Consistency and Completeness of Knowledge Acquired by Language Models

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

eingereicht von
Nora Kassner

München, den 09.09.2024

Erstgutachter: *Hinrich Schütze, Professor*
Zweitgutachter: *Antoine Bosselut, Assistant Professor*
Drittgutachter: *Pasquale Minervini, Lecturer*

Tag der Einreichung: 09.09.2024
Tag der mündlichen Prüfung: 14.08.2025

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, den 09.09.2024

Nora Kassner

Abstract

The introduction of Deep Learning based pretrained Language Models (LMs) has brought large improvements throughout Natural Language Processing (NLP). Pretrained in an unsupervised fashion to predict missing words from incomplete pieces of text, they can subsequently be adapted to any task of interest and replace the need to develop and train specialized architectures. Their ability to excel is rooted in the fact that pretrained LMs store a multitude of task relevant knowledge parametrically. This knowledge is not limited to linguistic capabilities. Pretrained LMs were also shown to acquire significant amounts of world and commonsense knowledge.

We explore two aspects of world/commonsense knowledge acquired by LMs. First, **consistency**, that is, we expect a model's behavior to be consistent across a set of implied queries. Second, **completeness** with respect to the factual queries which may be put to the model.

We analyze knowledge **consistency** with respect to negation, adversarial distractors, multilinguality, paraphrasing and commonsense reasoning and find that although the LMs under investigation contain significant amounts of world knowledge, they are prone to answer factual implications inconsistently and self-contradictory. As a result, it can be hard to identify what the model actually "believes" about the world, making it susceptible to inconsistent behavior. Building on this, we develop a new architecture where an LM is enhanced with a 'symbolic executive' - an evolving, symbolic memory of prior beliefs. For new incoming queries the LM has the ability to reflect back on related beliefs, enabling it to improve over time.

To improve knowledge **completeness**, we explore different ways of integrating knowledge not acquired by the model. i) We enhance an LM with a retrieval component over external knowledge sources for improved Question Answering. ii) We explore LMs' reasoning capabilities during pretraining to deduce knowledge not explicitly seen. iii) We build a model that integrates novel entities into LM-based Entity Linking systems.

In analyzing and improving knowledge consistency and completeness, this thesis makes a significant step towards LM-based architectures with a systematic notion of belief, enabling them to construct a more coherent picture of the world, and improve over time without model retraining.

Zusammenfassung

Die Einführung der Deep Learning basierten, Pretrained Language Models (LMs) führte zu großen Druchbrüchen im Bereich der Natürlichen Sprachverarbeitung. Pretrained LMs ersätzten weitgehend den Bedarf an Aufgaben spezialisierten Architekturen. Stattdessen kann ein LM nach einer allgemeinen Pretrainingsphase, in der das Modell lernt Textstücke zu vervollständigen, durch Finetuning oder Prompting, leicht an unterschiedlichste Aufgaben angepasst werden. Das LM erlernt wäherend der Pretrainingsphase eine Vielzahl an aufgabenrelevantem Wissens parametrisch, welches dann aufgabenspezifisch abgefragt werden kann. Bemerkenswert ist, dass sich das erlernte Wissen nicht auf linguistische Fähigkeiten beschränkt, sondern auch enzyklopädisches Wissen und Commonsense beinhaltet.

Wir untersuchen zwei Aspekte des von LMs erlernten Wissens. Erstens, **Konsistenz**, wo wir erwarten, dass sich das LM im Bezug auf implizierte Fakten konsitent verhält. Zweitens, **Vollstkändigkeit** im Bezug auf mögliche Wissensfragen, die man dem Modell stellen könnte.

Wir analysieren **Konsistenz** bezüglich Negation, irreführendem Priming, Mehrsprachigkeit, Paraphrasierung und Commonsensezusammenhängen. Trotz der Menge angesammelten Wissens, zeigt sich, dass die von uns untersuchten LMs anfällig für inkonsistentes Verhalten sind. Folglich ist es schwierig aus einzelnen Outputs eines LMs, die sich wiedersprechen können, auf das grundlegende Erfassen von Fakten zu schließen.

Aufbauend darauf entwickeln wir eine neue Architektur, bei der ein LM durch eine symbolische Komponente erweitert wird. Diese symbolische Komponente ist ein sich entwickelndes, symbolisches Gedächtnis vorausgehender Outputs. Für neu eintreffende Anfragen hat das LM die Fähigkeit, sich auf verwandte, vorrausgehende Outputs zu stützen, was es ermöglicht, sich im Laufe der Zeit zu verbessern.

