13.

C.2 λ for IR task

See Table 14.

D Additional Baselines

D.1 DocNADE vs SCHOLAR

PPL scores over 20 topics: DocNADE (752) and SCHOLAR (921), i.e., DocNADE outperforms SCHOLAR in terms of generalization.

Topic coherence (NPMI) using 20 topics: DocNADE (.18) and SCHOLAR (.35), i.e., SCHOLAR (Card et al., 2017) generates more coherence topics than DocNADE, though worse in PPL and text classification (see section 3.3) than DocNADE, ctx-DocNADE and ctx-DocNADEe.

IR tasks: Since, SCHOLAR (Card et al., 2017) without meta-data equates to ProdLDA and we have shown in section 3.3 that ProdLDA is worse on IR tasks than our proposed models, therefore one can infer the performance of SCHOLAR on IR task.

The experimental results above suggest that the DocNADE is better than SCHOLAR in generating good representations for downstream tasks such as information retrieval or classification, however falls behind SCHOLAR in interpretability. The investigation opens up an interesting direction for future research.

Chapter 9

Replicated Siamese LSTM in Ticketing System for Similarity Learning and Retrieval in Asymmetric Texts

Replicated Siamese LSTM in Ticketing System for Similarity Learning and Retrieval in Asymmetric Texts

Pankaj Gupta[1,2] **Bernt Andrassy**[1] **Hinrich Schütze**[2]
[1]Corporate Technology, Machine-Intelligence (MIC-DE), Siemens AG Munich, Germany
[2]CIS, University of Munich (LMU) Munich, Germany
pankaj.gupta@siemens.com | bernt.andrassy@siemens.com
pankaj.gupta@campus.lmu.de | inquiries@cis.lmu.de

Abstract

The goal of our industrial ticketing system is to retrieve a relevant solution for an input query, by matching with historical tickets stored in knowledge base. A query is comprised of subject and description, while a historical ticket consists of subject, description and solution. To retrieve a relevant solution, we use textual similarity paradigm to learn similarity in the query and historical tickets. The task is challenging due to significant term mismatch in the query and ticket pairs of asymmetric lengths, where subject is a short text but description and solution are multi-sentence texts. We present a novel Replicated Siamese LSTM model to learn similarity in asymmetric text pairs, that gives 22% and 7% gain (Accuracy@10) for retrieval task, respectively over unsupervised and supervised baselines. We also show that the topic and distributed semantic features for short and long texts improved both similarity learning and retrieval.

1 Introduction

Semantic Textual Similarity (STS) is the task to find out if the text pairs mean the same thing. The important tasks in Natural Language Processing (NLP), such as Information Retrieval (IR) and text understanding may be improved by modeling the underlying semantic similarity between texts.

With recent progress in deep learning, the STS task has gained success using LSTM (Mueller and Thyagarajan, 2016) and CNN (Yin et al., 2016) based architectures; however, these approaches model the underlying semantic similarity between example pairs, each with a single sentence or phrase with term overlaps. In the domain of question retrieval (Cai et al., 2011; Zhang et al., 2014), users retrieve historical questions which precisely match their questions (single sentence) semantically equivalent or relevant. However, we investigate similarity learning between texts of asymmetric lengths, such as short (phrase) Vs longer (paragraph/documents) with significant term mismatch. The application of textual understanding in retrieval becomes more challenging when the relevant document-sized retrievals are stylistically distinct with the input short texts. Learning a similarity metric has gained much research interest, however due to limited availability of labeled data and complex structures in variable length sentences, the STS task becomes a hard problem. The performance of IR system is sub-optimal due to significant term mismatch in similar texts (Zhao, 2012), limited annotated data and complex structures in variable length sentences. We address the challenges in a real-world industrial application.

Our ticketing system (Figure 1(a)) consists of a query and historical tickets (Table 1). A query (reporting issue, q) has 2 components: *subject* (SUB) and *description* (DESC), while a historical ticket ($\mathfrak{t}$) stored in the knowledge base (KB) has 3 components: SUB, DESC and *solution* (SOL). A SUB is a short text, but DESC and SOL consist of multiple sentences. Table 1 shows that SUB $\in q$ and SUB $\in \mathfrak{t}$ are semantically similar and few terms in SUB $\in q$ overlap with DESC $\in \mathfrak{t}$. However, the expected SOL $\in \mathfrak{t}$ is distinct from both SUB and DESC $\in q$. The goal is to retrieve an optimal action (i.e. SOL from $\mathfrak{t}$) for the input q.

To improve retrieval for an input q, we adapt the Siamese LSTM (Mueller and Thyagarajan, 2016) for similarity learning in asymmetric text pairs, using the available information in q and $\mathfrak{t}$. For instance,

QUERY (q)

SUB: GT Trip - Low Frequency Pulsations

DESC: GT Tripped due to a sudden increase in Low Frequency Pulsations. The machine has been restarted and is now operating normally. Alarm received was: GT XXX Low Frequency Pulsation.

HISTORICAL TICKET (t)

SUB: Narrow Frequency Pulsations

DESC: Low and Narrow frequency pulsations were detected. The peak value for the Low Frequency Pulsations is ## mbar.

SOL: XXXX combustion support is currently working on the issue. The action is that the machine should not run until resolved.

Table 1: Example of a Query and Historical Ticket

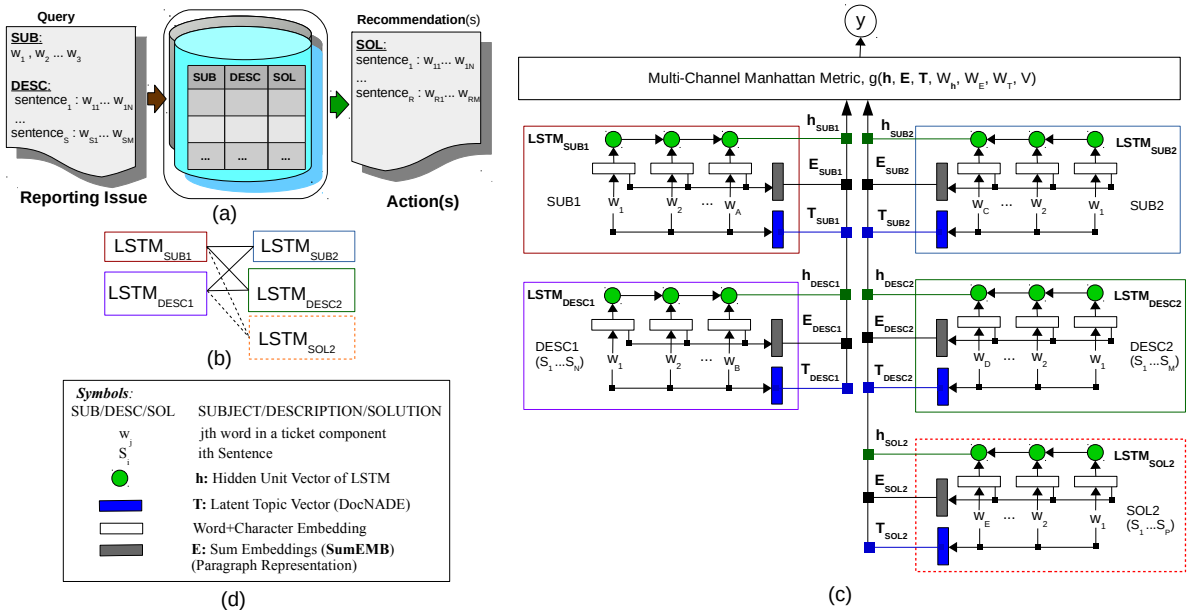


Figure 1: (a): Intelligent Ticketing System (ITS) (b): High-level illustration of Siamese LSTM for cross-level pairwise similarity. (c): Replicated Siamese with multi-channel (SumEMB, LSTM and topic vectors) and multi-level (SUB, DESC and/or SOL) inputs in the objective function, g. y: similarity score. The dotted lines indicate ITS output. (d): Symbols used.

we compute *multi-level* similarity between (SUB $\in q$, SUB $\in$ t) and (DESC $\in q$, DESC $\in$ t). However, observe in Table 1 that the *cross-level* similarities such as between (SUB $\in q$, DESC $\in$ t), (DESC $\in q$, SUB $\in$ t) or (SUB $\in q$, SOL $\in$ t), etc. can supplement IR performance. See Figure 1(b).

The *contributions* of this paper are as follows: (1) Propose a novel architecture (Replicated Siamese LSTM) for similarity learning in asymmetric texts via multi-and-cross-level semantics (2) Investigate distributed and neural topic semantics for similarity learning via multiple channels (3) Demonstrate a gain of 22% and 7% in Accuracy@10 for retrieval, respectively over unsupervised and supervised baselines in the industrial application of a ticketing system.

2 Methodology

Siamese networks (Chopra et al., 2005) are dual-branch networks with tied weights and an objective function. The aim of training is to learn text pair representations to form a highly structured space where they reflect complex semantic relationships. Figure 1 shows the proposed Replicated Siamese neural network architecture such that ($\text{LSTM}_{\text{SUB1}}$+$\text{LSTM}_{\text{DESC1}}$) = ($\text{LSTM}_{\text{SUB2}}$+$\text{LSTM}_{\text{DESC2}}$+$\text{LSTM}_{\text{SOL2}}$), to learn similarities in asymmetric texts, where a query (SUB1+DESC1) is stylistically distinct from a historical ticket (SUB2+DESC2+SOL2).

Note, the *query components are suffixed by "1" and historical ticket components by "2"* in context of the following work for pairwise comparisons.

$$g(h, E, T, W_h, W_E, W_T, V) =$$
$$\exp\Big(-\sum_{p\ \in\ \{SUB1, DESC1\}}\ \sum_{q\ \in\ \{SUB2, DESC2, SOL2\}} V_{\{p,q\}}\ \big(W_h\|h_p - h_q\|_1 + W_E\|E_p - E_q\|_1 + W_T\|T_p - T_q\|_1\big)\Big) \quad (1)$$

Figure 2: Multi-Channel Manhattan Metric

2.1 Replicated, Multi-and-Cross-Level, Multi-Channel Siamese LSTM

Manhattan LSTM (Mueller and Thyagarajan, 2016) learns similarity in text pairs, each with a single sentence; however, we advance the similarity learning task in asymmetric texts pairs consisting of one or more sentences, where similarity is computed between different-sized subject and description or solution texts. As the backbone of our work, we compute similarity scores to learn a highly structured space via LSTM (Hochreiter and Schmidhuber, 1997) for representation of each pair of the query (SUB1 and DESC1) or historical ticket (SUB2, DESC2 and SOL2) components, which includes multi-level (SUB1-SUB2, DESC1-DESC2) and cross-level (SUB1-DESC2, SUB1-SOL2, etc.) asymmetric textual similarities, Figure 1(b) and (c). To accumulate the semantics of variable-length sentences $(x_1, ..., x_T)$, recurrent neural networks (RNNs) (Vu et al., 2016a; Gupta et al., 2016; Gupta and Andrassy, 2018), especially the LSTMs (Hochreiter and Schmidhuber, 1997) have been successful.

LSTMs are superior in learning long range dependencies through their memory cells. Like the standard RNN (Mikolov et al., 2010; Gupta et al., 2015a; Vu et al., 2016b), LSTM sequentially updates a hidden-state representation, but it introduces a memory state c_t and three gates that control the flow of information through the time steps. An output gate o_t determines how much of c_t should be exposed to the next node. An input gate i_t controls how much the input x_t be stored in memory, while the forget gate f_t determines what should be forgotten from memory. The dynamics:

$$\begin{aligned} i_t &= \sigma(W_i x_t + U_i h_{t-1}) \\ f_t &= \sigma(W_f x_t + U_f h_{t-1}) \\ o_t &= \sigma(W_o x_t + U_o h_{t-1}) \\ \tilde{c}_t &= tanh(W_c x_t + U_c h_{t-1}) \\ c_t &= i_t \odot \tilde{c}_t + f_t \odot c_{t-1} \\ h_t &= o_t \odot tanh(c_t) \end{aligned} \quad (2)$$

where $\sigma(x) = \frac{1}{1+e^{-x}}$ and $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$. The proposed architecture, Figure 1(c) is composed of multiple uni-directional LSTMs each for subject, description and solution within the Siamese framework, where the weights at over levels are shared between the left and right branch of the network. Therefore, the name *replicated*.

Each LSTM learns a mapping from space of variable length sequences, including asymmetric texts, to a hidden-state vector, h. Each sentence $(w_1, ...w_T)$ is passed to LSTM, which updates hidden state via eq 2. A final encoded representation (e.g. h_{SUB1}, h_{SUB2} in Figure 1(c)) is obtained for each query or ticket component. A single LSTM is run over DESC and SOL components, consisting of one or more sentences. Therefore, the name *multi-level* Siamese.

The representations across the text components (SUB DESC or SOL) are learned in order to maximize the similarity and retrieval for a query with the historical tickets. Therefore, the name *cross-level* Siamese.

The sum-average strategy over word embedding (Mikolov et al., 2010) for short and longer texts has demonstrated a strong baseline for text classification (Joulin et al., 2016) and pairwise similarity learning (Wieting et al., 2016). This simple baseline to represent sentences as bag of words (BoW) inspires us to use the BoW for each query or historical ticket component, for instance E_{SUB1}. We refer the approach as *SumEMB* in the context of this paper.

We supplement the similarity metric (g) with *SumEMB* (E), latent topic (T) (section 2.2) and hidden vectors (h) of LSTM for each text component from both the Siamese branches. Therefore, the name *multi-channel* Siamese.

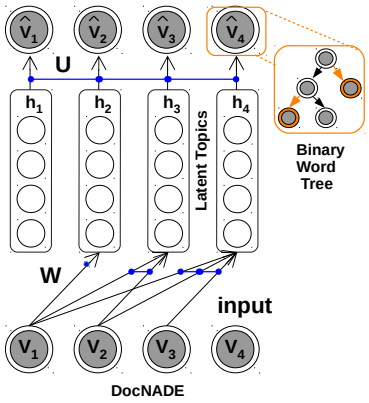


Figure 3: DocNADE: Neural Auto-regressive Topic Model

Parameter	*Search*	*Optimal*
E	[350]	350
T	[20, 50, 100]	100
h	[50, 100]	50
W_h	[0.6, 0.7, 0.8]	0.7
W_E	[0.3, 0.2, 0.1]	0.1
W_T	[0.3, 0.2, 0.1]	0.2
$V_{SUB1-SUB2}$	[0.3, 0.4	0.3
$V_{DESC1-DESC2}$	[0.3, 0.4]	0.3
$V_{SUB1-DESC2}$	[0.10, 0.15, 0.20]	0.20
$V_{SUB1-SOL2}$	[0.10, 0.15, 0.20]	0.10
$V_{DESC1-SOL2}$	[0.10, 0.15, 0.20]	0.10

Table 2: Hyperparameters in the Replicated Siamese LSTM (experiment #No:22)

2.2 Neural Auto-Regressive Topic Model

Topic models such as Latent Dirichlet allocation (LDA) (Blei et al., 2003) and Replicated Softmax (RSM) (Hinton and Salakhutdinov, 2009; Gupta et al., 2018c) have been popular in learning meaningful representations of unlabeled text documents. Recently, a new type of topic model called the Document Neural Autoregressive Distribution Estimator (DocNADE) (Larochelle and Lauly, 2012; Zheng et al., 2016; Gupta et al., 2018a) was proposed and demonstrated the state-of- the-art performance for text document modeling. DocNADE models are advanced variants of Restricted Boltzmann Machine (Hinton, 2002; Salakhutdinov et al., 2007; Gupta et al., 2015b; Gupta et al., 2015c), and have shown to outperform LDA and RSM in terms of both log-likelihood of the data and document retrieval. In addition, the training complexity of a DocNADE model scales logarithmically with vocabulary size, instead linear as in RSM. The features are important for an industrial task along with quality performance. Therefore, we adopt DocNADE model for learning latent representations of tickets and retrieval in unsupervised fashion. See Larochelle and Lauly (2012) and Gupta et al. (2018a) for further details, and Figure 3 for the DocNADE architecture, where we extract the last hidden topic layer (h_4) to compute document representation.

2.3 Multi-Channel Manhattan Metric

Chopra et al. (2005) indicated that using l_2 instead of l_1 norm in similarity metric can lead to undesirable plateaus. Mueller and Thyagarajan (2016) showed stable and improved results using Manhattan distance over cosine similarity.

Mueller and Thyagarajan (2016) used a Manhattan metric (l_1-norm) for similarity learning in single sentence pairs. However, we adapt the similarity metric for 2-tuple (SUB1, DESC1) vs 3-tuple (SUB2, DESC2 and SOL2) pairs, where the error signals are back-propagated in the multiple levels and channels during training to force the Siamese network to entirely capture the semantic differences across the query and historical tickets components. The similarity metric, $g \in [0,1]$ is given in eq 1, where $||\cdot||$ is l_1 norm. W_h, W_E and W_T are the three channels weights for h, E and and T, respectively. The weights (V) are the multi-level weights between the ticket component pairs. Observe that a single weight is being used in the ordered ticket component pairs, for instance $V_{SUB1-DESC2}$ is same as $V_{DESC2-SUB1}$.

3 Evaluation and Analysis

We evaluate the proposed method on our industrial data for textual similarity learning and retrieval tasks in the ticketing system. Table 4 shows the different model configurations used in the following exper-

Held-out	*Perplexity* (100 topics)			
Ticket	$M1$: SUB+DESC		$M2$: SUB+DESC+SOL	
Component	LDA	DocNADE	LDA	DocNADE
DESC	380	**362**	565	**351**
SUB+DESC	480	**308**	515	**289**
SUB+DESC+SOL	553	**404**	541	**322**

(a)

Query Component	Perplexity (100 topics)			
	DocNADE:**M1**		DocNADE:**M2**	
	$\|Q\|_L$	$\|Q\|_U$	$\|Q\|_L$	$\|Q\|_U$
DESC1	192	177	132	118
SUB1+DESC1	164	140	130	118

(b)

Table 3: **(a)** Perplexity by DocNADE and LDA trained with $M1$: SUB+DESC or $M2$: SUB+DESC+SOL on all tickets and evaluated on 50 held-out tickets with their respective components or their combination. Observe that when DocNADE is trained with SUB+DESC+SOL, it performs better when training with SUB+DESC+SOL and outperforms LDA. **(b)** Perplexity by DocNADE: $M1$ trained on SUB+ DESC and $M2$ on SUB+DESC+SOL of the historical tickets.

Model	Model Configuration
T ($X1$–$X2$)	Compute Similarity using topic vector (*T*) pairs of a query ($X1$) and historical ticket ($X2$) components
E ($X1$–$X2$)	Compute Similarity using embedding vector (*E*) pairs of a query ($X1$) and historical ticket ($X2$) components
$X + Y + Z$	Merge text components (SUB, DESC or SOL), representing a single document
T ($X1 + Y1$–$X2 + Y2 + Z2$)	Compute Similarity using topic vector (*T*) pairs of a query ($X1 + Y1$) and historical ticket ($X2 + Y2 + Z2$) components
S-LSTM ($X1$–$X2$)	Compute Similarity using Standard Siamese LSTM on a query ($X1$) and historical ticket ($X2$) components
ML ($X1$–$X2$, $Y1$–$Y2$)	Multi-level Replicated Siamese LSTM. Compute similarity in ($X1$–$X2$) and ($Y1$–$Y2$) components of a query and historical ticket
CL (X, Y, Z)	Cross-level Replicated Siamese LSTM. Compute similarity in ($X1$–$Y2$), ($X1$–$Z2$), ($Y1$–$X2$) and ($Y1$–$Z2$) pairs

Table 4: Different model configurations for the experimental setups and evaluations. See Figure 1(c) for LSTM configurations.

imental setups. We use Pearson correlation, Spearman correlation and Mean Squared Error[1] (MSE) metrics for STS and 9 different metrics (Table 5) for IR task.

3.1 Industrial Dataset for Ticketing System

Our industrial dataset consist of queries and historical tickets. As shown in Table 1, a query consists of *subject* and *description* texts, while a historical ticket in knowledge base (KB) consists of *subject*, *description* and *solution* texts. The goal of the ITS is to automatically recommend an optimal action i.e. *solution* for an input query, retrieved from the existing KB.

There are $\mathfrak{T} = 949$ historical tickets in the KB, out of which 421 pairs are labeled with their relatedness score. We randomly split the labeled pairs by 80-20% for train (P_{tr}) and development (P_{dev}). The relatedness labels are: *YES* (similar that provides correct solution), *REL* (does not provide correct solution, but close to a solution) and *NO* (not related, not relevant and provides no correct solution). We convert the labels into numerical scores [1,5], where *YES*:5.0, *REL*:3.0 and *NO*:1.0. The average length (#words) of SUB, DESC and SOL are 4.6, 65.0 and 74.2, respectively.

The end-user (customer) additionally supplies 28 unique queries (Q_U) (exclusive to the historical tickets) to test system capabilities to retrieve the optimal solution(s) by computing 28×949 pairwise ticket similarities. We use these queries for the end-user qualitative evaluation for the 28×10 proposals (top 10 retrievals for each query).

3.2 Experimental Setup: Unsupervised

We establish baseline for similarity and retrieval by the following two unsupervised approaches:

(1) Topic Semantics T: As discussed in section 2.2, we use DocNADE topic model to learn document representation. To train, we take 50 held-out samples from the historical tickets $\mathfrak{T}$. We compute perplexity on 100 topics for each ticket component from the held-out set, comparing LDA and DocNADE models trained individually with SUB+DESC ($M1$) and SUB+DESC+SOL texts[2] ($M2$). Table 3a shows that DocNADE outperforms LDA.