Um die **Vollstkändigkeit** des Wissens zu erweitern, erkunden wir verschiedene Möglichkeiten, nicht erfasstes Wissen zu integrieren. i) Wir erweitern ein LM mit einer Abfragekomponente, die Zugriff auf externe Wissensquellen hat. ii) Wir erforschen die Fähigkeit eines LMs während des Pretrainings aus zusammenhängenden Fakten Schlussfolgerungen zu ziehen und sich somit selbst, nicht explizit gesehenes Wissen, abzuleiten. iii) Wir entwickeln ein Modell, das nicht erfasste Entitäten in LM-basierte Entitätsverknüpfungssysteme integriert.

Durch die Analyse und Verbesserung von Konsistenz und Vollständigkeit des von LM erfassten Wissens leistet diese Arbeit einen bedeutenden Schritt hin zu LM-basierten Architekturen, die ein kohärenteres Bild der Welt erfassen köennen und sich im Laufe der Zeit verbessern.

Acknowledgments

This work would not have been possible without the strong support by many people. First and foremost, I would like to thank my supervisor Hinrich Schütze. Thank you for welcoming me to the field of Natural Language Processing. Your decision to take me on as a student impacted my life immensely. I would not be in the position I am today without your support. I have learned a great deal from you, from conducting my initial experiments to fleshing out my first paper and eventually shaping my own research agenda. Your guidance has been invaluable every step of the way.

Many thanks to Antoine Bosselut and Pasquale Minervini for reviewing the dissertation in such detail and guiding a very interesting discussion during the disputation, and to the whole committee for taking the time for the examination.

Next, I want to thank all my great collaborators and most importantly Yanai Elazar and Peter Clark. Thank you Yanai for reaching out at the very first virtual ACL that led to our collaboration and turned into a friendship. Thank you Peter for offering me my first PhD internship, that led to our long and fruitful collaboration. Your support and the fact that you were so vocal about it had an immense impact on my career.

The years at CIS have been great. I will miss discussing research with my colleagues, stimulating reading groups and fun lunches. Specifically, thanks to Philipp Dufter and Alexandra Chronopoulou. Philipp, thank you for being my steady office mate and collaborator and the continued exchange also after your time at CIS. Alexandra, thank you for your friendship that gave great support through all the self-doubts and struggles that a PhD student faces.

Thanks to Peggy Hobmaier and Susanne Grienberger for help in maneuvering the administrative jungle of German universities and Thomas Schäfer for being an incredibly supportive and fast technical admin. Our center would not be the same without all of you.

I am grateful for the Munich Center for Machine Learning for supporting my PhD and connecting me to a great batch of research peers. This research was funded by the European Research Council (# 740516) and by the German Federal Ministry of Education and Research (BMBF) under Grant No 01IS18036A.

I also want to acknowledge the amazing Allen Institute for AI for being such a special place that fosters free, open and collaborative research and constitutes a career catapult for so many junior NLP researchers. I am thankful and honoured for having been acknowledged with the Outstanding Intern of the Year Award in 2021.

Finally, I want to thank family and friends outside of work for supporting me through the ups and downs that a PhD journey naturally comes with:

Ann Christin and Auguste, it's been more than a decade since we met and started our science journey together. Thank you for your understanding friendship and support. I am looking forward to walking through the next decades together.

I am grateful to my dad for laying the foundation of my interest in science, the encouragement to pursue it as a career and for the in-depth interest in my work. You are the person that had the greatest influence on shaping who I am. I am grateful to my mum and sister for their patience, emotional support and for keeping me grounded in the world outside of science. I could not be at this point without the support of my family.

Lastly but most importantly, I thank Karan for his love, understanding and patience. Thank you, for believing in me, supporting me throughout and cheering my success.

Publications and Declaration of Co-Authorship

Chapter 2 corresponds to the following publication:

> **Nora Kassner**, Hinrich Schütze. *Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly.* In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 7811–7818. 2020.

I conceived of the original research contributions and performed all implementations and evaluations. I wrote the initial draft of the article and did most of the subsequent corrections. I regularly discussed this work with my co-author who assisted me in improving the draft.