[1] http://alt.qcri.org/semeval2016/task1/

[2] +: merge texts to treat them as a single document

#No	Model (Query-Historical Ticket)	Similarity Task			Retrieval Task								
		r	ρ	MSE	MAP@1	MAP@5	MAP@10	MRR@1	MRR@5	MRR@10	Acc@1	Acc@5	Acc@10
1	T (SUB1–SUB2) (**unsupervised baseline**)	0.388	0.330	5.122	0.08	0.08	0.07	1.00	0.28	0.10	0.04	0.19	0.30
2	T (SUB1–DESC2)	0.347	0.312	3.882	0.09	0.07	0.07	0.00	0.05	0.08	0.04	0.13	0.21
3	T (DESC1–SUB2)	0.321	0.287	3.763	0.08	0.09	0.09	0.00	0.05	0.11	0.03	0.20	0.31
4	T (DESC1–DESC2)	0.402	0.350	3.596	0.08	0.08	0.08	0.00	0.04	0.10	0.03	0.19	0.33
5	T (SUB1–SUB2+DESC2)	0.413	0.372	3.555	0.09	0.09	0.08	0.00	0.05	0.11	0.04	0.20	0.32
6	T (SUB1+DESC1–SUB2)	0.330	0.267	3.630	0.09	0.10	0.09	0.00	0.26	0.12	0.04	0.23	0.35
7	T (SUB1+DESC1–DESC2)	0.400	0.350	3.560	0.07	0.08	0.08	0.00	0.00	0.10	0.03	0.19	0.35
8	T (SUB1+DESC1–SUB2+DESC2)	0.417	0.378	3.530	0.05	0.07	0.08	0.00	0.07	0.11	0.03	0.22	0.37
9	T (SUB1+DESC1–SUB2+DESC2+SOL2)	0.411	0.387	3.502	0.09	0.09	0.08	0.00	0.06	0.12	0.04	0.20	0.40
11	E (SUB1–SUB2) (**unsupervised baseline**)	0.141	0.108	3.636	0.39	0.38	0.36	0.00	0.03	0.08	0.02	0.13	0.24
12	E (DESC1–DESC2)	0.034	0.059	4.201	**0.40**	**0.40**	**0.39**	0.00	0.10	0.07	0.03	0.12	0.18
13	E (SUB1+DESC1–SUB2+DESC2)	0.103	0.051	5.210	0.16	0.16	0.15	0.00	0.03	0.11	0.07	0.16	0.20
14	E (SUB1+DESC1–SUB2+DESC2+SOL2)	0.063	0.041	5.607	0.20	0.17	0.16	0.00	0.03	0.13	0.05	0.13	0.22
15	S-LSTM(SUB1–SUB2) (**supervised baseline**)	0.530	0.501	3.778	0.272	0.234	0.212	0.000	0.128	0.080	0.022	0.111	0.311
16	S-LSTM (DESC1–DESC2)	0.641	0.586	3.220	0.277	0.244	0.222	0.100	**0.287**	0.209	0.111	0.3111	0.489
17	S-LSTM (SUB1+DESC1–SUB2+DESC2)	0.662	0.621	2.992	0.288	0.251	0.232	0.137	0.129	0.208	0.111	0.342	0.511
18	S-LSTM (SUB1+DESC1–SUB2+DESC2+SOL2)	0.693	0.631	2.908	0.298	0.236	0.241	0.143	0.189	0.228	0.133	0.353	0.548
19	ML-LSTM (SUB1–SUB2, DESC1–DESC2)	0.688	0.644	2.870	0.290	0.255	0.234	0.250	0.121	0.167	0.067	0.289	0.533
20	+ CL-LSTM (SUB, DESC, SOL)	0.744	0.680	2.470	0.293	0.259	0.238	0.143	0.179	0.286	**0.178**	0.378	0.564
21	+ weighted channels (h*0.8, E*0.2)	0.758	0.701	2.354	0.392	0.376	0.346	**0.253**	0.176	0.248	0.111	0.439	0.579
22	+ weighted channels (h*0.7, E*0.1, T*0.2)	**0.792**	**0.762**	**2.052**	0.382	0.356	0.344	0.242	0.202	**0.288**	0.133	**0.493**	**0.618**

Table 5: Results on Development set: Pearson correlation (*r*), Spearmans rank correlation coefficient (ρ), Mean Squared Error (MSE), Mean Average Precision@k (MAP@k), Mean Reciprocal Rank@k (MRR@k) and Accuracy@k (Acc@k) for the multi-level (ML) and cross-level (CL) similarity learning, and retrieving the k-most similar tickets for each query (SUB1+DESC1). **#[1-14]**: Unsupervised baselines with DocNADE (T) and SumEMB (E). **#[15-18]**: Supervised Standard Siamese baselines. **#[19-22]**: Supervised Replicated Siamese with multi-channel and cross-level features.

Model	Similarity Task			Retrieval Task								
	r	ρ	MSE	MAP@1	MAP@5	MAP@10	MRR@1	MRR@5	MRR@10	Acc@1	Acc@5	Acc@10
T (SUB1-SUB2)	0.414	0.363	5.062	0.04	0.03	0.03	**0.29**	0.24	0.10	0.01	0.17	0.28
T (SUB1-DESC2)	0.399	0.362	3.791	0.04	0.03	0.03	0.00	0.05	0.07	0.03	0.12	0.19
T (DESC1-SUB2)	0.371	0.341	3.964	0.05	0.06	0.05	0.25	0.07	0.11	0.04	0.21	0.33
T (DESC1-DESC2)	0.446	0.398	3.514	0.05	0.05	0.04	0.00	0.04	0.10	0.04	0.18	0.34
T (SUB1-SUB2+DESC2)	0.410	0.370	3.633	0.05	0.04	0.04	0.00	0.12	0.08	0.04	0.13	0.20
T (SUB1+DESC2-SUB2)	0.388	0.326	3.561	0.06	0.06	0.05	0.25	**0.29**	0.13	0.05	0.22	0.38
T (SUB1+DESC1-DESC2)	0.443	0.396	3.477	0.04	0.04	0.04	0.00	0.00	0.10	0.03	0.17	0.37
T (SUB1+DESC1, SUB2+DESC2)	**0.466**	**0.417**	3.460	0.05	0.05	0.04	0.00	0.06	0.11	0.03	**0.24**	0.37
T (SUB1+DESC1, SUB2+DESC2+SOL2)	0.418	0.358	**3.411**	**0.07**	0.06	**0.06**	0.00	0.09	**0.14**	0.05	0.20	**0.39**

Table 6: DocNADE ($M2$) performance for the queries $Q_L \in (P_{tr} + P_{dev})$ in the labeled pairs in unsupervised fashion.

Next, we need to determine which DocNADE model ($M1$ or $M2$) is less perplexed to the queries. Therefore, we use $M1$ and $M2$ to evaluate DESC1 and SUB1+DESC1 components of the two sets of queries: (1) Q_L is the set of queries from labeled (421) pairs and (2) Q_U is the end-user set. Table 3b shows that $M2$ performs better than $M1$ for both the sets of queries with DESC1 or SUB1+DESC1 texts. We choose $M2$ version of the DocNADE to setup baseline for the similarity learning and retrieval in unsupervised fashion.

To compute a similarity score for the given query q and historical ticket t where (q, t)$\in P_{dev}$, we first compute a latent topic vector (*T*) each for q and t using DocNADE ($M2$) and then apply the similarity metric g (eq 1). To evaluate retrieval for q, we retrieve the top 10 similar tickets, ranked by the similarity scores on their topic vectors. Table 5 (#No [1-9]) shows the performance of DocNADE for similarity and retrieval tasks. Observe that #9 achieves the best MSE (3.502) and Acc@10 (0.40) out of [1-9], suggesting that the topic vectors of query (SUB1+DESC1) and historical ticket (SUB2+DESC2+SOL2) are the key in recommending a relevant SOL2. See the performance of DocNADE for all labeled pairs i.e. queries and historical tickets ($P_{tr} + P_{dev}$) in the Table 6.

(2) Distributional Semantics E: Beyond topic models, we establish baseline using the SumEMB method (section 2.1), where an embedding vector *E* is computed following the topic semantics approach. The experiments #11-14 show that the SumEMB results in lower performance for both the tasks, suggesting a need of a supervised paradigm in order to learn similarities in asymmetric texts. Also, the comparison with DocNADE indicates that the topic features are important in the retrieval of tickets.

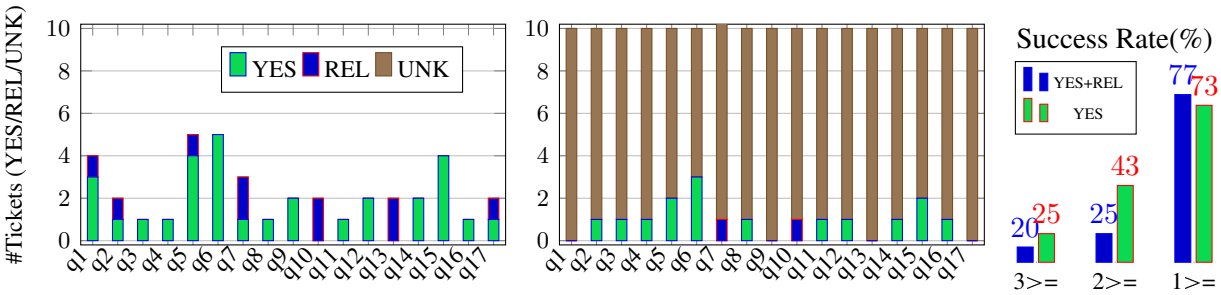


Figure 4: Evaluation on End-user Queries (sub-sample). UNK: Unknown. (Left) Gold Data: The count of similar (YES) and relevant (REL) tickets for each query (q1-q17). (Middle) ITS Results: For each query, ITS proposes the top 10 YES/REL retrievals. The plot depicts the count of YES/REL proposals matched out of the top 10 gold proposals for each q. UNK may include YES, REL or NO, not annotated in the gold pairs. (Right) Success Rate: YES: percentage of correct similar(YES) proposal out of the top 10; YES+REL: percentage of correct similar (YES) and relevant(REL) proposals out of the top 10.

3.3 Experimental Setup: Supervised

For semantic relatedness scoring, we train the Replicated Siamese, using backpropagation-through-time under the Mean Squared Error (MSE) loss function (after rescaling the training-set relatedness labels to lie $\in$ [0, 1]). After training, we apply an additional non-parametric regression step to obtain better-calibrated predictions $\in$ [1, 5], same as (Mueller and Thyagarajan, 2016). We then evaluate the trained model for IR task, where we retrieve the top 10 similar results (SUB2+DESC2+SOL2), ranked by their similarity scores, for each query (SUB1+DESC1) in the development set and compute MAP@K, MRR@K and Acc@K, where K=1, 5, and 10.

We use 300-dimensional pre-trained *word2vec*[3] embeddings for input words, however, to generalize beyond the limited vocabulary in *word2vec* due to industrial domain data with technical vocabulary, we also employ char-BLSTM (Lample et al., 2016) to generate additional embeddings (=50 dimension[4]).The resulting dimension for word embeddings is 350. We use 50-dimensional hidden vector, h_t, memory cells, c_t and Adadelta (Zeiler, 2012) with dropout and gradient clipping (Pascanu et al., 2013) for optimization. The topics vector (T) size is 100. We use python NLTK toolkit[5] for sentence tokenization. See Table 2 for the hyperparameters in Replicated Siamese LSTM for experiment #No:22.

3.4 Results: State-of-the-art Comparisons

Table 5 shows the similarity and retrieval scores for unsupervised and supervised baseline methods. The #9, #18 and #20 show that the supervised approach performs better than unsupervised topic models. #17 and #19 suggest that the multi-level Siamese improves (Acc@10: 0.51 vs. 0.53) both STS and IR. Comparing #18 and #20, the cross-level Siamese shows performance gains (Acc@10: 0.55 vs. 0.57). Finally, #21 and #22 demonstrates improved similarity (MSE: 2.354 vs. 2.052) and retrieval (Acc@10: 0.58 vs. 0.62) due to weighted multi-channel (h, E and T) inputs.

The replicated Siamese (#22) with different features best results in 2.052 for MSE and 0.618 (= 61.8%) for Acc@10. We see 22% and 7% gain in Acc@10 for retrieval task, respectively over unsupervised (#9 vs. #22: 0.40 vs. 0.62) and supervised (#18 vs. #22: 0.55 vs. 0.62) baselines. The experimental results suggest that the similarity learning in supervised fashion improves the ranking of relevant tickets.

3.5 Success Rate: End-User Evaluation

We use the trained similarity model to retrieve the top 10 similar tickets from KB for each end-user query Q_U, and compute the number of correct similar and relevant tickets. For ticket ID $q6$ (Figure 4, Middle),

[3]Publicly available at: code.google.com/p/word2vec
[4]Run forward-backward character LSTM for every word and concatenate the last hidden units (25 dimension each)
[5]http://www.nltk.org/api/nltk.tokenize.html

Query	Recommendation_1	Recommendation_2	Recommendation_3
SUB: GT Trip - Low Frequency Pulsations **DESC:** GT Tripped due to a sudden increase in Low Frequency Pulsations. The machine has been restarted and is now operating normally. Alarm received was: GT XXX Low Frequency Pulsation	**SUB:** Narrow Frequency Pulsations **DESC:** Low and Narrow frequency pulsations were detected. The peak value for the Low Frequency Pulsations is ## mbar. **SOL:** XXXXX combustion support is currently working on the issue. The recommended action for now is that the machine XXXX at **load** XXXX ## **MW**.	**SUB:** Low frequency pulsations **DESC:** High level low frequency pulsations were detected when active **load** is XXXX. **SOL:** Since the machine is running with XXXX, the XXX be changed in the register. After adjustment is complete, monitor the machine behavior between ## **MW** to ## **load**.	**SUB:** GT3 - High Low Frequency Pulsation alarms after trip **DESC:** Yesterday, after Steam Turbine tripped, GT-3 experienced high Low Frequency Pulsation alarm. The**load** of GT-3 was ## **MW** and went up as high as ## **MW**. During the time, Low Frequency Pulsation for 3 pulsation devices went up as high as ##. The Low frequency pulsation was a XXX. **SOL:** A **load** XXXX from ## **MW** to ## **MW** is an event XXX the unit XXXX trip. The XXXX to low frequency pulsation during similar event, should be XXXX. Check that XXXX from after the XXXX (XX005/XX01) into combustion chamber (XX030/XX01), XXXX should be XXXX. Repeat until XXXX is within the range of ## -##.
(Rank, Similarity Score)	(1, 4.75)	(2, 4.71)	(3, 4.60)
#Topics {**#83**, #7, #30}	{**#83**, #16, #30}	{#7, **#83**, #19}	{#7, **#83**, #19}

Table 7: Top-3 Tickets Retrieved and ordered by their (rank, similarity score) for an input test query. $\#Topics$: the top 3 most probable associated topics. **SOL** of the retrieved tickets is returned as recommended action. Underline: Overlapping words; XXXX and ##: Confidential text and numerical terms.

3 out of 10 proposed tickets are marked similar, where the end-user expects 4 similar tickets (Figure 4, Left). For ticket ID $q1$, $q13$ and $q17$, the top 10 results do not include the corresponding expected tickets due to no term matches and we find that the similarity scores for all the top 10 tickets are close to 4.0 or higher, which indicates that the system proposes more similar tickets (than the expected tickets), not included in the gold annotations. The top 10 proposals are evaluated for each query by success rate (success, if N/10 proposals supply the expected solution). We compute success rate (Figure 4, Right) for (1 or more), (2 or more) and (3 or more) correct results out of the top 10 proposals.

4 Qualitative Inspections for STS and IR

Table 7 shows a real example for an input query, where the top 3 recommendations are proposed from the historical tickets using the trained Replicated Siamese model. The recommendations are ranked by their similarity scores with the query. The underline shows the overlapping texts.

We also show the most probable topics (#) that the query or each recommendation is associated with. The topics shown (Table 8) are learned from DocNADE model and are used in multi-channel network. Observe that the improved retrieval scores (Table 5 #22) are attributed to the overlapping topic semantics in query and the top retrievals. For instance, the topic #83 is the most probable topic feature for the query and recommendations. We found terms, especially *load* and *MW* in SOL (frequently appeared for other *Frequency Pulsations* tickets) that are captured in topics #7 and #83, respectively.

5 Related Work

Semantic Textual Similarity has diverse applications in information retrieval (Larochelle and Lauly, 2012; Gupta et al., 2018a), search, summarization (Gupta et al., 2011), recommendation systems, etc. For shared STS task in SemEval 2014, numerous researchers applied competitive methods that utilized both heterogeneous features (e.g. word overlap/similarity, negation modeling, sentence/phrase composition) as well as external resources (e.g. Wordnet (Miller, 1995)), along with machine learning approaches such as LSA (Zhao et al., 2014) and word2vec neural language model (Mikolov et al., 2013). In the domain of question retrieval (Cai et al., 2011; Zhang et al., 2014), users retrieve historical questions which precisely match their questions (single sentence) semantically equivalent or relevant.

Neural network based architectures, especially CNN (Yin et al., 2016), LSTM (Mueller and Thyagarajan, 2016), RNN encoder-decoder (Kiros et al., 2015), etc. have shown success in similarity learning

ID	Topic Words (Top 10)
#83	pulsation, frequency, low, load, high, pulsations, increase, narrow, XXXX, mw
#7	trip, turbine, vibration, gas, alarm, gt, time, tripped, pressure, load
#30	start, flame, unit, turbine, combustion, steam, temperature, compressor, XXXX, detector
#16	oil, XXXX, XXXX, pressure, kpa, dp, level, high, mbar, alarm
#19	valve, XXXX, fuel, valves, gas, bypass, check, control, XXXX, XXXX

Table 8: Topics Identifier and words captured by DocNADE

task in Siamese framework (Mueller and Thyagarajan, 2016; Chopra et al., 2005). These models are adapted to similarity learning in sentence pairs using complex learners. Wieting et al. (2016) observed that word vector averaging and LSTM for similarity learning perform better in short and long text pairs, respectively. Our learning objective exploits the multi-channel representations of short and longer texts and compute cross-level similarities in different components of the query and tickets pairs. Instead of learning similarity in a single sentence pair, we propose a novel task and neural architecture for asymmetric textual similarities. To our knowledge, this is the first advancement of Siamese architecture towards multi-and-cross level similarity learning in asymmetric text pairs with an industrial application.

6 Conclusion and Discussion

We have demonstrated deep learning application in STS and IR tasks for an industrial ticketing system. The results indicate that the proposed LSTM is capable of modeling complex semantics by explicit guided representations and does not rely on hand-crafted linguistic features, therefore being generally applicable to any domain. We have showed improved similarity and retrieval via the proposed multi-and-cross-level Replicated Siamese architecture, leading to relevant recommendations especially in industrial use-case. As far we we know, this is the first advancement of Siamese architecture for similarity learning and retrieval in asymmetric text pairs with an industrial application.

We address the challenges in a real-world industrial application of ticketing system. Industrial assets like power plants, production lines, turbines, etc. need to be serviced well because an unplanned outage always leads to significant financial loss. It is an established process in industry to report issues (via query) i.e. symptoms which hint at an operational anomaly to the service provider. This reporting usually leads to textual descriptions of the issue in a ticketing system. The issue is then investigated by service experts who evaluate recommended actions or solutions to the reported issue. The recommended actions or solutions are usually attached to the reported issues and form a valuable knowledge base on how to resolve issues. Since industrial assets tend to be similar over the various installations and since they don't change quickly it is expected that the issues occurring over the various installations may be recurring. Therefore, if for a new issue similar old issues could be easily found this would enable service experts to speed up the evaluation of recommended actions or solutions to the reported issue. The chosen approach is to evaluate the pairwise semantic similarity of the issues describing texts.

We have compared unsupervised and supervised approach for both similarity learning and retrieval tasks, where the supervised approach leads the other. However, we foresee significant gains with the larger amount of similarity data as the amount of labeled similarity data grows and the continuous feedback is incorporated for optimization within the industrial domain, where quality results are desired. In future work, we would also like to investigate attention (Bahdanau et al., 2014) mechanism and dependency (Socher et al., 2012; Gupta et al., 2018b) structures in computing tickets' representation.

Acknowledgements

We thank our colleagues Mark Buckley, Stefan Langer, Subburam Rajaram and Ulli Waltinger, and anonymous reviewers for their review comments. This research was supported by Bundeswirtschaftsministerium (bmwi.de), grant 01MD15010A (Smart Data Web) at Siemens AG- CT Machine Intelligence, Munich Germany.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *Proceedings of the International Conference on Learning Representation*, Alberta, Canada.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

Li Cai, Guangyou Zhou, Kang Liu, and Jun Zhao. 2011. Learning the latent topics for question retrieval in community qa. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 273–281, Chiang Mai, Thailand. Association of Computational Linguistics.

Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 539–546, San Diego, CA, USA. IEEE.

Pankaj Gupta and Bernt Andrassy. 2018. Device and method for natural language processing. US Patent 2018-0,157,643.

Pankaj Gupta, Vijay Shankar Pendluri, and Ishant Vats. 2011. Summarizing text by ranking text units according to shallow linguistic features. Seoul, South Korea. IEEE.

Pankaj Gupta, Thomas Runkler, Heike Adel, Bernt Andrassy, Hans-Georg Zimmermann, and Hinrich Schütze. 2015a. Deep learning methods for the extraction of relations in natural language text. Technical report, Technical University of Munich, Germany.

Pankaj Gupta, Thomas Runkler, and Bernt Andrassy. 2015b. Keyword learning for classifying requirements in tender documents. Technical report, Technical University of Munich, Germany.

Pankaj Gupta, Udhayaraj Sivalingam, Sebastian Pölsterl, and Nassir Navab. 2015c. Identifying patients with diabetes using discriminative restricted boltzmann machines. Technical report, Technical University of Munich, Germany.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2537–2547, Osaka, Japan.

Pankaj Gupta, Florian Buettner, and Hinrich Schütze. 2018a. Document informed neural autoregressive topic models. Researchgate preprint doi: 10.13140/RG.2.2.12322.73925.

Pankaj Gupta, Subburam Rajaram, Bernt Andrassy, Thomas Runkler, and Hinrich Schütze. 2018b. Neural relation extraction within and across sentence boundaries. Researchgate preprint doi: 10.13140/RG.2.2.16517.04327.

Pankaj Gupta, Subburam Rajaram, Hinrich Schütze, and Bernt Andrassy. 2018c. Deep temporal-recurrent-replicated-softmax for topical trends over time. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 1079–1089, New Orleans, USA. Association of Computational Linguistics.

Geoffrey E Hinton and Ruslan R Salakhutdinov. 2009. Replicated softmax: an undirected topic model. In *Advances in neural information processing systems*, pages 1607–1614, Vancouver, Canada.

Geoffrey E Hinton. 2002. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, Montreal, Canada.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270. Association for Computational Linguistics.

Hugo Larochelle and Stanislas Lauly. 2012. A neural autoregressive topic model. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, pages 1607–1614, Lake Tahoe, USA. Curran Associates, Inc.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*, page 3.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, Lake Tahoe, USA.

G.A. Miller. 1995. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):3941.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *the thirtieth AAAI conference on Artificial Intelligence*, volume 16, pages 2786–2792, Phoenix, Arizona USA.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the International Conference on Machine Learning*, pages 1310–1318, Atlanta, USA.

Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. Restricted boltzmann machines for collaborative filtering. In *Proceedings of the 24th International Conference on Machine learning*, pages 791–798, Oregon, USA. Association for Computing Machinery.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. 2012. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of the 2012 joint conference on empirical methods in natural language processing and computational natural language learning*, pages 1201–1211, Jeju Island, Korea. Association for Computational Linguistics.

Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schütze. 2016a. Combining recurrent and convolutional neural networks for relation classification. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 534–539, San Diego, California USA. Association for Computational Linguistics.

Ngoc Thang Vu, Pankaj Gupta, Heike Adel, and Hinrich Schütze. 2016b. Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *Proceedings of the Acoustics, Speech and Signal Processing (ICASSP)*, pages 6060–6064, Shanghai, China. IEEE.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Towards universal paraphrastic sentence embeddings. In *Proceedings of the International Conference on Learning Representations*, San Juan, Puerto Rico.

Wenpeng Yin, Hinrich Schuetze, Bing Xiang, and Bowen Zhou. 2016. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, 4:259–272.

Matthew D Zeiler. 2012. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*.

Kai Zhang, Wei Wu, Haocheng Wu, Zhoujun Li, and Ming Zhou. 2014. Question retrieval with high quality answers in community question answering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 371–380, Shanghai, China. Association for Computing Machinery.

Jiang Zhao, Tiantian Zhu, and Man Lan. 2014. Ecnu: One stone two birds: Ensemble of heterogenous measures for semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 271–277, Dublin, Ireland.

Le Zhao. 2012. Modeling and solving term mismatch for full-text retireval. *ACM SIGIR*, pages 117–118.

Yin Zheng, Yu-Jin Zhang, and Hugo Larochelle. 2016. A deep and autoregressive approach for topic modeling of multimodal data. *IEEE transactions on pattern analysis and machine intelligence*, 38(6):1056–1069.

Chapter 10

LISA: Explaining Recurrent Neural Network Judgments via Layer-wIse Semantic Accumulation and Example to Pattern Transformation

LISA: Explaining Recurrent Neural Network Judgments via Layer-wIse Semantic Accumulation and Example to Pattern Transformation

Pankaj Gupta[1,2], Hinrich Schütze[2]
[1]Corporate Technology, Machine-Intelligence (MIC-DE), Siemens AG Munich, Germany
[2]CIS, University of Munich (LMU) Munich, Germany
pankaj.gupta@siemens.com
pankaj.gupta@campus.lmu.de | inquiries@cislmu.org

Abstract

Recurrent neural networks (RNNs) are temporal networks and cumulative in nature that have shown promising results in various natural language processing tasks. Despite their success, it still remains a challenge to understand their hidden behavior. In this work, we analyze and interpret the cumulative nature of RNN via a proposed technique named as *Layer-wIse-Semantic-Accumulation* (LISA) for explaining decisions and detecting the most likely (i.e., saliency) patterns that the network relies on while decision making. We demonstrate (1) *LISA*: "How an RNN accumulates or builds semantics during its sequential processing for a given text example and expected response" (2) *Example2pattern*: "How the saliency patterns look like for each category in the data according to the network in decision making". We analyse the sensitiveness of RNNs about different inputs to check the increase or decrease in prediction scores and further extract the saliency patterns learned by the network. We employ two relation classification datasets: SemEval 10 Task 8 and TAC KBP Slot Filling to explain RNN predictions via the *LISA* and *example2pattern*.

1 Introduction

The interpretability of systems based on deep neural network is required to be able to explain the reasoning behind the network prediction(s), that offers to (1) verify that the network works as expected and identify the cause of incorrect decision(s) (2) understand the network in order to improve data or model with or without human intervention. There is a long line of research in techniques of interpretability of Deep Neural networks (DNNs) via different aspects, such as explaining network decisions, data generation, etc. Erhan et al. (2009); Hinton (2012); Simonyan et al. (2013) and Nguyen et al. (2016) focused on model aspects to interpret neural networks via activation maximization approach by finding inputs that maximize activations of given neurons. Goodfellow et al. (2014) interprets by generating adversarial examples. However, Baehrens et al. (2010) and Bach et al. (2015); Montavon et al. (2017) explain neural network predictions by sensitivity analysis to different input features and decomposition of decision functions, respectively.

Recurrent neural networks (RNNs) (Elman, 1990) are temporal networks and cumulative in nature to effectively model sequential data such as text or speech. RNNs and their variants such as LSTM (Hochreiter and Schmidhuber, 1997) have shown success in several natural language processing (NLP) tasks, such as entity extraction (Lample et al., 2016; Ma and Hovy, 2016), relation extraction (Vu et al., 2016a; Miwa and Bansal, 2016; Gupta et al., 2016, 2018c), language modeling (Mikolov et al., 2010; Peters et al., 2018), slot filling (Mesnil et al., 2015; Vu et al., 2016b), machine translation (Bahdanau et al., 2014), sentiment analysis (Wang et al., 2016; Tang et al., 2015), semantic textual similarity (Mueller and Thyagarajan, 2016; Gupta et al., 2018a) and dynamic topic modeling (Gupta et al., 2018d).

Past works (Zeiler and Fergus, 2014; Dosovitskiy and Brox, 2016) have mostly analyzed deep neural network, especially CNN in the field of computer vision to study and visualize the features learned by neurons. Recent studies have investigated visualization of RNN and its variants. Tang et al. (2017) visualized the memory vectors to understand the behavior of LSTM and gated recurrent unit (GRU) in speech recognition task. For given words in a sentence, Li et al. (2016) employed heat maps to study sensitivity and meaning composition in recurrent networks. Ming et al. (2017) proposed a tool, RNNVis to visualize hidden states based on RNN's expected response to

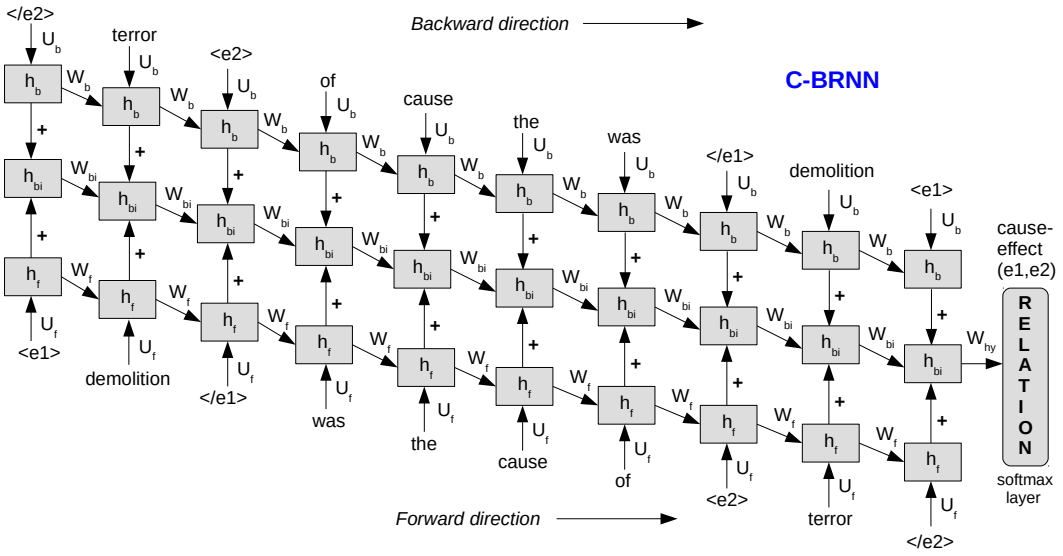


Figure 1: Connectionist Bi-directional Recurrent Neural Network (C-BRNN) (Vu et al., 2016a)

inputs. Peters et al. (2018) studied the internal states of deep bidirectional language model to learn contextualized word representations and observed that the higher-level hidden states capture word semantics, while lower-level states capture syntactical aspects. Despite the possibility of visualizing hidden state activations and performance-based analysis, there still remains a challenge for humans to interpret hidden behavior of the"black box" networks that raised questions in the NLP community as to verify that the network behaves as expected. In this aspect, we address the cumulative nature of RNN with the text input and computed response to answer "how does it aggregate and build the semantic meaning of a sentence word by word at each time point in the sequence for each category in the data".

Contribution: In this work, we analyze and interpret the cumulative nature of RNN via a proposed technique named as *Layer-wIse-Semantic-Accumulation* (LISA) for explaining decisions and detecting the most likely (i.e., saliency) patterns that the network relies on while decision making. We demonstrate (1) *LISA*: "How an RNN accumulates or builds semantics during its sequential processing for a given text example and expected response" (2) *Example2pattern*: "How the saliency patterns look like for each category in the data according to the network in decision making". We analyse the sensitiveness of RNNs about different inputs to check the increase or decrease in prediction scores. For an example sentence that is classified correctly, we identify and extract a saliency pattern (N-grams of words in order learned by the network) that contributes the most in prediction score. Therefore, the term *example2pattern* transformation for each category in the data. We employ two relation classification datasets: SemEval 10 Task 8 and TAC KBP Slot Filling (SF) Shared Task (ST) to explain RNN predictions via the proposed *LISA* and *example2pattern* techniques.

2 Connectionist Bi-directional RNN

We adopt the bi-directional recurrent neural network architecture with ranking loss, proposed by Vu et al. (2016a). The network consists of three parts: a forward pass which processes the original sentence word by word (Equation 1); a backward pass which processes the reversed sentence word by word (Equation 2); and a combination of both (Equation 3). The forward and backward passes are combined by adding their hidden layers. There is also a connection to the previous combined hidden layer with weight W_{bi} with a motivation to include all intermediate hidden layers into the final decision of the network (see Equation 3). They named the neural architecture as 'Connectionist Bi-directional RNN' (C-BRNN). Figure 1 shows the C-BRNN architecture, where all the three parts are trained jointly.

$$h_{f_t} = f(U_f \cdot w_t + W_f \cdot h_{f_{t-1}}) \quad (1)$$

$$h_{b_t} = f(U_b \cdot w_{n-t+1} + W_b \cdot h_{b_{t+1}}) \quad (2)$$

$$h_{bi_t} = f(h_{f_t} + h_{b_t} + W_{bi} \cdot h_{bi_{t-1}}) \quad (3)$$

where w_t is the word vector of dimension d for a word at time step t in a sentence of length n.

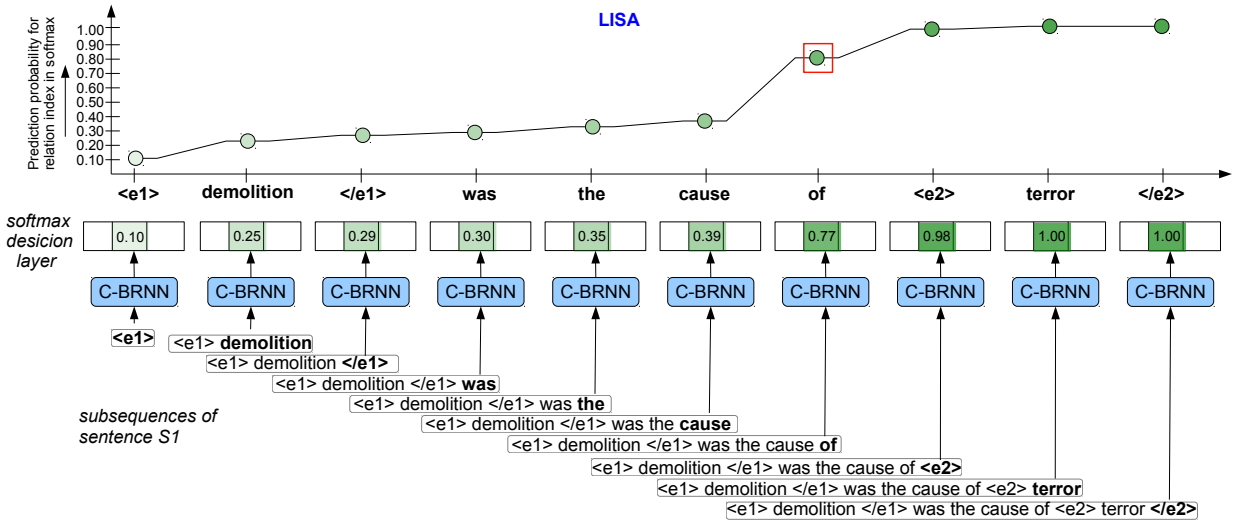


Figure 2: An illustration of Layer-wIse Semantic Accumulation (LISA) in C-BRNN, where we compute prediction score for a (known) relation type at each of the input subsequence. The highlighted indices in the softmax layer signify one of the relation types, i.e., *cause*-effect(e1, e2) in SemEval10 Task 8 dataset. The bold signifies the last word in the subsequence. Note: Each word is represented by N-gram (N=3, 5 or 7), therefore each input subsequence is a sequence of N-grams. E.g., the word 'of' $\rightarrow$ 'cause of <e2>' for N=3. To avoid complexity in this illustration, each word is shown as a uni-gram.

D is the hidden unit dimension. $U_f \in \mathbb{R}^{d \times D}$ and $U_b \in \mathbb{R}^{d \times D}$ are the weight matrices between hidden units and input w_t in forward and backward networks, respectively; $W_f \in \mathbb{R}^{D \times D}$ and $W_b \in \mathbb{R}^{D \times D}$ are the weights matrices connecting hidden units in forward and backward networks, respectively. $W_{bi} \in \mathbb{R}^{D \times D}$ is the weight matrix connecting the hidden vectors of the combined forward and backward network. Following Gupta et al. (2015) during model training, we use 3-gram and 5-gram representation of each word w_t at timestep t in the word sequence, where a 3-gram for w_t is obtained by concatenating the corresponding word embeddings, i.e., $w_{t-1}w_tw_{t+1}$.

Ranking Objective: Similar to Santos et al. (2015) and Vu et al. (2016a), we applied the ranking loss function to train C-BRNN. The ranking scheme offers to maximize the distance between the true label y^+ and the best competitive label c^- given a data point x. It is defined as-

$$\begin{aligned}\mathcal{L} &= \log(1 + \exp(\gamma(m^+ - s_\theta(x)_{y^+}))) \\ &+ \log(1 + \exp(\gamma(m^- + s_\theta(x)_{c^-})))\end{aligned} \quad (4)$$

where $s_\theta(x)_{y^+}$ and $s_\theta(x)_{c^-}$ being the scores for the classes y^+ and c^-, respectively. The parameter γ controls the penalization of the prediction errors and m^+ and m are margins for the correct and incorrect classes. Following Vu et al. (2016a), we set $\gamma = 2$, $m^+ = 2.5$ and $m^- = 0.5$.

Model Training and Features: We represent each word by the concatenation of its word embedding and position feature vectors. We use word2vec (Mikolov et al., 2013) embeddings, that are updated during model training. As position features in relation classification experiments, we use position indicators (PI) (Zhang and Wang, 2015) in C-BRNN to annotate target entity/nominals in the word sequence, without necessity to change the input vectors, while it increases the length of the input word sequences, as four independent words, as position indicators (<e1>, </ e1>, <e2>, </e2>) around the relation arguments are introduced.

In our analysis and interpretation of recurrent neural networks, we use the trained C-BRNN (Figure 1) (Vu et al., 2016a) model.

3 LISA and Example2Pattern in RNN

There are several aspects in interpreting the neural network, for instance via (1) *Data*: "Which dimensions of the data are the most relevant for the task" (2) *Prediction* or *Decision*: "Explain why a certain pattern" is classified in a certain way (3) *Model*: "How patterns belonging to each category in the data look like according to the network".

In this work, we focus to explain RNN via *decision* and *model* aspects by finding the patterns that explains "why" a model arrives at a particu-

lar decision for each category in the data and verifies that model behaves as expected. To do so, we propose a technique named as LISA that interprets RNN about "how it accumulates and builds meaningful semantics of a sentence word by word" and "how the saliency patterns look like according to the network" for each category in the data while decision making. We extract the saliency patterns via *example2pattern* transformation.

LISA Formulation: To explain the cumulative nature of recurrent neural networks, we show how does it build semantic meaning of a sentence word by word belonging to a particular category in the data and compute prediction scores for the expected category on different inputs, as shown in Figure 2. The scheme also depicts the contribution of each word in the sequence towards the final classification score (prediction probability).

At first, we compute different subsequences of word(s) for a given sequence of words (i.e., sentence). Consider a sequence $\mathbf{S}$ of words $[w_1, w_2, ..., w_k, ..., w_n]$ for a given sentence S of length n. We compute n number of subsequences, where each subsequence $\mathbf{S}_{\leq k}$ is a subvector of words $[w_1, ...w_k]$, i.e., $\mathbf{S}_{\leq k}$ consists of words preceding and including the word w_k in the sequence $\mathbf{S}$. In context of this work, extending a subsequence by a word means appending the subsequence by the next word in the sequence. Observe that the number of subsequences, n is equal to the total number of time steps in the C-BRNN.

Next is to compute RNN prediction score for the category R associated with sentence S. We compute the score via the autoregressive conditional $P(R|\mathbf{S}_{\leq k}, \mathbb{M})$ for each subsequence $\mathbf{S}_{\leq k}$, as-

$$P(R|\mathbf{S}_{\leq k}, \mathbb{M}) = softmax(W_{hy} \cdot h_{bi_k} + b_y) \quad (5)$$

using the trained C-BRNN (Figure 1) model $\mathbb{M}$. For each $k \in [1, n]$, we compute the network prediction, $P(R|\mathbf{S}_{\leq k}, \mathbb{M})$ to demonstrate the cumulative property of recurrent neural network that builds meaningful semantics of the sequence $\mathbf{S}$ by extending each subsequence $\mathbf{S}_{\leq k}$ word by word. The internal state h_{bi_k} (attached to softmax layer as in Figure 1) is involved in decision making for each input subsequence $\mathbf{S}_{\leq k}$ with bias vector $b_y \in \mathbb{R}^C$ and hidden-to-softmax weights matrix $W_{hy} \in \mathbb{R}^{D \times C}$ for C categories.

The *LISA* is illustrated in Figure 2, where each word in the sequence contributes to final classification score. It allows us to understand the network decisions via peaks in the prediction score over different subsequences. The peaks signify the saliency patterns (i.e., sequence of words) that the network has learned in order to make decision. For instance, the input word '*of*' following the subsequence '*<e1> demolition </e1> was the cause*' introduces a sudden increase in prediction score for the relation type *cause-effect*(e1, e2). It suggests that the C-BRNN collects the semantics layer-wise via temporally organized subsequences. Observe that the subsequence '*...cause of*' is salient enough in decision making (i.e., prediction score=0.77), where the next subsequence '*...cause of <e2>*' adds in the score to get 0.98.