Chapter 3 corresponds to the following publication:

> Yanai Elazar, **Nora Kassner**, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, Yoav Goldberg. *Measuring and improving consistency in pretrained language models.* In Transactions of the Association for Computational Linguistics (TACL), Volume 9, pp. 1012–1031. 2021.

Yanai Elazar conceived of the original research contributions. Yanai Elazar, Shauli Ravfogel, Abhilasha Ravichander and I created the dataset jointly. Yanai Elazar implemented and performed the evaluation. I designed and performed experimentation on a consistency improved model . Yanai Elazar wrote the initial draft of the article and Yanai Elazar, Shauli Ravfogel, Abhilasha Ravichander, Hinrich Schütze, Yoav Goldberg and I assisted in improving the draft.

Chapter 4 corresponds to the following publication:

> **Nora Kassner***, Philipp Dufter*, Hinrich Schütze. *Multilingual LAMA: Investigating knowledge in multilingual pretrained language models.*

In Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics (EACL), pp. 3250–3258. 2021. *equal contribution. Best Short Paper Award.

Philipp Dufter conceived of the original research contributions. I designed the evaluation and performed the experiments, while Philipp Dufter created the automatic translations of the dataset. Philipp Dufter and I wrote the initial draft of the article and we did most of the subsequent corrections together. We regularly discussed this work with Hinrich Schütze who assisted in improving the draft.

Chapter 5 corresponds to the following publication:

> **Nora Kassner**, Oyvind Tafjord, Hinrich Schütze, Peter Clark. *BeliefBank: Adding Memory to a Pre-Trained Language Model for a Systematic Notion of Belief.* In Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 8849–8861. 2021.

I proposed the original research idea. We jointly developed the model. I performed implementation and evaluation and developed the dataset. I wrote the initial draft of the article and Peter Clark and I did most of the subsequent corrections. We regularly discussed this work with Oyvind Tafjord and Hinrich Schütze.

Chapter 6 corresponds to the following publication:

> **Nora Kassner**, Oyvind Tafjord, Ashish Sabharwal, Kyle Richardson, Hinrich Schütze, Peter Clark. *Language Models with Rationality.* In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 14190–14201. 2023.

This work is a continuation of the BeliefBank research idea. We regularly discussed this work as a group and jointly iterated over the model idea. Oyvind Tafjord queried the Entailer model and I performed implementation and evaluation of the REFLEX system. Peter Clark, Ashish Sabharwal and I wrote most of the paper and did most of the subsequent corrections.

Chapter 7 corresponds to the following publication:

> **Nora Kassner***, Benno Krojer*, Hinrich Schütze. *Are Pretrained Language Models Symbolic Reasoners Over Knowledge?* In Proceedings of the 24th Conference on Computational Natural Language Learning (CoNLL), pp. 552–564. 2020. *equal contribution.

I conceived of the original research contributions. I performed the first proof of concept implementation and evaluation. Benno Krojer performed and implemented subsequent experiments on reasoning which we regularly discussed. I performed and implemented subsequent experiments on memorization. I wrote the initial draft of the article and did most of the subsequent corrections. I regularly discussed this work with Hinrich Schütze who assisted me in improving the draft.

Chapter 8 corresponds to the following publication:

> **Nora Kassner**, Hinrich Schütze. *BERT-kNN: Adding a kNN search component to pretrained language models for better QA*. In Findings of the Association for Computational Linguistics: EMNLP 2020, pp. 3424–3430. 2020.

I conceived of the original research contributions and performed all implementations and evaluations. I wrote the initial draft of the article and did most of the subsequent corrections. I regularly discussed this work with my co-author who assisted me in improving the draft.

Chapter 9 corresponds to the following publication:

> **Nora Kassner**, Fabio Petroni, Mikhail Plekhanov, Sebastian Riedel, Nicola Cancedda. *EDIN: An End-to-end Benchmark and Pipeline for Unknown Entity Discovery and Indexing.* In Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 8659–8673. 2022.

Nicola Cancedda conceived of the original research idea. I conceived of the practical realization, designed and constructed the benchmark, did all of the implementation and evaluation. I regularly discussed this work with my co-authors. I wrote the initial draft of the article and made most of the subsequent edits. Nicola Cancedda helped in editing the paper.