Algorithm 1 Example2pattern Transformation

Input: sentence S, length n, category R, threshold τ, C-BRNN $\mathbb{M}$, N-gram size N
Output: N-gram saliency pattern $patt$

for k in 1 to n **do**
  compute N-gram$_k$ (eqn 8) of words in S
for k in 1 to n **do**
  compute $\mathbf{S}_{\leq k}$ (eqn 7) of N-grams
  compute $P(R|\mathbf{S}_{\leq k}, \mathbb{M})$ using eqn 5
  if $P(R|\mathbf{S}_{\leq k}, \mathbb{M}) \geq \tau$ **then**
    return $patt \leftarrow \mathbf{S}_{\leq k}[-1]$

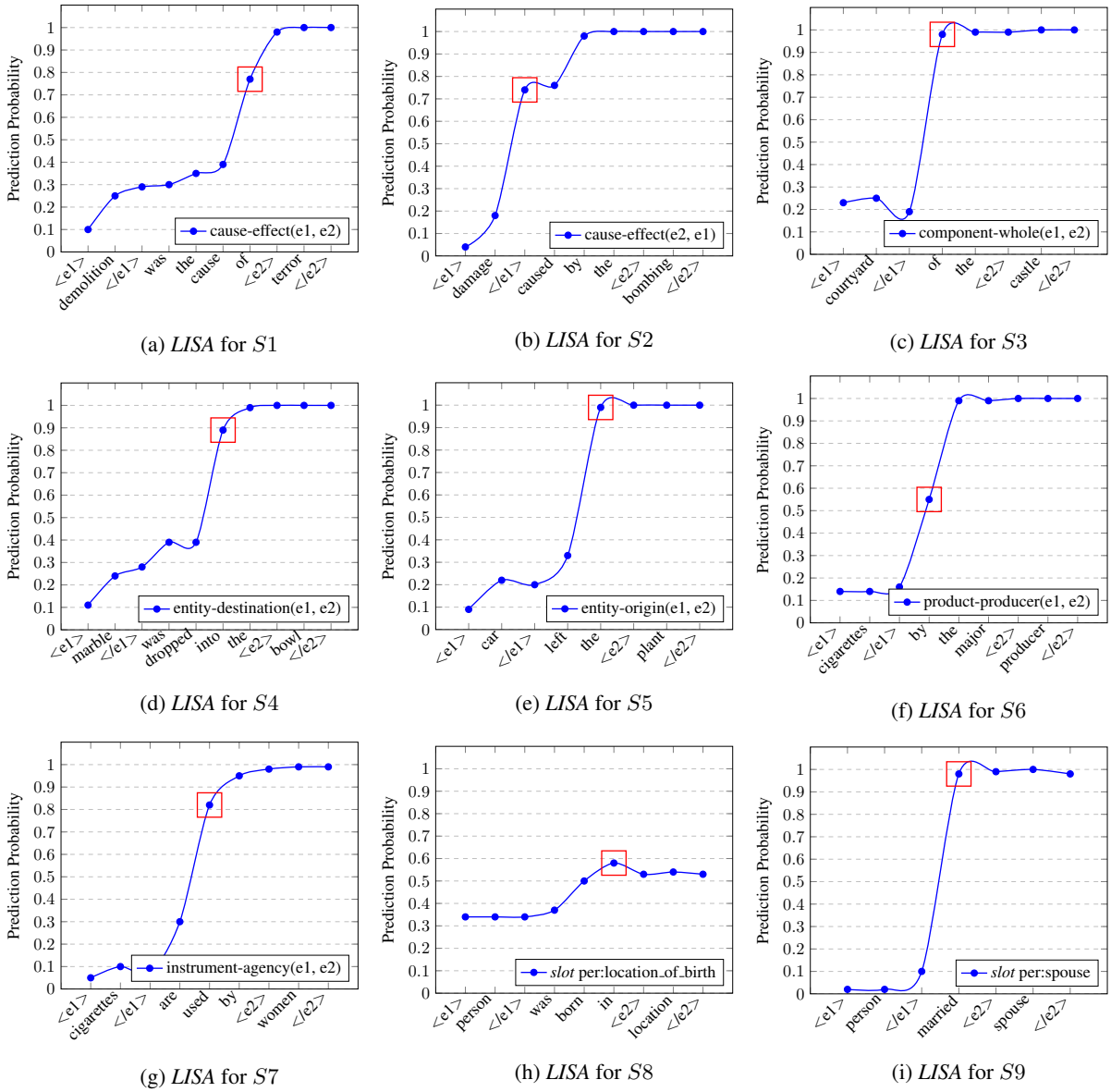
Example2pattern for Saliency Pattern: To further interpret RNN, we seek to identify and extract the most likely input pattern (or phrases) for a given class that is discriminating enough in decision making. Therefore, each example input is transformed into a saliency pattern that informs us about the network learning. To do so, we first compute N-gram for each word w_t in the sentence S. For instance, a 3-gram representation of w_t is given by w_{t-1}, w_t, w_{t+1}. Therefore, an N-gram (for N=3) sequence $\mathbf{S}$ of words is represented as $[[w_{t-1}, w_t, w_{t+1}]_{t=1}^{n}]$, where w_0 and w_{n+1} are PADDING (zero) vectors of embedding dimension.

Following Vu et al. (2016a), we use N-grams (e.g., tri-grams) representation for each word in each subsequence $\mathbf{S}_{\leq k}$ that is input to C-BRNN to compute $P(R|\mathbf{S}_{\leq k})$, where the N-gram (N=3) subsequence $\mathbf{S}_{\leq k}$ is given by,

$$\mathbf{S}_{\leq k} = [[PADDING, w_1, w_2]_1, [w_1, w_2, w_3]_2, ..., [w_{t-1}, w_t, w_{t+1}]_t, ..., [w_{k-1}, w_k, w_{k+1}]_k] \quad (6)$$

$$\mathbf{S}_{\leq k} = [tri_1, tri_2, ..., tri_t, ...tri_k] \quad (7)$$

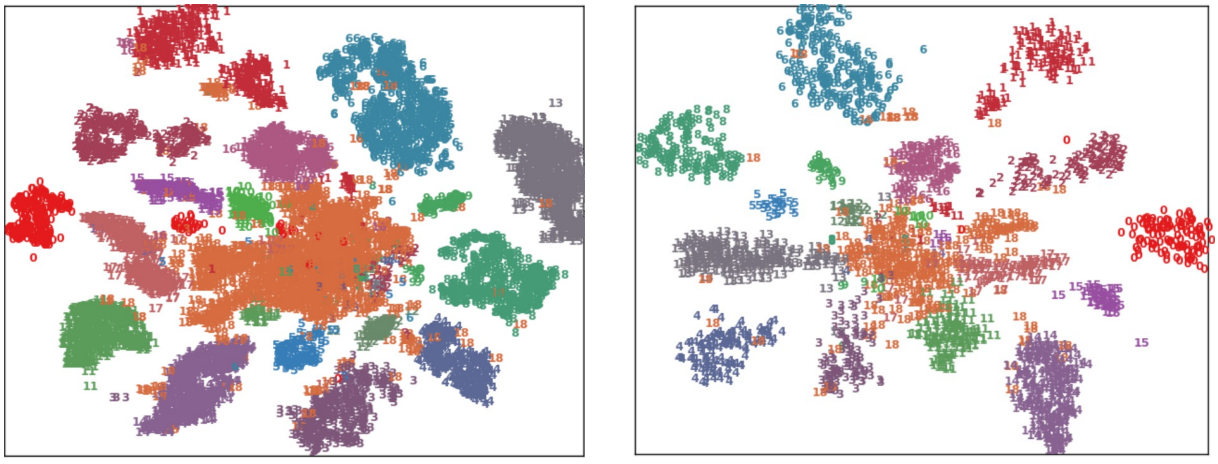
for $k \in [1, n]$. Observe that the 3-gram tri_k con-



(a) *LISA* for $S1$ (b) *LISA* for $S2$ (c) *LISA* for $S3$

(d) *LISA* for $S4$ (e) *LISA* for $S5$ (f) *LISA* for $S6$

(g) *LISA* for $S7$ (h) *LISA* for $S8$ (i) *LISA* for $S9$



(j) t-SNE Visualization for training set (k) t-SNE Visualization for testing set

Figure 3: (a-i) Layer-wIse Semantic Accumulation (LISA) by C-BRNN for different relation types in SemEval10 Task 8 and TAC KBP Slot Filling datasets. The square in red color signifies that the relation is correctly detected with the input subsequence (enough in decision making). (j-k) t-SNE visualization of the last combined hidden unit (h_{bi}) of C-BRNN computed using the SemEval10 train and test sets.

ID	Relation/Slot Types	Example Sentences	Example2Pattern
$S1$	cause-effect(e1, e2)	<e1> demolition </e1> was the cause of <e2> terror </e2>	cause of <e2>
$S2$	cause-effect(e2, e1)	<e1> damage </e1> caused by the <e2> bombing </e2>	damage </e1> caused
$S3$	component-whole(e1, e2)	<e1> countyard </e1> of the <e2> castle </e2>	</e1> of the
$S4$	entity-destination(e1,e2)	<e1> marble </e1> was dropped into the <e2> bowl </e2>	dropped into the
$S5$	entity-origin(e1, e2)	<e1> car </e1> left the <e2> plant </e2>	left the <e2>
$S6$	product-produce(e1, e2)	<e1> cigarettes </e1> by the major <e2> producer </e2>	</e1> by the
$S7$	instrument-agency(e1, e2)	<e1> cigarettes </e1> are used by <e2> women </e2>	</e1> are used
$S8$	per:loc_of_birth(e1, e2)	<e1> person </e1> was born in <e2> location </e2>	born in <e2>
$S9$	per:spouse(e1, e2)	<e1> person </e1> married <e2> spouse </e2>	</e1> married <e2>

Table 1: Example Sentences for *LISA* and *example2pattern* illustrations. The sentences $S1$-$S7$ belong to SemEval10 Task 8 dataset and $S8$-$S9$ to TAC KBP Slot Filling (SF) shared task dataset.

sists of the word w_{k+1}, if k $\neq$ n. To generalize for $i \in [1, \lfloor N/2 \rfloor]$, an N-gram$_k$ of size N for word w_k in C-BRNN is given by-

$$\text{N-gram}_k = [w_{k-i}, ..., w_k, ..., w_{k+i}]_k \quad (8)$$

Algorithm 1 shows the transformation of an example sentence into pattern that is salient in decision making. For a given example sentence S with its length n and category R, we extract the most salient N-gram (N=3, 5 or 7) pattern $patt$ (the last N-gram in the N-gram subsequence $\mathbf{S}_{\leq k}$) that contributes the most in detecting the relation type R. The threshold parameter τ signifies the probability of prediction for the category R by the model $\mathbb{M}$. For an input N-gram sequence $\mathbf{S}_{\leq k}$ of sentence S, we extract the last N-gram, e.g., tri_k that detects the relation R with prediction score above τ. By manual inspection of patterns extracted at different values (0.4, 0.5, 0.6, 0.7) of τ, we found that $\tau = 0.5$ generates the most salient and interpretable patterns. The saliency pattern detection follows LISA as demonstrated in Figure 2, except that we use N-gram (N =3, 5 or 7) input to detect and extract the key relationship patterns.

4 Analysis: Relation Classification

Given a sentence and two annotated nominals, the task of binary relation classification is to predict the semantic relations between the pairs of nominals. In most cases, the context in between the two nominals define the relationship. However, Vu et al. (2016a) has shown that the extended context helps. In this work, we focus on the building semantics for a given sentence using relationship contexts between the two nominals.

We analyse RNNs for *LISA* and *example2pattern* using two relation classification datsets: (1) SemEval10 Shared Task 8 (Hendrickx et al., 2009) (2) TAC KBP Slot Filling (SF) shared task[1] (Adel and Schütze, 2015). We demonstrate the sensitiveness of RNN for different subsequences (Figure 2), input in the same order as in the original sentence. We explain its predictions (or judgments) and extract the salient relationship patterns learned for each category in the two datasets.

Input word sequence to C-BRNN	pp
<**e1**>	0.10
<e1> **demolition**	0.25
<e1> demolition <**/e1**>	0.29
<e1> demolition </e1> **was**	0.30
<e1> demolition </e1> was **the**	0.35
<e1> demolition </e1> was the **cause**	0.39
<e1> demolition </e1> was the cause **of**	<u>0.77</u>
<e1> demolition </e1> was the cause of <**e2**>	0.98
<e1> demolition </e1> was the cause of <e2> **terror**	1.00
<e1> demolition </e1> was the cause of <e2> terror <**/e2**>	1.00

Table 2: Semantic accumulation and sensitivity of C-BRNN over subsequences for sentence $S1$. Bold indicates the last word in the subsequence. pp: prediction probability in the softmax layer for the relation type. The underline signifies that the pp is sufficient enough (τ=0.50) in detecting the relation. Saliency patterns, i.e., N-grams can be extracted from the input subsequence that leads to a sudden peak in pp, where $pp \geq \tau$.

4.1 SemEval10 Shared Task 8 dataset

The relation classification dataset of the Semantic Evaluation 2010 (SemEval10) shared task 8 (Hendrickx et al., 2009) consists of 19 relations (9 directed relations and one artificial class `Other`), 8,000 training and 2,717 testing sentences. We split the training data into train (6.5k) and development (1.5k) sentences to optimize the C-BRNN

[1]data from the slot filler classification component of the slot filling pipeline, treated as relation classification

Relation	3-gram Patterns	5-gram Patterns	7-gram Patterns
cause-effect(e1,e2)	</e1> cause <e2> </e1> caused a that cause respiratory which cause acne leading causes of	the leading causes of <e2> the main causes of <e2> </e1> leads to <e2> inspiration </e1> that results in <e2> </e1> resulted in the <e2>	is one of the leading causes of is one of the main causes of </e1> that results in <e2> hardening </e2> </e1> resulted in the <e2> loss </e2> <e1> sadness </e1> leads to <e2> inspiration
cause-effect(e2,e1)	caused due to comes from the arose from an caused by the radiated from a	</e1> has been caused by </e1> are caused by the </e1> arose from an <e2> </e1> caused due to <e2> infection </e2> results in an	</e1> is caused by a <e2> comet </e1> however has been caused by the </e1> that has been caused by the that has been caused by the <e2> <e1> product </e1> arose from an <e2>
content-container(e1,e2)	in a <e2> was inside a contained in a hidden in a stored in a	</e1> was contained in a </e1> was discovered inside a </e1> were in a <e2> is hidden in a <e2> </e1> was contained in a	</e1> was contained in a <e2> box </e1> was in a <e2> suitcase </e2> </e1> were in a <e2> box </e2> </e1> was inside a <e2> box </e2> </e1> was hidden in an <e2> envelope
product-produce(e1,e2)	</e1> released by </e1> issued by </e1> created by by the <e2> of the <e1>	</e1> issued by the <e2> </e1> was prepared by <e2> was written by a <e2> </e1> built by the <e2> </e1> are made by <e2>	<e1> products </e1> created by an <e2> </e1> by an <e2> artist </e2> who </e1> written by most of the <e2> temple </e1> has been built by <e2> </e1> were founded by the <e2> potter
whole(e1, e2) component-	</e1> of the of the <e2> part of the </e1> of <e2> </e1> on a	</e1> of the <e2> device </e1> was a part of </e1> is part of the is a basic element of </e1> is part of a	the <e1> timer </e1> of the <e2> </e1> was a part of the romulan </e1> was the best part of the </e1> is a basic element of the are core components of the <e2> solutions
entity-destination(e1,e2)	put into a released into the </e1> into the moved into the added to the	have been moving into the was dropped into the <e2> </e1> moved into the <e2> were released into the <e2> </e1> have been exported to	</e1> have been moving back into <e2> </e1> have been moving into the <e2> </e1> have been dropped into the <e2> </e1> have been released back into the power </e1> is exported to the <e2>
instrument-agency(e1,e2)	</e1> are used used by <e2> </e1> is used set by the </e1> set by	</e1> assists the <e2> eye </e1> are used by <e2> </e1> were used by some </e1> with which the <e2> readily associated with the <e2>	cigarettes </e1> are used by <e2> women <e1> telescope </e1> assists the <e2> eye <e1> practices </e1> for <e2> engineers </e2> the best <e1> tools </e1> for <e2> <e1> wire </e1> with which the <e2>

Table 3: SemEval10 Task 8 dataset: N-Gram (3, 5 and 7) saliency patterns extracted for different relation types by C-BRNN with PI

network. For instance, an example sentence with relation label is given by-

`The <e1> demolition </e1> was the cause of <e2> terror </e2> and communal divide is just a way of not letting truth prevail.` $\rightarrow$ `cause-effect(e1,e2)`

The terms `demolition` and `terror` are the relation arguments or nominals, where the phrase `was the cause of` is the relationship context between the two arguments. Table 1 shows the examples sentences (shortened to argument1+relationship context+argument2) drawn from the development and test sets that we employed to analyse the C-BRNN for semantic accumulation in our experiments. We use the similar experimental setup as Vu et al. (2016a).

LISA Analysis: As discussed in Section 3, we interpret C-BRNN by explaining its predictions via the semantic accumulation over the subsequences $\mathbf{S}_{\leq k}$ (Figure 2) for each sentence S. We select the example sentences $S1$-$S7$ (Table 1) for which the network predicts the correct relation type with high scores. For an example sentence $S1$, Table 2 illustrates how different subsequences are input to C-BRNN in order to compute prediction scores pp in the softmax layer for the relation `cause-effect(e1, e2)`. We use tri-gram (section 3) word representation for each word for the examples $S1$-$S7$.

Figures 3a, 3b, 3c, 3d 3e, 3f and 3g demonstrate the cumulative nature and sensitiveness of RNN via prediction probability (pp) about different inputs for sentences $S1$-$S7$, respectively. For

Slots	N-gram Patterns
per-spouse(e1,e2)	</e1> wife of </e1> , wife </e1> wife </e1> married <e2> </e1> marriages to
per-location_of_birth(e1,e2)	was born in born in <e2> a native of </e1> from <e2> </e1> 's hometown

Table 4: TAC KBP SF dataset: Tri-gram saliency patterns extracted for slots *per*:*spouse*(e1, e2) and *per*:*location_of_birth*(e1,e2)

instance in Figure 3a and Table 2, the C-BRNN builds meaning of the sentence $S1$ word by word, where a sudden increase in pp is observed when the input subsequence `<e1> demolition </e1> was the cause` is extended with the next term `of` in the word sequence **S**. Note that the relationship context between the arguments `demolition` and `terror` is sufficient enough in detecting the relationship type. Interestingly, we also observe that the prepositions (such as `of`, `by`, `into`, etc.) in combination with verbs are key features in building the meaningful semantics.

Saliency Patterns via example2pattern Transformation: Following the discussion in Section 3 and Algorithm 1, we transform each correctly identified example into pattern by extracting the most likely N-gram in the input subsequence(s). In each of the Figures 3a, 3b, 3c, 3d 3e, 3f and 3g, the square box in red color signifies that the relation type is correctly identified (when $\tau = 0.5$) at this particular subsequence input (without the remaining context in the sentence). We extract the last N-gram of such a subsequence.

Table 1 shows the *example2pattern* transformations for sentences $S1$-$S7$ in SemEval10 dataset, derived from Figures 3a-3g, respectively with N=3 (in the N-grams). Similarly, we extract the salient patterns (3-gram, 5-gram and 7-gram) (Table 3) for different relationships. We also observe that the relation types `content-container(e1, e2)` and `instrument-agency(e1, e2)` are mostly defined by smaller relationship contexts (e.g, 3-gram), however `entity-destination(e1,e2)` by larger contexts (7-gram).

4.2 TAC KBP Slot Filling dataset

We investigate another dataset from TAC KBP Slot Filling (SF) shared task (Surdeanu, 2013), where we use the relation classification dataset by Adel et al. (2016) in the context of slot filling. We have selected the two slots: *per:loc_of_birth* and *per:spouse* out of 24 types.

LISA Analysis: Following Section 4.1, we analyse the C-BRNN for LISA using sentences $S8$ and $S9$ (Table 1). Figures 3h and 3i demonstrate the cumulative nature of recurrent neural network, where we observe that the salient patterns `born in <e2>` and `</e1> married` $e2$ lead to correct decision making for $S8$ and $S9$, respectively. Interestingly for $S8$, we see a decrease in prediction score from 0.59 to 0.52 on including terms in the subsequence, following the term `in`.

Saliency Patterns via example2pattern Transformation: Following Section 3 and Algorithm 1, we demonstrate the *example2pattern* transformation of sentences $S8$ and $S9$ in Table 1 with tir-grams. In addition, Table 4 shows the tri-gram salient patterns extracted for the two slots.

5 Visualizing Latent Semantics

In this section, we attempt to visualize the hidden state of each test (and train) example that has accumulated (or built) the meaningful semantics during sequential processing in C-BRNN. To do this, we compute the last hidden vector h_{bi} of the combined network (e.g., h_{bi} attached to the softmax layer in Figure 1) for each test (and train) example and visualize (Figure 3k and 3j) using t-SNE (Maaten and Hinton, 2008). Each color represents a relation-type. Observe the distinctive clusters of accumulated semantics in hidden states for each category in the data (SemEval10 Task 8).

6 Conclusion and Future Work

We have demonstrated the cumulative nature of recurrent neural networks via sensitivity analysis over different inputs, i.e., *LISA* to understand how they build meaningful semantics and explain predictions for each category in the data. We have also detected a salient pattern in each of the example sentences, i.e., *example2pattern transformation* that the network learns in decision making. We extract the salient patterns for different categories in two relation classification datasets.

In future work, it would be interesting to analyse the sensitiveness of RNNs with corruption in

the salient patterns. One could also investigate visualizing the dimensions of hidden states (activation maximization) and word embedding vectors with the network decisions over time. We forsee to apply *LISA* and *example2pattern* on different tasks such as document categorization, sentiment analysis, language modeling, etc. Another interesting direction would be to analyze the bag-of-word neural topic models such as DocNADE (Larochelle and Lauly, 2012) and iDocNADE (Gupta et al., 2018b) to interpret their semantic accumulation during autoregressive computations in building document representation(s). We extract the saliency patterns for each category in the data that can be effectively used in instantiating pattern-based information extraction systems, such as bootstrapping entity (Gupta and Manning, 2014) and relation extractors (Gupta et al., 2018e).

Acknowledgments

We thank Heike Adel for providing us with the TAC KBP dataset used in our experiments. We express appreciation for our colleagues Bernt Andrassy, Florian Buettner, Ulli Waltinger, Mark Buckley, Stefan Langer, Subbu Rajaram, Yatin Chaudhary, and anonymous reviewers for their in-depth review comments. This research was supported by Bundeswirtschaftsministerium (`bmwi.de`), grant 01MD15010A (Smart Data Web) at Siemens AG- CT Machine Intelligence, Munich Germany.

References

Heike Adel, Benjamin Roth, and Hinrich Schütze. 2016. Comparing convolutional neural networks to traditional models for slot filling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 828–838. Association for Computational Linguistics.

Heike Adel and Hinrich Schütze. 2015. Cis at tac cold start 2015: Neural networks and coreference resolution for slot filling. *Proc. TAC2015*.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. 2015. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert MÃžller. 2010. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11(Jun):1803–1831.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Alexey Dosovitskiy and Thomas Brox. 2016. Inverting visual representations with convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4829–4837.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2009. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Pankaj Gupta, Bernt Andrassy, and Hinrich Schütze. 2018a. Replicated siamese lstm in ticketing system for similarity learning and retrieval in asymmetric texts. In *Proceedings of the Workshop on Semantic Deep Learning (SemDeep-3) in the 27th International Conference on Computational Linguistics (COLING2018)*. The COLING 2018 organizing committee.

Pankaj Gupta, Florian Buettner, and Hinrich Schütze. 2018b. Document informed neural autoregressive topic models. Researchgate preprint doi: 10.13140/RG.2.2.12322.73925.

Pankaj Gupta, Subburam Rajaram, Thomas Runkler, Hinrich Schütze, and Bernt Andrassy. 2018c. Neural relation extraction within and across sentence boundaries. Researchgate preprint doi: 10.13140/RG.2.2.16517.04327.

Pankaj Gupta, Subburam Rajaram, Hinrich Schütze, and Bernt Andrassy. 2018d. Deep temporal-recurrent-replicated-softmax for topical trends over time. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 1079–1089, New Orleans, USA. Association of Computational Linguistics.

Pankaj Gupta, Benjamin Roth, and Hinrich Schütze. 2018e. Joint bootstrapping machines for high confidence relation extraction. In *Proceedings of the 16th Annual Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Long Papers)*, pages 26–36, New Orleans, USA. Association of Computational Linguistics.

Pankaj Gupta, Thomas Runkler, Heike Adel, Bernt Andrassy, Hans-Georg Zimmermann, and Hinrich Schütze. 2015. Deep learning methods for the extraction of relations in natural language text. Technical report, Technical University of Munich, Germany.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. 2016. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2537–2547. The COLING 2016 Organizing Committee.

Sonal Gupta and Christopher Manning. 2014. Spied: Stanford pattern based information extraction and diagnostics. In *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*, pages 38–44.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2009. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, pages 94–99. Association for Computational Linguistics.

Geoffrey E Hinton. 2012. A practical guide to training restricted boltzmann machines. In *Neural networks: Tricks of the trade*, pages 599–619. Springer.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270. Association for Computational Linguistics.

Hugo Larochelle and Stanislas Lauly. 2012. A neural autoregressive topic model. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2708–2716. Curran Associates, Inc.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. 2016. Visualizing and understanding neural models in nlp. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691. Association for Computational Linguistics.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074. Association for Computational Linguistics.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. 2015. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):530–539.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *Proceedings of the Workshop at ICLR*.

Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Eleventh Annual Conference of the International Speech Communication Association*.

Yao Ming, Shaozu Cao, Ruixiang Zhang, Zhen Li, Yuanzhe Chen, Yangqiu Song, and Huamin Qu. 2017. Understanding hidden memories of recurrent neural networks. *arXiv preprint arXiv:1710.10777*.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116. Association for Computational Linguistics.

Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. 2017. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *AAAI*, volume 16, pages 2786–2792.

Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. 2016. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In *Advances in Neural Information Processing Systems*, pages 3387–3395.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics.

Cicero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. 2015. Classifying relations by ranking with convolutional neural networks. In *Proceedings of ACL*. Association for Computational Linguistics.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2013. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*.

Mihai Surdeanu. 2013. Overview of the tac2013 knowledge base population evaluation: English slot filling and temporal slot filling. In *TAC*.

Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1432. Association for Computational Linguistics.

Zhiyuan Tang, Ying Shi, Dong Wang, Yang Feng, and Shiyue Zhang. 2017. Memory visualization for gated recurrent neural networks in speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*, pages 2736–2740. IEEE.

Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schütze. 2016a. Combining recurrent and convolutional neural networks for relation classification. In *Proceedings of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 534–539, San Diego, California USA. Association for Computational Linguistics.

Ngoc Thang Vu, Pankaj Gupta, Heike Adel, and Hinrich Schütze. 2016b. Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *Proceedings of the Acoustics, Speech and Signal Processing (ICASSP)*, pages 6060–6064, Shanghai, China. IEEE.

Yequan Wang, Minlie Huang, xiaoyan zhu, and Li Zhao. 2016. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615. Association for Computational Linguistics.

Matthew D Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer.

Dongxu Zhang and Dong Wang. 2015. Relation classification via recurrent neural network.

Bibliography

David H. Ackley, Geoffrey E. Hinton, and Terrence J. Sejnowski. A learning algorithm for boltzmann machines. *Cognitive Science*, 9(1):147–169, 1985. doi: 10.1207/s15516709cog0901_7. URL https://doi.org/10.1207/s15516709cog0901_7.

Heike Adel and Hinrich Schütze. CIS at TAC cold start 2015: Neural networks and coreference resolution for slot filling. In *Proceedings of the 2015 Text Analysis Conference, TAC*. NIST, 2015. URL https://tac.nist.gov/publications/2015/participant.papers/TAC2015.CIS.proceedings.pdf.

Heike Adel and Hinrich Schütze. Global normalization of convolutional neural networks for joint entity and relation classification. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 1723–1729. Association for Computational Linguistics, 2017. URL https://aclanthology.info/papers/D17-1181/d17-1181.

Heike Adel, Benjamin Roth, and Hinrich Schütze. Comparing convolutional neural networks to traditional models for slot filling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 828–838. Association for Computational Linguistics, 2016. doi: 10.18653/v1/N16-1097. URL http://aclweb.org/anthology/N16-1097.

Eugene Agichtein and Luis Gravano. *Snowball*: Extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM Conference on Digital Libraries, June 2-7, 2000, San Antonio, TX, USA*, pages 85–94. Association for Computing Machinery, 2000. doi: 10.1145/336597.336644. URL https://doi.org/10.1145/336597.336644.

Jean Aitchison. Language change. In *The Routledge Companion to Semiotics and Linguistics*, pages 111–120. Routledge, 2005.

Alan Akbik, Duncan Blythe, and Roland Vollgraf. Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1638–1649. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/C18-1139.

Nikolaos Aletras and Mark Stevenson. Evaluating topic coherence using distributional semantics. In Katrin Erk and Alexander Koller, editors, *Proceedings of the 10th International Conference on Computational Semantics, IWCS 2013, March 19-22, 2013, University of Potsdam, Potsdam, Germany*, pages 13–22. The Association for Computer Linguistics, 2013. URL http://aclweb.org/anthology/W/W13/W13-0102.pdf.

James Allan. Introduction to topic detection and tracking. In *Topic detection and tracking*, pages 1–16. Springer, 2002.

James Allan, Jaime G Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. Topic detection and tracking pilot study final report. In Luc De Raedt and Peter A. Flach, editors, *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218, 1998. URL http://maroo.cs.umass.edu/getpdf.php?id=14.

Moustafa Alzantot, Yash Sharma, Ahmed Elgohary, Bo-Jhang Ho, Mani B. Srivastava, and Kai-Wei Chang. Generating natural language adversarial examples. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2890–2896. Association for Computational Linguistics, 2018. URL https://aclanthology.info/papers/D18-1316/d18-1316.

Shun-ichi Amari, Hyeyoung Park, and Kenji Fukumizu. Adaptive method of realizing natural gradient learning for multilayer perceptrons. *Neural Computation*, 12(6):1399–1409, 2000. doi: 10.1162/089976600300015420. URL https://doi.org/10.1162/089976600300015420.

Gabor Angeli, Melvin Jose Johnson Premkumar, and Christopher D. Manning. Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 344–354, Beijing, China, July 2015.

Association for Computational Linguistics. doi: 10.3115/v1/P15-1034. URL https://www.aclweb.org/anthology/P15-1034.

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. Dbpedia: A nucleus for a web of open data. In Karl Aberer, Key-Sun Choi, Natasha Fridman Noy, Dean Allemang, Kyung-Il Lee, Lyndon J. B. Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer, 2007. URL https://doi.org/10.1007/978-3-540-76298-0_52.

Sebastian Bach, Alexander Binder, Grégoire Montavon, Klauschen Frederick, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS ONE*, 10(7), 2015. URL https://doi.org/10.1371/journal.pone.0130140.

David Baehrens, Timon Schroeter, Stefan Harmeling, Motoaki Kawanabe, Katja Hansen, and Klaus-Robert Müller. How to explain individual classification decisions. *Journal of Machine Learning Research*, 11:1803–1831, 2010. URL http://portal.acm.org/citation.cfm?id=1859912.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1409.0473.

Robert Bamler and Stephan Mandt. Dynamic word embeddings. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pages 380–389. Proceedings of Machine Learning Research, 2017. URL http://proceedings.mlr.press/v70/bamler17a.html.

Michele Banko, Michael J. Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. Open information extraction from the web. In Manuela M. Veloso, editor, *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007*, pages 2670–2676, 2007. URL http://ijcai.org/Proceedings/07/Papers/429.pdf.

Pierpaolo Basile, Annalina Caputo, and Giovanni Semeraro. Analysing word meaning over time by exploiting temporal random indexing. In *Proceedings of the 1st Italian Conference on Computational Linguistics (CLiC-it)*. Pisa University Press, 2014. URL https://www.researchgate.net/publication/269392475_Analysing_Word_Meaning_over_Time_by_Exploiting_Temporal_Random_Indexing.

Hannah Bast, Florian Bäurle, Björn Buchhold, and Elmar Haußmann. Easy access to the freebase dataset. In *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014, Companion Volume*, pages 95–98. Association for Computing Machinery, 2014. doi: 10.1145/2567948.2577016. URL https://doi.org/10.1145/2567948.2577016.

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, David Warde-Farley, and Yoshua Bengio. Theano: New features and speed improvements. *CoRR*, abs/1211.5590, 2012. URL http://arxiv.org/abs/1211.5590.

David S. Batista, Bruno Martins, and Mário J. Silva. Semi-supervised bootstrapping of relationship extractors with distributional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 499–504. Association for Computational Linguistics, 2015. URL http://aclweb.org/anthology/D/D15/D15-1056.pdf.

Giannis Bekoulis, Johannes Deleu, Thomas Demeester, and Chris Develder. Joint entity recognition and relation extraction as a multi-head selection problem. *Expert Syst. Appl.*, 114:34–45, 2018. doi: 10.1016/j.eswa.2018.07.032. URL https://doi.org/10.1016/j.eswa.2018.07.032.

Yonatan Belinkov and Yonatan Bisk. Synthetic and natural noise both break neural machine translation. *CoRR*, abs/1711.02173, 2017. URL http://arxiv.org/abs/1711.02173.

Yonatan Belinkov and James Glass. Analysis methods in neural language processing: A survey. *CoRR*, abs/1812.08951, 2018. URL http://arxiv.org/abs/1812.08951.

Yoshua Bengio. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127, 2009. URL https://doi.org/10.1561/2200000006.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *Journal of Machine Learning Research*, 3:1137–1155, 2003. URL http://www.jmlr.org/papers/v3/bengio03a.html.

Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann, editors, *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*, pages 153–160. MIT Press, 2006. URL http://papers.nips.cc/paper/3048-greedy-layer-wise-training-of-deep-networks.

James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: A cpu and gpu math compiler in python. In *Proceedings of the 9th Python in Science Conference*, volume 1, 2010. URL http://www.iro.umontreal.ca/~lisa/pointeurs/theano_scipy2010.pdf.

David M. Blei. Probabilistic topic models. *Commun. ACM*, 55(4):77–84, 2012. doi: 10.1145/2133806.2133826. URL http://doi.acm.org/10.1145/2133806.2133826.

David M. Blei and John D. Lafferty. Dynamic topic models. In William W. Cohen and Andrew Moore, editors, *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML)*, volume 148 of *ACM International Conference Proceeding Series*, pages 113–120. Association for Computing Machinery, 2006. doi: 10.1145/1143844.1143859. URL https://doi.org/10.1145/1143844.1143859.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003. URL http://www.jmlr.org/papers/v3/blei03a.html.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 2017. URL https://transacl.org/ojs/index.php/tacl/article/view/999.

Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In Jason Tsong-Li Wang, editor, *Proceedings of the ACM SIGMOD*

International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008, pages 1247–1250. Association for Computing Machinery, 2008. URL https://doi.org/10.1145/1376616.1376746.

Robert Bossy, Julien Jourde, Philippe Bessières, Maarten van de Guchte, and Claire Nèdellec. Bionlp shared task 2011 - Bacteria biotope. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 56–64. Association for Computational Linguistics, 2011. URL http://aclweb.org/anthology/W11-1809.

Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012. URL http://icml.cc/2012/papers/590.pdf.

Sergey Brin. Extracting patterns and relations from the world wide web. In Paolo Atzeni, Alberto O. Mendelzon, and Giansalvatore Mecca, editors, *The World Wide Web and Databases, International Workshop WebDB'98, Valencia, Spain, March 27-28, 1998, Selected Papers*, volume 1590 of *Lecture Notes in Computer Science*, pages 172–183. Springer, 1998. doi: 10.1007/10704656_11. URL https://doi.org/10.1007/10704656_11.

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a siamese time delay neural network. In Jack D. Cowan, Gerald Tesauro, and Joshua Alspector, editors, *Advances in Neural Information Processing Systems 6, [7th NIPS Conference, Denver, Colorado, USA, 1993]*, pages 737–744. Morgan Kaufmann, 1993. URL https://www.researchgate.net/publication/270283023_Signature_Verification_using_a_Siamese_Time_Delay_Neural_Network.

Mirko Bronzi, Zhaochen Guo, Filipe Mesquita, Denilson Barbosa, and Paolo Merialdo. Automatic evaluation of relation extraction systems on large-scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, pages 19–24. Association for Computational Linguistics, 2012. URL http://aclweb.org/anthology/W12-3004.

Razvan Bunescu and Raymond Mooney. A shortest path dependency kernel for relation extraction. In *Proceedings of Human Language Technology Conference*

and Conference on Empirical Methods in Natural Language Processing, page 724–731. Association for Computational Linguistics, 2005. URL http://aclweb.org/anthology/H05-1091.

Kseniya Buraya, Lidia Pivovarova, Sergey Budkov, and Andrey Filchenkov. Towards never ending language learning for morphologically rich languages. In *Proceedings of the 6th Workshop on Balto-Slavic Natural Language Processing*, pages 108–118. Association for Computational Linguistics, 2017. doi: 10.18653/v1/W17-1417. URL http://aclweb.org/anthology/W17-1417.

Hyeran Byun and Seong-Whan Lee. Applications of support vector machines for pattern recognition: A survey. In Seong-Whan Lee and Alessandro Verri, editors, *Pattern Recognition with Support Vector Machines, First International Workshop, SVM 2002, Niagara Falls, Canada, August 10, 2002, Proceedings*, volume 2388 of *Lecture Notes in Computer Science*, pages 213–236. Springer, 2002. doi: 10.1007/3-540-45665-1_17. URL https://doi.org/10.1007/3-540-45665-1_17.

Li Cai, Guangyou Zhou, Kang Liu, and Jun Zhao. Learning the latent topics for question retrieval in community qa. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 273–281. Asian Federation of Natural Language Processing, 2011. URL http://aclweb.org/anthology/I11-1031.

Rui Cai, Xiaodong Zhang, and Houfeng Wang. Bidirectional recurrent convolutional neural network for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016. URL http://aclweb.org/anthology/P/P16/P16-1072.pdf.

Ziqiang Cao, Sujian Li, Yang Liu, Wenjie Li, and Heng Ji. A novel neural topic model and its supervised extension. In Blai Bonet and Sven Koenig, editors, *Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2210–2216. AAAI Press, 2015. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9303.

Dallas Card, Chenhao Tan, and Noah A. Smith. A neural framework for generalized topic models. *CoRR*, abs/1705.09296, 2017. URL http://arxiv.org/abs/1705.09296.

Miguel Á. Carreira-Perpiñán and Geoffrey E. Hinton. On contrastive divergence learning. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS 2005, Bridgetown, Barbados, January 6-8, 2005*. Society for Artificial Intelligence and Statistics, 2005. URL http://www.gatsby.ucl.ac.uk/aistats/fullpapers/217.pdf.

Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997. URL https://doi.org/10.1023/A:1007379606734.

George Casella and Edward I George. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174, 1992. URL http://biostat.jhsph.edu/~mmccall/articles/casella_1992.pdf.

Hau Chan and Leman Akoglu. External evaluation of topic models: A graph mining approach. In Hui Xiong, George Karypis, Bhavani M. Thuraisingham, Diane J. Cook, and Xindong Wu, editors, *2013 IEEE 13th International Conference on Data Mining*, pages 973–978. IEEE Computer Society, 2013. doi: 10.1109/ICDM.2013.112. URL https://doi.org/10.1109/ICDM.2013.112.

Jonathan Chang, Jordan L. Boyd-Graber, Sean Gerrish, Chong Wang, and David M. Blei. Reading tea leaves: How humans interpret topic models. In Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta, editors, *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems*, pages 288–296. Curran Associates, Inc., 2009. URL http://users.umiacs.umd.edu/~jbg/docs/nips2009-rtl.pdf.

Laura Chiticariu, Yunyao Li, and Frederick R. Reiss. Rule-based information extraction is dead! long live rule-based information extraction systems! In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 827–832. Association for Computational Linguistics, 2013. URL http://aclweb.org/anthology/D13-1079.

KyungHyun Cho, Tapani Raiko, and Alexander Ilin. Gaussian-bernoulli deep boltzmann machine. In *The 2013 International Joint Conference on Neural Networks, IJCNN 2013, Dallas, TX, USA, August 4-9, 2013*, pages 1–7. IEEE, 2013a. doi: 10.1109/IJCNN.2013.6706831. URL https://doi.org/10.1109/IJCNN.2013.6706831.

KyungHyun Cho, Tapani Raiko, and Alexander Ilin. Enhanced gradient for training restricted boltzmann machines. *Neural Computation*, 25(3):805–

831, 2013b. doi: 10.1162/NECO_a_00397. URL https://doi.org/10.1162/NECO_a_00397.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. In Dekai Wu, Marine Carpuat, Xavier Carreras, and Eva Maria Vecchi, editors, *Proceedings of SSST@EMNLP 2014, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation, Doha, Qatar, 25 October 2014*, pages 103–111. Association for Computational Linguistics, 2014. URL http://aclweb.org/anthology/W/W14/W14-4012.pdf.

Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 539–546. IEEE Computer Society, 2005. doi: 10.1109/CVPR.2005.202. URL https://doi.org/10.1109/CVPR.2005.202.

Junyoung Chung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014. URL http://arxiv.org/abs/1412.3555.

Junyoung Chung, Çaglar Gülçehre, Kyunghyun Cho, and Yoshua Bengio. Gated feedback recurrent neural networks. In Francis R. Bach and David M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 2067–2075. JMLR.org, 2015. URL http://jmlr.org/proceedings/papers/v37/chung15.html.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 160–167. Association for Computing Machinery, 2008. doi: 10.1145/1390156.1390177. URL https://doi.org/10.1145/1390156.1390177.

Rajarshi Das, Manzil Zaheer, and Chris Dyer. Gaussian lda for topic models with word embeddings. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 795–804.

Association for Computational Linguistics, 2015. doi: 10.3115/v1/P15-1077. URL http://aclweb.org/anthology/P15-1077.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990. URL https://doi.org/10.1002/(SICI)1097-4571(199009)41:6\<391::AID-ASI1\>3.0.CO;2-9.

Louise Deléger, Robert Bossy, Estelle Chaix, Mouhamadou Ba, Arnaud Ferré, Philippe Bessières, and Claire Nedellec. Overview of the bacteria biotope task at bionlp shared task 2016. In Claire Nedellec, Robert Bossy, and Jin-Dong Kim, editors, *Proceedings of the 4th BioNLP Shared Task Workshop, BioNLP 2016*, pages 12–22. Association for Computational Linguistics, 2016. doi: 10.18653/v1/W16-3002. URL https://doi.org/10.18653/v1/W16-3002.

Adji B. Dieng, Chong Wang, Jianfeng Gao, and John W. Paisley. TopicRNN: A recurrent neural network with long-range semantic dependency. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=rJbbOLcex.

Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):677–691, 2017. doi: 10.1109/TPAMI.2016.2599174. URL https://doi.org/10.1109/TPAMI.2016.2599174.

Daxiang Dong, Hua Wu, Wei He, Dianhai Yu, and Haifeng Wang. Multi-task learning for multiple language translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 1723–1732, 2015. URL http://aclweb.org/anthology/P/P15/P15-1166.pdf.

Cícero Nogueira dos Santos, Bing Xiang, and Bowen Zhou. Classifying relations by ranking with convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pages 626–634. The Association for

Computer Linguistics, 2015. URL http://aclweb.org/anthology/P/P15/P15-1061.pdf.

Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *CoRR*, abs/1702.08608, 2017. URL https://arxiv.org/pdf/1702.08608.pdf.

Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 4829–4837. IEEE Computer Society, 2016. doi: 10.1109/CVPR.2016.522. URL https://doi.org/10.1109/CVPR.2016.522.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for text classification. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers*, pages 31–36. Association for Computational Linguistics, 2018. URL https://aclanthology.info/papers/P18-2006/p18-2006.

Jeffrey L. Elman. Finding structure in time. *Cognitive Science*, 14(2):179–211, 1990. doi: 10.1207/s15516709cog1402_1. URL https://doi.org/10.1207/s15516709cog1402_1.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009. URL https://www.researchgate.net/publication/265022827_Visualizing_Higher-Layer_Features_of_a_Deep_Network.

Dumitru Erhan, Yoshua Bengio, Aaron C. Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660, 2010. URL https://dl.acm.org/citation.cfm?id=1756025.

Oren Etzioni, Michael J. Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artif. Intell.*, 165(1):91–134, 2005. URL https://doi.org/10.1016/j.artint.2005.03.001.

Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam. Open information extraction: The second generation. In

Toby Walsh, editor, *IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence, Barcelona, Catalonia, Spain, July 16-22, 2011*, pages 3–10. IJCAI/AAAI, 2011. doi: 10.5591/978-1-57735-516-8/IJCAI11-012. URL https://doi.org/10.5591/978-1-57735-516-8/IJCAI11-012.

Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1535–1545. Association for Computational Linguistics, 2011. URL http://aclweb.org/anthology/D11-1142.

Ronen Feldman and Benjamin Rosenfeld. Boosting unsupervised relation extraction by using NER. In Dan Jurafsky and Éric Gaussier, editors, *EMNLP 2006, Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, 22-23 July 2006, Sydney, Australia*, pages 473–481. Association for Computational Linguistics, 2006. URL http://www.aclweb.org/anthology/W06-1656.

Christiane Fellbaum. Wordnet: An electronic lexical database and some of its applications, 1998.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 363–370. Association for Computational Linguistics, 2005. URL http://aclweb.org/anthology/P05-1045.

Yoav Freund and David Haussler. Unsupervised learning of distributions on binary vectors using two layer networks. In *Advances in neural information processing systems*, pages 912–919. Morgan Kaufmann, 1992. URL https://www.researchgate.net/publication/2821917_Unsupervised_Learning_of_Distributions_on_Binary_Vectors_Using_Two_Layer_Networks.

Brendan J. Frey, Geoffrey E. Hinton, and Peter Dayan. Does the wake-sleep algorithm produce good density estimators? In David S. Touretzky, Michael Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8, NIPS, Denver, CO, USA, November 27-30, 1995*, pages 661–667. MIT Press, 1995. URL https://www.researchgate.net/publication/2638404_Does_the_Wake-sleep_Algorithm_Produce_Good_Density_Estimators.

Brendan J Frey, J Frey Brendan, and Brendan J Frey. *Graphical models for machine learning and digital communication*. MIT press, 1998.

Pablo Gamallo, Marcos Garcia, and Santiago Fernández-Lanza. Dependency-based open information extraction. In *Proceedings of the Joint Workshop on Unsupervised and Semi-Supervised Learning in NLP*, pages 10–18. Association for Computational Linguistics, 2012. URL http://aclweb.org/anthology/W12-0702.

Peter V. Gehler, Alex Holub, and Max Welling. The rate adapting poisson model for information retrieval and object recognition. In William W. Cohen and Andrew Moore, editors, *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML*, volume 148 of *ACM International Conference Proceeding Series*, pages 337–344. Association for Computing Machinery, 2006. doi: 10.1145/1143844.1143887. URL https://doi.org/10.1145/1143844.1143887.

Matthew Gerber and Joyce Chai. Beyond nombank: A study of implicit arguments for nominal predicates. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1583–1592. Association for Computational Linguistics, 2010. URL http://aclweb.org/anthology/P10-1160.

Yoav Goldberg and Michael Elhadad. An efficient algorithm for easy-first non-directional dependency parsing. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2-4, 2010, Los Angeles, California, USA*, pages 742–750. The Association for Computational Linguistics, 2010. URL http://www.aclweb.org/anthology/N10-1115.

Sujatha Das Gollapalli and Xiaoli Li. Emnlp versus acl: Analyzing nlp research over time. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2002–2006. Association for Computational Linguistics, 2015. doi: 10.18653/v1/D15-1235. URL http://aclweb.org/anthology/D15-1235.

Christoph Goller and Andreas Küchler. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of International Conference on Neural Networks (ICNN'96)*, volume 1, pages 347–352. IEEE, 1996. URL https://pdfs.semanticscholar.org/794e/6ed81d21f1bf32a0fd3be05c44c1fa362688.pdf.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *CoRR*, abs/1412.6572, 2014. URL http://arxiv.org/abs/1412.6572.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword. *Linguistic Data Consortium, Philadelphia*, 4(1):34, 2003. URL https://catalog.ldc.upenn.edu/LDC2003T05.

Edouard Grave, Tomas Mikolov, Armand Joulin, and Piotr Bojanowski. Bag of tricks for efficient text classification. In Mirella Lapata, Phil Blunsom, and Alexander Koller, editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017, Valencia, Spain, April 3-7, 2017, Volume 2: Short Papers*, pages 427–431. Association for Computational Linguistics, 2017. URL https://aclanthology.info/papers/E17-2068/e17-2068.

Alex Graves, Marcus Liwicki, S. Fernandez, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(5):855–868, 2009. URL https://doi.org/10.1109/TPAMI.2008.137.

Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 6645–6649. IEEE, 2013. doi: 10.1109/ICASSP.2013.6638947. URL https://doi.org/10.1109/ICASSP.2013.6638947.

Ralph Grishman and Beth Sundheim. Message understanding conference- 6: A brief history. In *COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, pages 466–471, 1996. URL http://aclweb.org/anthology/C96-1079.

Cyril Grouin. Identification of mentions and relations between bacteria and biotope from pubmed abstracts. In Claire Nedellec, Robert Bossy, and Jin-Dong Kim, editors, *Proceedings of the 4th BioNLP Shared Task Workshop, BioNLP 2016*, pages 64–72. Association for Computational Linguistics, 2016. doi: 10.18653/v1/W16-3008. URL https://doi.org/10.18653/v1/W16-3008.

Pankaj Gupta. Lecture-05: Recurrent neural networks (Deep Learning & AI). Technical report, Lecture, Ludwig-Maximilians-University of Munich, Germany, 2019. URL https://www.researchgate.net/

publication/329029430_Lecture-05_Recurrent_Neural_Networks_Deep_Learning_AI.

Pankaj Gupta and Hinrich Schütze. LISA: Explaining recurrent neural network judgments via layer-wise semantic accumulation and example to pattern transformation. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 154–164. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/W18-5418.

Pankaj Gupta, Vijay Shankar Pendluri, and Ishant Vats. Summarizing text by ranking text units according to shallow linguistic features. In *Advanced communication technology (ICACT), 2011 13th international conference on*, pages 1620–1625. IEEE, 2011. URL https://ieeexplore.ieee.org/document/5746114/.

Pankaj Gupta, Thomas Runkler, Heike Adel, Bernt Andrassy, Hans-Georg Zimmermann, and Hinrich Schütze. Deep learning methods for the extraction of relations in natural language text. Technical report, Technical report, Technical University of Munich, Germany, 2015a. URL https://www.researchgate.net/publication/316684826_Deep_Learning_Methods_for_the_Extraction_of_Relations_in_Natural_Language_Text.

Pankaj Gupta, Thomas Runkler, and Bernt Andrassy. Keyword learning for classifying requirements in tender documents. Technical report, Technical report, Technical University of Munich, Germany, 2015b. URL https://www.researchgate.net/publication/316684951_Keyword_Learning_for_Classifying_Requirements_in_Tender_Documents.

Pankaj Gupta, Udhayaraj Sivalingam, Sebastian Pölsterl, and Nassir Navab. Identifying patients with diabetes using discriminative restricted boltzmann machines. Technical report, Technical report, Technical University of Munich, Germany, 2015c. URL https://www.researchgate.net/publication/316685123_Identifying_Patients_with_Diabetes_using_Discriminative_Restricted_Boltzmann_Machines.

Pankaj Gupta, Hinrich Schütze, and Bernt Andrassy. Table filling multi-task recurrent neural network for joint entity and relation extraction. In *Proceedings*

of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 2537–2547. The COLING 2016 Organizing Committee, 2016. URL http://aclweb.org/anthology/C16-1239.

Pankaj Gupta, Bernt Andrassy, and Hinrich Schütze. Replicated siamese LSTM in ticketing system for similarity learning and retrieval in asymmetric texts. In *Proceedings of the Third Workshop on Semantic Deep Learning*, pages 1–11. Association for Computational Linguistics, 2018a. URL http://aclweb.org/anthology/W18-4001.

Pankaj Gupta, Subburam Rajaram, Hinrich Schütze, and Bernt Andrassy. Deep temporal-recurrent-replicated-softmax for topical trends over time. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1079–1089. Association for Computational Linguistics, 2018b. doi: 10.18653/v1/N18-1098. URL http://aclweb.org/anthology/N18-1098.

Pankaj Gupta, Benjamin Roth, and Hinrich Schütze. Joint bootstrapping machines for high confidence relation extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 26–36. Association for Computational Linguistics, 2018c. doi: 10.18653/v1/N18-1003. URL http://aclweb.org/anthology/N18-1003.

Pankaj Gupta, Yatin Chaudhary, Florian Buettner, and Hinrich Schütze. Document informed neural autoregressive topic models with distributional prior. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, 2019a. URL www.aaai.org/Papers/AAAI/2019/AAAI-GuptaPankaj1.4838.pdf.

Pankaj Gupta, Yatin Chaudhary, Florian Buettner, and Hinrich Schütze. textTOvec: Deep contextualized neural autoregressive topic models of language with distributed compositional prior. In *International Conference on Learning Representations (ICLR 2019)*, 2019b. URL https://arxiv.org/abs/1810.03947.

Pankaj Gupta, Subburam Rajaram, Hinrich Schütze, and Thomas Runkler. Neural relation extraction within and across sentence boundaries. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, 2019c. URL https://arxiv.org/abs/1810.05102.

Sonal Gupta and Christopher Manning. Improved pattern learning for bootstrapped entity extraction. In *Proceedings of the Eighteenth Conference on Computational Natural Language Learning*, pages 98–108. Association for Computational Linguistics, 2014a. doi: 10.3115/v1/W14-1611. URL http://aclweb.org/anthology/W14-1611.

Sonal Gupta and Christopher Manning. SPIED: Stanford pattern based information extraction and diagnostics. In *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*, pages 38–44. Association for Computational Linguistics, 2014b. doi: 10.3115/v1/W14-3106. URL http://aclweb.org/anthology/W14-3106.

Aric A. Hagberg, Daniel A. Schult, and Pieter J. Swart. Exploring network structure, dynamics, and function using networkx. Technical report, Technical report, Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008. URL https://conference.scipy.org/proceedings/scipy2008/paper_2/full_text.pdf.

David Hall, Daniel Jurafsky, and Christopher D. Manning. Studying the history of ideas using topic models. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 363–371. Association for Computational Linguistics, 2008. URL http://aclweb.org/anthology/D08-1038.

Henk Harkema, Robert Gaizauskas, Mark Hepple, Angus Roberts, Ian Roberts, Neil Davis, and Yikun Guo. A large scale terminology resource for biomedical text processing. In *HLT-NAACL 2004 Workshop: Linking Biological Literature, Ontologies and Databases*, 2004. URL http://aclweb.org/anthology/W04-3110.

Takaaki Hasegawa, Satoshi Sekine, and Ralph Grishman. Discovering relations among named entities from large corpora. In Donia Scott, Walter Daelemans, and Marilyn A. Walker, editors, *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain.*, pages 415–422. Association for Computational Linguistics, 2004. URL http://aclweb.org/anthology/P/P04/P04-1053.pdf.

W. Keith Hastings. *Monte Carlo sampling methods using Markov chains and their applications*. Oxford University Press, 1970.

Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 15th International Conference on Computational Linguistics*, 1992. URL http://aclweb.org/anthology/C92-2082.

Georg Heigold, Stalin Varanasi, Günter Neumann, and Josef van Genabith. How robust are character-based word embeddings in tagging and MT against wrod scramlbing or randdm nouse? In Colin Cherry and Graham Neubig, editors, *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas, AMTA 2018, Boston, MA, USA, March 17-21, 2018 - Volume 1: Research Papers*, pages 68–80. Association for Machine Translation in the Americas, 2018. URL https://aclanthology.info/papers/W18-1807/w18-1807.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions (SEW-2009)*, pages 94–99. Association for Computational Linguistics, 2009. URL http://aclweb.org/anthology/W09-2415.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38. Association for Computational Linguistics, 2010. URL http://aclweb.org/anthology/S10-1006.

Geoffrey E. Hinton. Learning distributed representations of concepts. In *Proceedings of the Annual Conference of the Cognitive Science Society*, volume 1, page 12. Amherst, MA, 1986. URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.408.7684&rep=rep1&type=pdf.

Geoffrey E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002. doi: 10.1162/089976602760128018. URL https://doi.org/10.1162/089976602760128018.

Geoffrey E. Hinton. A practical guide to training restricted boltzmann machines. In Grégoire Montavon, Genevieve B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade - Second Edition*, volume 7700 of *Lecture Notes in Computer Science*, pages 599–619. Springer, 2012. doi: 10.1007/978-3-642-35289-8_32. URL https://doi.org/10.1007/978-3-642-35289-8_32.

Geoffrey E. Hinton and Ruslan R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. URL https://www.cs.toronto.edu/~hinton/science.pdf.

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006. doi: 10.1162/neco.2006.18.7.1527. URL https://doi.org/10.1162/neco.2006.18.7.1527.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke S. Zettlemoyer, and Daniel S. Weld. Knowledge-based weak supervision for information extraction of overlapping relations. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 541–550. The Association for Computer Linguistics, 2011. URL http://www.aclweb.org/anthology/P11-1055.

Thomas Hofmann. Probabilistic latent semantic analysis. In Kathryn B. Laskey and Henri Prade, editors, *UAI '99: Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, Stockholm, Sweden, July 30 - August 1, 1999*, pages 289–296. Morgan Kaufmann, 1999. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=179&proceeding_id=15.

John J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79 (8):2554–2558, 1982. URL https://www.pnas.org/content/79/8/2554.

Tomoharu Iwata, Takeshi Yamada, Yasushi Sakurai, and Naonori Ueda. Online multiscale dynamic topic models. In Bharat Rao, Balaji Krishnapuram, Andrew Tomkins, and Qiang Yang, editors, *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 663–672. Association for Computing Machinery, 2010. doi: 10.1145/1835804.1835889. URL https://doi.org/10.1145/1835804.1835889.

Jing Jiang and ChengXiang Zhai. A systematic exploration of the feature space for relation extraction. In Candace L. Sidner, Tanja Schultz, Matthew Stone,

and ChengXiang Zhai, editors, *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, April 22-27, 2007, Rochester, New York, USA*, pages 113–120. The Association for Computational Linguistics, 2007. URL http://www.aclweb.org/anthology/N07-1015.

Xiaotian Jiang, Quan Wang, Peng Li, and Bin Wang. Relation extraction with multi-instance multi-label convolutional neural networks. In Nicoletta Calzolari, Yuji Matsumoto, and Rashmi Prasad, editors, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 1471–1480. Association for Computational Linguistics, 2016. URL http://aclweb.org/anthology/C/C16/C16-1139.pdf.

Michael I. Jordan. Serial order: A parallel distributed processing approach. In *Advances in psychology*, volume 121, pages 471–495. Elsevier, 1997. URL http://digitalcollections.library.cmu.edu/awweb/awarchive?type=file&item=49573.

Dan Jurafsky and James H. Martin. *Speech and language processing: an introduction to natural language processing, computational linguistics, and speech recognition, 2nd Edition*. Prentice Hall series in artificial intelligence. Prentice Hall, Pearson Education International, 2009. URL http://www.worldcat.org/oclc/315913020.

Nanda Kambhatla. Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction. In *Proceedings of the ACL Interactive Poster and Demonstration Sessions, 22*. Association for Computational Linguistics, 2004. URL http://aclweb.org/anthology/P04-3022.

Rohit J. Kate and Raymond Mooney. Joint entity and relation extraction using card-pyramid parsing. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning*, pages 203–212. Association for Computational Linguistics, 2010. URL http://aclweb.org/anthology/W10-2924.

Henry J. Kelley. Gradient theory of optimal flight paths. *Ars Journal*, 30(10): 947–954, 1960. URL https://www.gwern.net/docs/statistics/decision/1960-kelley.pdf.

Eliyahu Kiperwasser and Yoav Goldberg. Easy-first dependency parsing with hierarchical tree lstms. *Transactions of the Association for Computational Lin-*

guistics, 4:445–461, 2016. URL https://transacl.org/ojs/index.php/tacl/article/view/798.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Skip-thought vectors. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems*, pages 3294–3302, 2015. URL http://papers.nips.cc/paper/5950-skip-thought-vectors.

Johannes Kirschnick, Holmer Hemsen, and Volker Markl. JEDI: joint entity and relation detection using type inference. In Sameer Pradhan and Marianna Apidianaki, editors, *Proceedings of ACL-2016 System Demonstrations, Berlin, Germany, August 7-12, 2016*, pages 61–66. Association for Computational Linguistics, 2016. doi: 10.18653/v1/P16-4011. URL https://doi.org/10.18653/v1/P16-4011.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.*, pages 1106–1114, 2012. URL https://www.researchgate.net/publication/267960550_ImageNet_Classification_with_Deep_Convolutional_Neural_Networks.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Carla E. Brodley and Andrea Pohoreckyj Danyluk, editors, *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001), Williams College, Williamstown, MA, USA, June 28 - July 1, 2001*, pages 282–289. Morgan Kaufmann, 2001. URL https://repository.upenn.edu/cgi/viewcontent.cgi?article=1162&context=cis_papers.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270. Association for Computational Linguistics, 2016.

doi: 10.18653/v1/N16-1030. URL http://aclweb.org/anthology/N16-1030.

Hugo Larochelle and Yoshua Bengio. Classification using discriminative restricted boltzmann machines. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 536–543. Association for Computing Machinery, 2008. doi: 10.1145/1390156.1390224. URL https://doi.org/10.1145/1390156.1390224.

Hugo Larochelle and Stanislas Lauly. A neural autoregressive topic model. In Peter L. Bartlett, Fernando C. N. Pereira, Christopher J. C. Burges, Léon Bottou, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems*, pages 2717–2725, 2012. URL http://papers.nips.cc/paper/4613-a-neural-autoregressive-topic-model.

Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In Geoffrey J. Gordon, David B. Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS*, volume 15 of *JMLR Proceedings*, pages 29–37. JMLR.org, 2011. URL http://www.jmlr.org/proceedings/papers/v15/larochelle11a/larochelle11a.pdf.

Jey Han Lau, Timothy Baldwin, and Trevor Cohn. Topically driven neural language model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 355–365. Association for Computational Linguistics, 2017. doi: 10.18653/v1/P17-1033. URL http://aclweb.org/anthology/P17-1033.

Stanislas Lauly, Yin Zheng, Alexandre Allauzen, and Hugo Larochelle. Document neural autoregressive distribution estimation. *Journal of Machine Learning Research*, 18:113:1–113:24, 2017. URL http://jmlr.org/papers/v18/16-017.html.

Hoang-Quynh Le, Duy-Cat Can, Sinh T. Vu, Thanh Hai Dang, Mohammad Taher Pilehvar, and Nigel Collier. Large-scale exploration of neural relation classification architectures. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2266–2277. Association for Computational

Linguistics, 2018. URL https://aclanthology.info/papers/D18-1250/d18-1250.

Quoc V. Le and Tomas Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1188–1196. JMLR.org, 2014. URL http://jmlr.org/proceedings/papers/v32/le14.html.

Quoc V. Le, Navdeep Jaitly, and Geoffrey E. Hinton. A simple way to initialize recurrent networks of rectified linear units. *CoRR*, abs/1504.00941, 2015. URL http://arxiv.org/abs/1504.00941.

Yann LeCun, Fu Jie Huang, and Léon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2004), with CD-ROM, 27 June - 2 July 2004, Washington, DC, USA*, pages 97–104. IEEE Computer Society, 2004. URL http://doi.ieeecomputersociety.org/10.1109/CVPR.2004.144.

Joohong Lee, Sangwoo Seo, and Yong Suk Choi. Semantic relation classification via bidirectional LSTM networks with entity-aware attention using latent entity typing. *CoRR*, abs/1901.08163, 2019. URL http://arxiv.org/abs/1901.08163.

Jiwei Li, Xinlei Chen, Eduard Hovy, and Dan Jurafsky. Visualizing and understanding neural models in nlp. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 681–691. Association for Computational Linguistics, 2016. doi: 10.18653/v1/N16-1082. URL http://aclweb.org/anthology/N16-1082.

Qi Li and Heng Ji. Incremental joint extraction of entity mentions and relations. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 402–412. Association for Computational Linguistics, 2014. doi: 10.3115/v1/P14-1038. URL http://aclweb.org/anthology/P14-1038.

Ming Liang and Xiaolin Hu. Recurrent convolutional neural network for object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3367–3375. IEEE Computer Society, 2015. URL https://doi.org/10.1109/CVPR.2015.7298958.

Winston Lin, Roman Yangarber, and Ralph Grishman. Bootstrapped learning of semantic classes from positive and negative examples. In *Proceedings of ICML-2003 Workshop on The Continuum from Labeled to Unlabeled Data*, page 21, 2003. URL https://nlp.cs.nyu.edu/pubs/papers/lin-icml03.ps.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016. URL http://aclweb.org/anthology/P/P16/P16-1200.pdf.

Zachary C. Lipton. The mythos of model interpretability. *Commun. ACM*, 61(10): 36–43, 2018. doi: 10.1145/3233231. URL https://doi.org/10.1145/3233231.

Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and Houfeng Wang. A dependency-based neural network for relation classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 2: Short Papers*, pages 285–290. The Association for Computer Linguistics, 2015a. URL http://aclweb.org/anthology/P/P15/P15-2047.pdf.

Yang Liu, Furu Wei, Sujian Li, Heng Ji, Ming Zhou, and Houfeng WANG. A dependency-based neural network for relation classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 285–290. Association for Computational Linguistics, 2015b. doi: 10.3115/v1/P15-2047. URL http://aclweb.org/anthology/P15-2047.

Huma Lodhi, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Christopher J. C. H. Watkins. Text classification using string kernels. *Journal of Machine Learning Research*, 2:419–444, 2002. URL http://www.jmlr.org/papers/v2/lodhi02a.html.

Xuezhe Ma and Eduard Hovy. End-to-end sequence labeling via bi-directional lstm-cnns-crf. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074.

Association for Computational Linguistics, 2016. doi: 10.18653/v1/P16-1101. URL http://aclweb.org/anthology/P16-1101.

Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9:2579–2605, 2008. URL http://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60. Association for Computational Linguistics, 2014. doi: 10.3115/v1/P14-5010. URL http://aclweb.org/anthology/P14-5010.

Christopher D. Manning and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT Press, 2001. ISBN 978-0-262-13360-9.

Christopher D Manning, Christopher D Manning, and Hinrich Schütze. *Foundations of statistical natural language processing*. MIT press, 1999.

James Martens and Ilya Sutskever. Training deep and recurrent networks with hessian-free optimization. In Grégoire Montavon, Genevieve B. Orr, and Klaus-Robert Müller, editors, *Neural Networks: Tricks of the Trade - Second Edition*, volume 7700 of *Lecture Notes in Computer Science*, pages 479–535. Springer, 2012. URL https://doi.org/10.1007/978-3-642-35289-8_27.

Mausam, Michael Schmitz, Stephen Soderland, Robert Bart, and Oren Etzioni. Open language learning for information extraction. In Jun'ichi Tsujii, James Henderson, and Marius Pasca, editors, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 523–534. Association for Computational Linguistics, 2012. URL http://www.aclweb.org/anthology/D12-1048.

Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4): 115–133, 1943. URL https://link.springer.com/article/10.1007%2FBF02478259.

Farrokh Mehryary, Jari Björne, Sampo Pyysalo, Tapio Salakoski, and Filip Ginter. Deep learning with minimal training data: Turkunlp entry in the bionlp shared task 2016. In *Proceedings of the 4th BioNLP Shared Task Workshop*, pages

73–81. Association for Computational Linguistics, 2016. doi: 10.18653/v1/W16-3009. URL http://aclweb.org/anthology/W16-3009.

Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Z. Hakkani-Tür, Xiaodong He, Larry P. Heck, Gökhan Tür, Dong Yu, and Geoffrey Zweig. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech & Language Processing*, 23(3):530–539, 2015. doi: 10.1109/TASLP.2014.2383614. URL https://doi.org/10.1109/TASLP.2014.2383614.

Filipe Mesquita, Jordan Schmidek, and Denilson Barbosa. Effectiveness and efficiency of open relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 447–457. Association for Computational Linguistics, 2013. URL http://aclweb.org/anthology/D13-1043.

Nicholas Metropolis and Stanislaw Ulam. The monte carlo method. *Journal of the American statistical association*, 44(247):335–341, 1949.

Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur. Recurrent neural network based language model. In Takao Kobayashi, Keikichi Hirose, and Satoshi Nakamura, editors, *Eleventh Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1045–1048. ISCA, 2010. URL http://www.isca-speech.org/archive/interspeech_2010/i10_1045.html.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceedings of the International Conference on Learning Representations, ICLR*, 2013a. URL http://arxiv.org/abs/1301.3781.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *Proceeding of the International Conference on Learning Representations Workshop Track*. ICLR, 2013b. URL http://arxiv.org/abs/1301.3781.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems*, pages 3111–3119, 2013c. URL https://arxiv.org/pdf/1310.4546.pdf.

Tomas Mikolov, Armand Joulin, Sumit Chopra, Michaël Mathieu, and Marc'Aurelio Ranzato. Learning longer memory in recurrent neural networks. *CoRR*, abs/1412.7753, 2014. URL http://arxiv.org/abs/1412.7753.

George A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995. doi: 10.1145/219717.219748. URL http://doi.acm.org/10.1145/219717.219748.

Yao Ming, Shaozu Cao, Ruixiang Zhang, Zhen Li, Yuanzhe Chen, Yangqiu Song, and Huamin Qu. Understanding hidden memories of recurrent neural networks. In Brian Fisher, Shixia Liu, and Tobias Schreck, editors, *IEEE Conference on Visual Analytics Science and Technology, VAST*, pages 13–24. IEEE Computer Society, 2017. doi: 10.1109/VAST.2017.8585721. URL https://doi.org/10.1109/VAST.2017.8585721.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. Distant supervision for relation extraction without labeled data. In Keh-Yih Su, Jian Su, and Janyce Wiebe, editors, *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*, pages 1003–1011. The Association for Computer Linguistics, 2009a. URL http://www.aclweb.org/anthology/P09-1113.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011. Association for Computational Linguistics, 2009b. URL http://aclweb.org/anthology/P09-1113.

Jeff Mitchell and Mirella Lapata. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429, 2010. URL https://doi.org/10.1111/j.1551-6709.2010.01106.x.

Makoto Miwa and Mohit Bansal. End-to-end relation extraction using lstms on sequences and tree structures. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1105–1116. Association for Computational Linguistics, 2016. doi: 10.18653/v1/P16-1105. URL http://aclweb.org/anthology/P16-1105.

Makoto Miwa and Yutaka Sasaki. Modeling joint entity and relation extraction with table representation. In *Proceedings of the 2014 Conference on Empirical*

Methods in Natural Language Processing (EMNLP), pages 1858–1869. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1200. URL http://aclweb.org/anthology/D14-1200.

Andriy Mnih and Geoffrey E. Hinton. A scalable hierarchical distributed language model. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 8-11, 2008*, pages 1081–1088. Curran Associates, Inc., 2008. URL https://www.cs.toronto.edu/~amnih/papers/hlbl_final.pdf.

Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018. doi: 10.1016/j.dsp.2017.10.011. URL https://doi.org/10.1016/j.dsp.2017.10.011.

Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In Robert G. Cowell and Zoubin Ghahramani, editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics, AISTATS 2005, Bridgetown, Barbados, January 6-8, 2005*. Society for Artificial Intelligence and Statistics, 2005. URL http://www.gatsby.ucl.ac.uk/aistats/fullpapers/208.pdf.

Amr Mousa and Björn Schuller. Contextual bidirectional long short-term memory recurrent neural network language models: A generative approach to sentiment analysis. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1023–1032. Association for Computational Linguistics, 2017. URL http://aclweb.org/anthology/E17-1096.

Jonas Mueller and Aditya Thyagarajan. Siamese recurrent architectures for learning sentence similarity. In Dale Schuurmans and Michael P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 2786–2792. AAAI Press, 2016. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12195.

Nina Narodytska and Shiva Prasad Kasiviswanathan. Simple black-box adversarial attacks on deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 1310–1318. IEEE Computer Society, 2017. doi: 10.1109/CVPRW.2017.172. URL https://doi.org/10.1109/CVPRW.2017.172.

Claire Nédellec, Robert Bossy, Jin-Dong Kim, Jung-Jae Kim, Tomoko Ohta, Sampo Pyysalo, and Pierre Zweigenbaum. Overview of bionlp shared task 2013. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 1–7. Association for Computational Linguistics, 2013. URL http://aclweb.org/anthology/W13-2001.

Anh Nguyen, Alexey Dosovitskiy, Jason Yosinski, Thomas Brox, and Jeff Clune. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems*, pages 3387–3395, 2016. URL https://arxiv.org/abs/1605.09304.

Dat P. T. Nguyen, Yutaka Matsuo, and Mitsuru Ishizuka. Relation extraction from wikipedia using subtree mining. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence*, volume 22, pages 1414–1420. AAAI Press, 2007. URL http://www.aaai.org/Library/AAAI/2007/aaai07-224.php.

Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics*, 3:299–313, 2015a. URL https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/582.

Dat Quoc Nguyen, Richard Billingsley, Lan Du, and Mark Johnson. Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics*, 3:299–313, 2015b. URL http://aclweb.org/anthology/Q15-1022.

Thien Huu Nguyen and Ralph Grishman. Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 39–48. Association for Computational Linguistics, 2015. doi: 10.3115/v1/W15-1506. URL http://aclweb.org/anthology/W15-1506.

Kamal Nigam, Andrew McCallum, Sebastian Thrun, and Tom M. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000. doi: 10.1023/A:1007692713085. URL https://doi.org/10.1023/A:1007692713085.

Nicolas Papernot, Patrick D. McDaniel, Ian J. Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In Ramesh Karri, Ozgur Sinanoglu, Ahmad-Reza Sadeghi, and Xun Yi, editors, *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017, Abu Dhabi, United Arab Emirates, April 2-6, 2017*, pages 506–519. Association for Computing Machinery, 2017. doi: 10.1145/3052973.3053009. URL https://doi.org/10.1145/3052973.3053009.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. English gigaword. *Linguistic Data Consortium*, 2011. URL https://catalog.ldc.upenn.edu/LDC2011T07.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 1310–1318. JMLR.org, 2013. URL http://jmlr.org/proceedings/papers/v28/pascanu13.html.

Sachin Pawar, Girish K. Palshikar, and Pushpak Bhattacharyya. Relation extraction : A survey. *CoRR*, abs/1712.05191, 2017. URL http://arxiv.org/abs/1712.05191.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. Cross-sentence n-ary relation extraction with graph lstms. *Transactions of the Association for Computational Linguistics*, 5:101–115, 2017. URL http://aclweb.org/anthology/Q17-1008.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1162. URL http://aclweb.org/anthology/D14-1162.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics, 2018. doi: 10.18653/v1/N18-1202. URL http://aclweb.org/anthology/N18-1202.

James Petterson, Alexander J. Smola, Tibério S. Caetano, Wray L. Buntine, and Shravan M. Narayanamurthy. Word features for latent dirichlet allocation. In John D. Lafferty, Christopher K. I. Williams, John Shawe-Taylor, Richard S. Zemel, and Aron Culotta, editors, *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems*, pages 1921–1929. Curran Associates, Inc., 2010. URL http://papers.nips.cc/paper/4094-word-features-for-latent-dirichlet-allocation.

Hieu Pham, Thang Luong, and Christopher Manning. Learning distributed representations for multilingual text sequences. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 88–94. Association for Computational Linguistics, 2015. doi: 10.3115/v1/W15-1512. URL http://aclweb.org/anthology/W15-1512.

Gualtiero Piccinini. The first computational theory of mind and brain: A close look at mcculloch and pitts's "logical calculus of ideas immanent in nervous activity". *Synthese*, 141(2):175–215, 2004. URL https://doi.org/10.1023/B:SYNT.0000043018.52445.3e.

Tony A. Plate. Holographic reduced representations. *IEEE Trans. Neural Networks*, 6(3):623–641, 1995. doi: 10.1109/72.377968. URL https://doi.org/10.1109/72.377968.

Hoifung Poon and Pedro M. Domingos. Unsupervised semantic parsing. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, 6-7 August 2009, Singapore, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1–10. Association for Computational Linguistics, 2009. URL http://www.aclweb.org/anthology/D09-1001.

Nina Pörner, Hinrich Schütze, and Benjamin Roth. Evaluating neural network explanation methods using hybrid documents and morphosyntactic agreement. In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 340–350. Association for Computational Linguistics, 2018. URL https://aclanthology.info/papers/P18-1032/p18-1032.

Iulian Pruteanu-Malinici, Lu Ren, John William Paisley, Eric Wang, and Lawrence Carin. Hierarchical bayesian modeling of topics in time-stamped documents. *IEEE transactions on pattern analysis and machine intelligence*, 32

(6):996–1011, 2010. doi: 10.1109/TPAMI.2009.125. URL https://doi.org/10.1109/TPAMI.2009.125.

Chris Quirk and Hoifung Poon. Distant supervision for relation extraction beyond the sentence boundary. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 1171–1182. Association for Computational Linguistics, 2017. URL http://aclweb.org/anthology/E17-1110.

Tapani Raiko, Li Yao, KyungHyun Cho, and Yoshua Bengio. Iterative neural autoregressive distribution estimator nade-k. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 325–333, 2014. URL https://arxiv.org/abs/1406.1485.

Paulo E. Rauber, Alexandre X. Falcão, and Alexandru C. Telea. Visualizing time-dependent data using dynamic t-sne. In *Eurographics Conference on Visualization, EuroVis 2016, Short Papers, Groningen, The Netherlands, 6-10 June 2016.*, pages 73–77, 2016. doi: 10.2312/eurovisshort.20161164. URL https://doi.org/10.2312/eurovisshort.20161164.

Deepak Ravichandran and Eduard H. Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA.*, pages 41–47. Association for Computational Linguistics, 2002. URL http://www.aclweb.org/anthology/P02-1006.pdf.

Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 1135–1144. Association for Computing Machinery, 2016. doi: 10.1145/2939672.2939778. URL https://doi.org/10.1145/2939672.2939778.

Sebastian Riedel, Limin Yao, and Andrew McCallum. Modeling relations and their mentions without labeled text. In José L. Balcázar, Francesco Bonchi, Aristides Gionis, and Michèle Sebag, editors, *Machine Learning and Knowledge Discovery in Databases, European Conference, ECML PKDD 2010, Barcelona, Spain, September 20-24, 2010, Proceedings, Part III*, volume 6323

of *Lecture Notes in Computer Science*, pages 148–163. Springer, 2010. URL https://doi.org/10.1007/978-3-642-15939-8_10.

Ellen Riloff. Automatically generating extraction patterns from untagged text. In William J. Clancey and Daniel S. Weld, editors, *Proceedings of the Thirteenth National Conference on Artificial Intelligence and Eighth Innovative Applications of Artificial Intelligence Conference, AAAI 96, IAAI 96, Portland, Oregon, USA, August 4-8, 1996, Volume 2.*, pages 1044–1049. AAAI Press / The MIT Press, 1996. URL http://www.aaai.org/Library/AAAI/1996/aaai96-155.php.

Bryan Rink and Sanda M. Harabagiu. UTD: classifying semantic relations by combining lexical and semantic resources. In Katrin Erk and Carlo Strapparava, editors, *Proceedings of the 5th International Workshop on Semantic Evaluation, SemEval@ACL 2010, Uppsala University, Uppsala, Sweden, July 15-16, 2010*, pages 256–259. The Association for Computer Linguistics, 2010. URL http://aclweb.org/anthology/S/S10/S10-1057.pdf.

Michael Röder, Andreas Both, and Alexander Hinneburg. Exploring the space of topic coherence measures. In Xueqi Cheng, Hang Li, Evgeniy Gabrilovich, and Jie Tang, editors, *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining, WSDM*, pages 399–408. Association for Computing Machinery, 2015. doi: 10.1145/2684822.2685324. URL https://doi.org/10.1145/2684822.2685324.

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958. URL https://psycnet.apa.org/record/1959-09865-001.

Dan Roth and Wen-tau Yih. A linear programming formulation for global inference in natural language tasks. In *Proceedings of the Eighth Conference on Computational Natural Language Learning (CoNLL-2004) at HLT-NAACL 2004*, 2004. URL http://aclweb.org/anthology/W04-2401.

Dan Roth and Wen-tau Yih. Global inference for entity and relation identification via a linear programming formulation. *Introduction to statistical relational learning*, pages 553–580, 2007. URL https://www.researchgate.net/publication/228949116_1_Global_Inference_for_Entity_and_Relation_Identification_via_a_Linear_Programming_Formulation.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016. URL http://arxiv.org/abs/1609.04747.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning internal representations by error propagation. Technical report, Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985. URL https://apps.dtic.mil/dtic/tr/fulltext/u2/a164453.pdf.

David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 5 (3):1, 1988. URL https://www.iro.umontreal.ca/~vincentp/ift3395/lectures/backprop_old.pdf.

Ankan Saha and Vikas Sindhwani. Learning evolving and emerging topics in social media: a dynamic nmf approach with temporal regularization. In Eytan Adar, Jaime Teevan, Eugene Agichtein, and Yoelle Maarek, editors, *Proceedings of the Fifth International Conference on Web Search and Web Data Mining, WSDM*, pages 693–702. Association for Computing Machinery, 2012. doi: 10.1145/2124295.2124376. URL https://doi.org/10.1145/2124295.2124376.

Mehran Sahami and Timothy D. Heilman. A web-based kernel function for measuring the similarity of short text snippets. In Les Carr, David De Roure, Arun Iyengar, Carole A. Goble, and Michael Dahlin, editors, *Proceedings of the 15th international conference on World Wide Web, WWW*, pages 377–386. Association for Computing Machinery, 2006. doi: 10.1145/1135777.1135834. URL https://doi.org/10.1145/1135777.1135834.

Hasim Sak, Andrew W. Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. In Haizhou Li, Helen M. Meng, Bin Ma, Engsiong Chng, and Lei Xie, editors, *15th Annual Conference of the International Speech Communication Association (INTERSPEECH 2014): Celebrating the Diversity of Spoken Languages*, pages 338–342. International Speech Communication Association, 2014. URL http://www.isca-speech.org/archive/interspeech_2014/i14_0338.html.

Keisuke Sakaguchi, Kevin Duh, Matt Post, and Benjamin Van Durme. Robust word reocginiton via semi-character recurrent neural network. In Satinder P. Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 3281–3287. AAAI Press, 2017. URL http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14332.

Ruslan Salakhutdinov and Geoffrey E. Hinton. Replicated softmax: An undirected topic model. In Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta, editors, *Advances in Neural Information Processing Systems 22: 23rd Annual Conference on Neural Information Processing Systems*, pages 1607–1614. Curran Associates, Inc., 2009. URL http://papers.nips.cc/paper/3856-replicated-softmax-an-undirected-topic-model.

Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey E. Hinton. Restricted boltzmann machines for collaborative filtering. In Zoubin Ghahramani, editor, *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 791–798. Association for Computing Machinery, 2007. doi: 10.1145/1273496.1273596. URL https://doi.org/10.1145/1273496.1273596.

Mostafa A. Salama, Aboul Ella Hassanien, and Aly A. Fahmy. Deep belief network for clustering and classification of a continuous data. In *Proceedings of the IEEE International Symposium on Signal Processing and Information Technology, ISSPIT 2010, December 15-18,2010, Luxor, Egypt*, pages 473–477. IEEE Computer Society, 2010. doi: 10.1109/ISSPIT.2010.5711759. URL https://doi.org/10.1109/ISSPIT.2010.5711759.

Aaron Schein, Hanna M. Wallach, and Mingyuan Zhou. Poisson-gamma dynamical systems. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems*, pages 5006–5014, 2016. URL http://papers.nips.cc/paper/6083-poisson-gamma-dynamical-systems.

Mike Schuster and Kuldip K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997. doi: 10.1109/78.650093. URL https://doi.org/10.1109/78.650093.

Michael L. Seltzer and Jasha Droppo. Multi-task learning in deep neural networks for improved phoneme recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 6965–6969. IEEE, 2013. doi: 10.1109/ICASSP.2013.6639012. URL https://doi.org/10.1109/ICASSP.2013.6639012.

Yatian Shen and Xuanjing Huang. Attention-based convolutional neural network for semantic relation extraction. In Nicoletta Calzolari, Yuji Matsumoto, and

Rashmi Prasad, editors, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2526–2536. Association for Computational Linguistics, 2016. URL http://aclweb.org/anthology/C/C16/C16-1238.pdf.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. Learning important features through propagating activation differences. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 3145–3153. Proceedings of Machine Learning Research, 2017. URL http://proceedings.mlr.press/v70/shrikumar17a.html.

Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013. URL http://arxiv.org/abs/1312.6034.

Gaurav Singh and Parminder Bhatia. Relation extraction using explicit context conditioning. *CoRR*, abs/1902.09271, 2019. URL http://arxiv.org/abs/1902.09271.

Sameer Singh, Sebastian Riedel, Brian Martin, Jiaping Zheng, and Andrew McCallum. Joint inference of entities, relations, and coreference. In Fabian M. Suchanek, Sebastian Riedel, Sameer Singh, and Partha Pratim Talukdar, editors, *Proceedings of the 2013 workshop on Automated knowledge base construction, AKBC@CIKM 13, San Francisco, California, USA, October 27-28, 2013*, pages 1–6. Association for Computing Machinery, 2013. URL https://doi.org/10.1145/2509558.2509559.

Paul Smolensky. Information processing in dynamical systems: Foundations of harmony theory. In *Parallel Distributed Processing*, volume 1, pages 194–281. MIT Press, Cambridge, 1986. URL https://apps.dtic.mil/dtic/tr/fulltext/u2/a620727.pdf.

Richard Socher, Eric H. Huang, Jeffrey Pennington, Andrew Y. Ng, and Christopher D. Manning. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In John Shawe-Taylor, Richard S. Zemel, Peter L. Bartlett, Fernando C. N. Pereira, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 801–809,

2011a. URL https://www.researchgate.net/publication/228452494_Dynamic_Pooling_and_Unfolding_Recursive_Autoencoders_for_Paraphrase_Detection.

Richard Socher, Cliff Chiung-Yu Lin, Andrew Y. Ng, and Christopher D. Manning. Parsing natural scenes and natural language with recursive neural networks. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 129–136. Omnipress, 2011b.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. Semantic compositionality through recursive matrix-vector spaces. In Jun'ichi Tsujii, James Henderson, and Marius Pasca, editors, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211. Association for Computational Linguistics, 2012. URL http://www.aclweb.org/anthology/D12-1110.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1631–1642. Association for Computational Linguistics, 2013. URL https://aclanthology.info/papers/D13-1170/d13-1170.

Richard Socher, Andrej Karpathy, Quoc V. Le, Christopher D. Manning, and Andrew Y. Ng. Grounded compositional semantics for finding and describing images with sentences. *Transactions of the Association for Computational Linguistics*, 2:207–218, 2014. URL http://aclweb.org/anthology/Q14-1017.

Stephen Soderland, Brendan Roof, Bo Qin, Shi Xu, Mausam, and Oren Etzioni. Adapting open information extraction to domain-specific relations. *AI Magazine*, 31(3):93–102, 2010. URL http://www.aaai.org/ojs/index.php/aimagazine/article/view/2305.

Akash Srivastava and Charles Sutton. Autoencoding variational inference for topic models. In *5th International Conference on Learning Representations, ICLR*, 2017. URL https://arxiv.org/pdf/1703.01488.pdf.

Veselin Stoyanov and Jason Eisner. Easy-first coreference resolution. In *Proceedings of COLING 2012*, pages 2519–2534. The COLING 2012 Organizing Committee, 2012. URL http://aclweb.org/anthology/C12-1154.

Ang Sun, Ralph Grishman, and Satoshi Sekine. Semi-supervised relation extraction with large-scale word clustering. In Dekang Lin, Yuji Matsumoto, and Rada Mihalcea, editors, *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA*, pages 521–529. The Association for Computer Linguistics, 2011. URL http://www.aclweb.org/anthology/P11-1053.

Mihai Surdeanu. Overview of the TAC2013 knowledge base population evaluation: English slot filling and temporal slot filling. In *Proceedings of the Sixth Text Analysis Conference, TAC*. NIST, 2013. URL http://www.nist.gov/tac/publications/2013/additional.papers/KBP2013_English_and_Temporal_Slot_Filling_overview.TAC2013.proceedings.pdf.

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. Multi-instance multi-label learning for relation extraction. In Jun'ichi Tsujii, James Henderson, and Marius Pasca, editors, *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*, pages 455–465. Association for Computational Linguistics, 2012. URL http://www.aclweb.org/anthology/D12-1042.

Ilya Sutskever and Geoffrey E. Hinton. Learning multilevel distributed representations for high-dimensional sequences. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics, AISTATS 2007, San Juan, Puerto Rico, March 21-24, 2007*, volume 2 of *JMLR Proceedings*, pages 548–555. JMLR.org, 2007. URL http://jmlr.org/proceedings/papers/v2/sutskever07a.html.

Ilya Sutskever, Geoffrey E. Hinton, and Graham W. Taylor. The recurrent temporal restricted boltzmann machine. In Daphne Koller, Dale Schuurmans, Yoshua Bengio, and Léon Bottou, editors, *Advances in Neural Information Processing Systems 21, Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems*, pages 1601–1608. Curran Associates, Inc., 2008. URL https://www.cs.toronto.edu/~hinton/absps/rtrbm.pdf.

Ilya Sutskever, James Martens, and Geoffrey E. Hinton. Generating text with recurrent neural networks. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 1017–1024. Omnipress, 2011. URL https://www.cs.utoronto.ca/~ilya/pubs/2011/LANG-RNN.pdf.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112, 2014. URL https://arxiv.org/abs/1409.3215.

Richard Sutton. Two problems with back propagation and other steepest descent learning procedures for networks. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society, 1986*, pages 823–832, 1986. URL https://ci.nii.ac.jp/naid/10022346408.

Kumutha Swampillai and Mark Stevenson. Inter-sentential relations in information extraction corpora. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*. European Languages Resources Association (ELRA), 2010. URL http://www.lrec-conf.org/proceedings/lrec2010/pdf/905_Paper.pdf.

Kumutha Swampillai and Mark Stevenson. Extracting relations within and across sentences. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 25–32. Association for Computational Linguistics, 2011. URL http://aclweb.org/anthology/R11-1004.

Duyu Tang, Bing Qin, and Ting Liu. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1422–1432. Association for Computational Linguistics, 2015a. doi: 10.18653/v1/D15-1167. URL http://aclweb.org/anthology/D15-1167.

Duyu Tang, Bing Qin, and Ting Liu. Document modeling with gated recurrent neural network for sentiment classification. In Lluís Màrquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1422–

1432. The Association for Computational Linguistics, 2015b. URL http://aclweb.org/anthology/D/D15/D15-1167.pdf.

Zhiyuan Tang, Ying Shi, Dong Wang, Yang Feng, and Shiyue Zhang. Memory visualization for gated recurrent neural networks in speech recognition. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP*, pages 2736–2740. IEEE, 2017. doi: 10.1109/ICASSP.2017.7952654. URL https://doi.org/10.1109/ICASSP.2017.7952654.

Graham W. Taylor, Geoffrey E. Hinton, and Sam T. Roweis. Modeling human motion using binary latent variables. In Bernhard Schölkopf, John C. Platt, and Thomas Hofmann, editors, *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems*, pages 1345–1352. MIT Press, 2006. URL http://www2.egr.uh.edu/~zhan2/ECE6111_Fall2015/modeling%20human%20motion%20using%20binary%20latent%20variables.pdf.

Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In William W. Cohen, Andrew McCallum, and Sam T. Roweis, editors, *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9, 2008*, volume 307 of *ACM International Conference Proceeding Series*, pages 1064–1071. Association for Computing Machinery, 2008. doi: 10.1145/1390156.1390290. URL https://doi.org/10.1145/1390156.1390290.

Tijmen Tieleman and Geoffrey E. Hinton. Using fast weights to improve persistent contrastive divergence. In Andrea Pohoreckyj Danyluk, Léon Bottou, and Michael L. Littman, editors, *Proceedings of the 26th Annual International Conference on Machine Learning, ICML 2009, Montreal, Quebec, Canada, June 14-18, 2009*, volume 382 of *ACM International Conference Proceeding Series*, pages 1033–1040. Association for Computing Machinery, 2009. doi: 10.1145/1553374.1553506. URL https://doi.org/10.1145/1553374.1553506.

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394. Association for Computational Linguistics, 2010. URL http://aclweb.org/anthology/P10-1040.

Matthew Turk and Alex Pentland. Eigenfaces for recognition. *Journal of cognitive neuroscience*, 3(1):71–86, 1991. URL https:

//www.mitpressjournals.org/doi/abs/10.1162/jocn.1991.3.1.71.

Peter D. Turney. Mining the web for synonyms: PMI-IR versus LSA on TOEFL. In Luc De Raedt and Peter A. Flach, editors, *Machine Learning: EMCL 2001, 12th European Conference on Machine Learning, Freiburg, Germany, September 5-7, 2001, Proceedings*, volume 2167 of *Lecture Notes in Computer Science*, pages 491–502. Springer, 2001. doi: 10.1007/3-540-44795-4_42. URL https://doi.org/10.1007/3-540-44795-4_42.

Benigno Uria, Iain Murray, and Hugo Larochelle. RNADE: the real-valued neural autoregressive density-estimator. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pages 2175–2183, 2013. URL https://arxiv.org/pdf/1306.0186.pdf.

Benigno Uria, Iain Murray, and Hugo Larochelle. A deep and tractable density estimator. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 467–475. JMLR.org, 2014. URL http://jmlr.org/proceedings/papers/v32/uria14.html.

Benigno Uria, Marc-Alexandre Côté, Karol Gregor, Iain Murray, and Hugo Larochelle. Neural autoregressive distribution estimation. *Journal of Machine Learning Research*, 17:205:1–205:37, 2016. URL http://jmlr.org/papers/v17/16-272.html.

Shikhar Vashishth, Rishabh Joshi, Sai Suman Prayaga, Chiranjib Bhattacharyya, and Partha Talukdar. RESIDE: improving distantly-supervised neural relation extraction using side information. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1257–1266. Association for Computational Linguistics, 2018. URL https://aclanthology.info/papers/D18-1157/d18-1157.

Ngoc Thang Vu, Heike Adel, Pankaj Gupta, and Hinrich Schütze. Combining recurrent and convolutional neural networks for relation classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages

534–539. Association for Computational Linguistics, 2016a. doi: 10.18653/v1/N16-1065. URL http://aclweb.org/anthology/N16-1065.

Ngoc Thang Vu, Pankaj Gupta, Heike Adel, and Hinrich Schütze. Bi-directional recurrent neural network with ranking loss for spoken language understanding. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2016, Shanghai, China, March 20-25, 2016*, pages 6060–6064. IEEE, 2016b. doi: 10.1109/ICASSP.2016.7472841. URL https://doi.org/10.1109/ICASSP.2016.7472841.

Hanna M. Wallach. Topic modeling: beyond bag-of-words. In *Proceedings of the Twenty-Third International Conference Machine Learning, (ICML 2006)*, volume 148 of *ACM International Conference Proceeding Series*, pages 977–984. Association for Computing Machinery, 2006. doi: 10.1145/1143844.1143967. URL https://doi.org/10.1145/1143844.1143967.

Xiaojun Wan and Jianguo Xiao. Single document keyphrase extraction using neighborhood knowledge. In Dieter Fox and Carla P. Gomes, editors, *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI*, pages 855–860. AAAI Press, 2008. URL http://www.aaai.org/Library/AAAI/2008/aaai08-136.php.

Linlin Wang, Zhu Cao, Gerard de Melo, and Zhiyuan Liu. Relation classification via multi-level attention cnns. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016a. URL http://aclweb.org/anthology/P/P16/P16-1123.pdf.

Wenlin Wang, Zhe Gan, Wenqi Wang, Dinghan Shen, Jiaji Huang, Wei Ping, Sanjeev Satheesh, and Lawrence Carin. Topic compositional neural language model. In Amos J. Storkey and Fernando Pérez-Cruz, editors, *International Conference on Artificial Intelligence and Statistics, AISTATS 2018*, volume 84, pages 356–365. Proceedings of Machine Learning Research, 2018. URL http://proceedings.mlr.press/v84/wang18a.html.

Xuerui Wang and Andrew McCallum. Topics over time: a non-markov continuous-time model of topical trends. In Tina Eliassi-Rad, Lyle H. Ungar, Mark Craven, and Dimitrios Gunopulos, editors, *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 424–433. Association for Computing Machinery, 2006. doi: 10.1145/1150402.1150450. URL https://doi.org/10.1145/1150402.1150450.

Xuerui Wang, Andrew McCallum, and Xing Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Proceedings of the 7th IEEE International Conference on Data Mining (ICDM 2007), October 28-31, 2007, Omaha, Nebraska, USA*, pages 697–702. IEEE Computer Society, 2007. doi: 10.1109/ICDM.2007.86. URL https://doi.org/10.1109/ICDM.2007.86.

Yequan Wang, Minlie Huang, xiaoyan zhu, and Li Zhao. Attention-based lstm for aspect-level sentiment classification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 606–615. Association for Computational Linguistics, 2016b. doi: 10.18653/v1/D16-1058. URL http://aclweb.org/anthology/D16-1058.

Max Welling, Michal Rosen-Zvi, and Geoffrey E. Hinton. Exponential family harmoniums with an application to information retrieval. In *Advances in Neural Information Processing Systems 17 [Neural Information Processing Systems, NIPS 2004, December 13-18, 2004, Vancouver, British Columbia, Canada]*, pages 1481–1488, 2004. URL https://www.ics.uci.edu/~welling/publications/papers/GenHarm3.pdf.

Paul J Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990. URL http://axon.cs.byu.edu/~martinez/classes/678/Papers/Werbos_BPTT.pdf.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. Towards universal paraphrastic sentence embeddings. In *Proceedings of the International Conference on Learning Representations*, 2016. URL http://arxiv.org/abs/1511.08198.

Minguang Xiao and Cong Liu. Semantic relation classification via hierarchical recurrent neural network with attention. In Nicoletta Calzolari, Yuji Matsumoto, and Rashmi Prasad, editors, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 1254–1263. Association for Computational Linguistics, 2016. URL http://aclweb.org/anthology/C/C16/C16-1119.pdf.

Eric P. Xing, Rong Yan, and Alexander G. Hauptmann. Mining associated text and images with dual-wing harmoniums. In *Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 633–641. Association for Uncertainty in Artificial Intelligence Press, 2005. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=1184&proceeding_id=21.

Kun Xu, Yansong Feng, Songfang Huang, and Dongyan Zhao. Semantic relation classification via convolutional neural networks with simple negative sampling. In Lluís Màrquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 536–540. The Association for Computational Linguistics, 2015a. URL http://aclweb.org/anthology/D/D15/D15-1062.pdf.

Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin. Classifying relations via long short term memory networks along shortest dependency paths. In Lluís Màrquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1785–1794. The Association for Computational Linguistics, 2015b. URL http://aclweb.org/anthology/D/D15/D15-1206.pdf.

Yan Xu, Ran Jia, Lili Mou, Ge Li, Yunchuan Chen, Yangyang Lu, and Zhi Jin. Improved relation classification by deep recurrent neural networks with data augmentation. In Nicoletta Calzolari, Yuji Matsumoto, and Rashmi Prasad, editors, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 1461–1470. Association for Computational Linguistics, 2016. URL http://aclweb.org/anthology/C/C16/C16-1138.pdf.

Ming-Hsuan Yang, Narendra Ahuja, and David J. Kriegman. Face recognition using kernel eigenfaces. In *Proceedings of the 2000 International Conference on Image Processing, ICIP 2000, Vancouver, BC, Canada, September 10-13, 2000*, pages 37–40. IEEE, 2000. doi: 10.1109/ICIP.2000.900886. URL https://doi.org/10.1109/ICIP.2000.900886.

Zhilin Yang, Ye Yuan, Yuexin Wu, William W. Cohen, and Ruslan Salakhutdinov. Review networks for caption generation. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 2361–2369, 2016. URL http://papers.nips.cc/paper/6167-review-networks-for-caption-generation.

Zijun Yao, Yifan Sun, Weicong Ding, Nikhil Rao, and Hui Xiong. Dynamic word embeddings for evolving semantic discovery. In Yi Chang, Chengxiang Zhai,

Yan Liu, and Yoelle Maarek, editors, *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM*, pages 673–681. Association for Computing Machinery, 2018. doi: 10.1145/3159652.3159703. URL https://doi.org/10.1145/3159652.3159703.

Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. Abcnn: Attention-based convolutional neural network for modeling sentence pairs. *Transactions of the Association for Computational Linguistics*, 4:259–272, 2016. URL http://aclweb.org/anthology/Q16-1019.

Katsumasa Yoshikawa, Sebastian Riedel, Tsutomu Hirao, Masayuki Asahara, and Yuji Matsumoto. Coreference based event-argument relation extraction on biomedical text. *Journal of Biomedical Semantics*, 2(S-5):S6, 2011. URL http://www.jbiomedsem.com/content/2/S5/S6.

Mo Yu, Matthew Gormley, and Mark Dredze. Factor-based compositional embedding models. In *NIPS Workshop on Learning Semantics*, pages 95–101, 2014. URL https://www.cs.cmu.edu/~mgormley/papers/yu+gormley+dredze.nipsw.2014.pdf.

Xiaofeng Yu and Wai Lam. Jointly identifying entities and extracting relations in encyclopedia text via A graphical model approach. In Chu-Ren Huang and Dan Jurafsky, editors, *COLING 2010, 23rd International Conference on Computational Linguistics, Posters Volume, 23-27 August 2010, Beijing, China*, pages 1399–1407. Chinese Information Processing Society of China, 2010. URL http://aclweb.org/anthology/C/C10/C10-2160.pdf.

Matthew D. Zeiler. ADADELTA: An adaptive learning rate method. *CoRR*, abs/1212.5701, 2012. URL http://arxiv.org/abs/1212.5701.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, volume 8689 of *Lecture Notes in Computer Science*, pages 818–833. Springer, 2014. doi: 10.1007/978-3-319-10590-1_53. URL https://doi.org/10.1007/978-3-319-10590-1_53.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. Relation classification via convolutional deep neural network. In Jan Hajic and Junichi Tsujii, editors, *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 2335–2344. Association for Computational Linguistics, 2014. URL http://aclweb.org/anthology/C/C14/C14-1220.pdf.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. Distant supervision for relation extraction via piecewise convolutional neural networks. In Lluís Màrquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton, editors, *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1753–1762. The Association for Computational Linguistics, 2015. URL http://aclweb.org/anthology/D/D15/D15-1203.pdf.

Dongxu Zhang and Dong Wang. Relation classification via recurrent neural network. *CoRR*, abs/1508.01006, 2015. URL http://arxiv.org/abs/1508.01006.

Kai Zhang, Wei Wu, Haocheng Wu, Zhoujun Li, and Ming Zhou. Question retrieval with high quality answers in community question answering. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management, CIKM*, pages 371–380. Association for Computing Machinery, 2014. doi: 10.1145/2661829.2661908. URL https://doi.org/10.1145/2661829.2661908.

Min Zhang, Jie Zhang, and Jian Su. Exploring syntactic features for relation extraction using a convolution tree kernel. In Robert C. Moore, Jeff A. Bilmes, Jennifer Chu-Carroll, and Mark Sanderson, editors, *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 4-9, 2006, New York, New York, USA*. The Association for Computational Linguistics, 2006a. URL http://aclweb.org/anthology/N/N06/N06-1037.pdf.

Min Zhang, Jie Zhang, Jian Su, and GuoDong Zhou. A composite kernel to extract relations between entities with both flat and structured features. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 825–832. Association for Computational Linguistics, 2006b. URL http://aclweb.org/anthology/P06-1104.

Xingxing Zhang and Mirella Lapata. Chinese poetry generation with recurrent neural networks. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 670–680. Association for Computational Linguistics, 2014. URL http://aclweb.org/anthology/D/D14/D14-1074.pdf.

Yu Zhang and Dit-Yan Yeung. A convex formulation for learning task relationships in multi-task learning. In Peter Grünwald and Peter Spirtes, editors, *Proceedings of the Twenty-Sixth Conference on Uncertainty in Artificial Intelligence, UAI*, pages 733–442. Association for Uncertainty in Artificial Intelligence Press, 2010. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=2117&proceeding_id=26.

Yuhao Zhang, Victor Zhong, Danqi Chen, Gabor Angeli, and Christopher D. Manning. Position-aware attention and supervised data improve slot filling. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 35–45. Association for Computational Linguistics, 2017. URL https://aclanthology.info/papers/D17-1004/d17-1004.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. Graph convolution over pruned dependency trees improves relation extraction. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 2205–2215. Association for Computational Linguistics, 2018. URL https://aclanthology.info/papers/D18-1244/d18-1244.

Jiang Zhao, Tiantian Zhu, and Man Lan. ECNU: One stone two birds: Ensemble of heterogenous measures for semantic relatedness and textual entailment. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pages 271–277. Association for Computational Linguistics, 2014. doi: 10.3115/v1/S14-2044. URL http://aclweb.org/anthology/S14-2044.

Le Zhao. Modeling and solving term mismatch for full-text retrieval. *SIGIR Forum*, 46(2):117–118, 2012. doi: 10.1145/2422256.2422277. URL https://doi.org/10.1145/2422256.2422277.

Suncong Zheng, Yuexing Hao, Dongyuan Lu, Hongyun Bao, Jiaming Xu, Hongwei Hao, and Bo Xu. Joint entity and relation extraction based on a hybrid neural network. *Neurocomputing*, 257:59–66, 2017a. doi: 10.1016/j.neucom.2016.12.075. URL https://doi.org/10.1016/j.neucom.2016.12.075.

Suncong Zheng, Feng Wang, Hongyun Bao, Yuexing Hao, Peng Zhou, and Bo Xu. Joint extraction of entities and relations based on a novel tagging

scheme. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1227–1236. Association for Computational Linguistics, 2017b. URL https://doi.org/10.18653/v1/P17-1113.

Yin Zheng, Yu-Jin Zhang, and Hugo Larochelle. A supervised neural autoregressive topic model for simultaneous image classification and annotation. *CoRR*, abs/1305.5306, 2013. URL http://arxiv.org/abs/1305.5306.

Yin Zheng, Yu-Jin Zhang, and Hugo Larochelle. A deep and autoregressive approach for topic modeling of multimodal data. *IEEE transactions on pattern analysis and machine intelligence*, 38(6):1056–1069, 2016. doi: 10.1109/TPAMI.2015.2476802. URL https://doi.org/10.1109/TPAMI.2015.2476802.

Guodong Zhou, Jian Su, Jie Zhang, and Min Zhang. Exploring various knowledge in relation extraction. In Kevin Knight, Hwee Tou Ng, and Kemal Oflazer, editors, *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, pages 427–434. The Association for Computer Linguistics, 2005. URL http://aclweb.org/anthology/P/P05/P05-1053.pdf.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics, 2016. URL http://aclweb.org/anthology/P/P16/P16-2034.pdf.

Qi Zhu, Xiang Ren, Jingbo Shang, Yu Zhang, Ahmed El-Kishky, and Jiawei Han. Integrating local context and global cohesiveness for open information extraction. In J. Shane Culpepper, Alistair Moffat, Paul N. Bennett, and Kristina Lerman, editors, *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 42–50. Association for Computing Machinery, 2019. doi: 10.1145/3289600.3291030. URL https://doi.org/10.1145/3289600.3291030.

Curriculum Vitae

Education

12/2015 – Present	**Doctoral Candidate** (LMU Munich, Germany) Specializations: Computer Science/Computational Linguistics Research focus: Information Extraction, Distributional representation for text, Topic Modeling, Intersection of Neural and Probabilistic Graphical models
09/2013 – 11/2015	**Master of Science** (TUM Munich, Germany) Specializations: Computer Science (Informatics)
09/2006 – 05/2010	**Bachelor of Science** (Amity University, India) Specializations: Information Technology

Practical Experience

11/2017 - Present	**Research Scientist** (Siemens AG, Munich, Germany) Research & Development: Machine Learning/NLP
10/2016 - 01/2017	**Research Intern** (IBM Research, Zurich, Switzerland) Research: Large scale unsupervised relation extraction
12/2015 - 10/2017	**Doctoral Candidate** (Siemens AG, Munich, Germany) Research: Information Extraction and Deep Learning
03/2014 - 11/2015	**Working Student** (Siemens AG, Munich, Germany) Research: Deep Learning for Information Extraction
06/2014 - 09/2015	**Research Assistant** (TUM, Munich, Germany) Implementations of neural and probabilistic networks
12/2011 - 09/2013	**Senior Software Engineer** (Aricent Technologies, India) Data Structures and C programmer, Networking
06/2010 - 12/2011	**Software Engineer** (Wipro Technologies, India) Data Structures and C programmer