Declaration on Writing Aids

I used ChatGPT (GPT-3.5 and GPT-4o) as an aid to write the introductory part of this thesis. ChatGPT was used as an advanced spelling and grammar checker and to improve choice of words, to restructure sentences and sometimes passages and to improve clarity and readability. Sometimes I provided general instructions and snippets of content and used ChatGPT to piece the passage together. I then carefully edited it to finalize the passage.

Contents

Chapter 1

Introduction

1.1 Outline

This thesis is structured in two parts. In this first, introductory part (Chapter 1), we motivate the research questions, summarize our contributions and provide background information relevant to this thesis. The second part (Chapters 2 to 9) comprises the publications part of this thesis.

The introductory part begins in Section 1.2 with a high level introduction, followed by Sections 1.3 and 1.4 that motivate its approach and contributions more concretely. Section 1.5 gives a more general review of knowledge representations in Language Models and in Sections 1.6 and 1.7, we summarize basic concepts of Deep Learning and Natural Language Processing relevant for this work. The introductory part concludes with Sections 1.8 and 1.9 listing limitations of this thesis and giving an outlook into future research directions.

The second part, from Chapter 2 till 6, corresponds to publications on consistency. Chapters 7 till 9 correspond to publications on completeness.

1.2 Motivation

The introduction of Deep Learning based pretrained Language Models (LMs [1]) in 2018 (Peters et al., 2018; Devlin et al., 2019) marked a significant breakthrough for the field of Natural Language Processing (NLP). These models effectively captured contextual information in text at an unprecedented scale and quickly emerged as universal tools that set new state-of-the-art performance on a wide range of NLP benchmarks (Wang et al., 2019).

[1] Throughout this thesis, we use this acronym to refer specifically to Deep Learning based pretrained Language Models introduced in 2018, and not to any preceding models, such as count-based models.

In the years that followed, researchers continued to push the boundaries of pretrained LMs, resulting in the development of models that grew exponentially in size and were trained on larger amounts of data, starting from the early pretrained models like Elmo (Peters et al., 2018) and BERT (Devlin et al., 2019), consisting of around hundreds of million parameters, reaching to T5 (Raffel et al., 2020) which is the largest model used as part of this thesis and which consists of 11 billion parameters. This trend continues today beyond the scope of this thesis with further advancements in scale, e.g., PaLM (Chowdhery et al., 2022) which surpasses the 500 billion parameter threshold.

These models are pretrained in an unsupervised fashion using the language modeling objective to predict the next word or sequence of words, given words in context.

To create task-specific models, early LMs followed the pretraining-finetuning paradigm which involves making several copies of the model to finetune them on task-specific data. Subsequently, the field shifted to the pretraining-prompting paradigm. This approach eliminates the need to create multiple separately trained copies of a model to make it task-specific. Instead, by providing a few task-specific samples or instructions, the model can be instantly adapted.

The fact that LMs excel on such a wide variety of tasks, combined with the success of prompting that does not require the integration of large amounts of external knowledge, suggests that much of the necessary knowledge is already encoded parametrically in the model.

This knowledge is not limited to linguistic capabilities (Goldberg, 2019; Hewitt and Manning, 2019; Tenney et al., 2019; Elazar et al., 2021). Research also shows their progressing ability to acquire and interact with world and common sense knowledge (Petroni et al., 2019; Davison et al., 2019; Peters et al., 2019; Jiang et al., 2020; Roberts et al., 2020).

How to best represent factual knowledge has been a long standing problem in artificial intelligence research (Brachman and Levesque, 2004). Parametric knowledge acquisition inside LMs offers new and unique opportunities. It enables wide domain coverage, is easily scalable due to their unsupervised construction and is not tied to any specific schema, which is specifically compelling in the context of common sense knowledge (Razniewski et al., 2021).

But with LMs' neural nature comes the challenge that their parametrical knowledge is encoded in a diffused manner, making it difficult to access, interpret, control and maintain.

In this context, this thesis studies two aspects of knowledge acquired by LMs, which we call *consistency*, that is, we expect a model's behavior to be consistent across a set of implied queries and *completeness* with respect to the factual queries which may be put to the model.

Consistency: Work quantifying the amount of factual knowledge captured by

LMs usually evaluates their performance on Question Answering (Q&A) tasks, e.g., Roberts et al. (2020). Although their performance on these tasks is substantial, it is not always clear how (or even if) an answer relates to implied statements and whether the LM is latently constructing an internal "belief [2] system".

Specifically, we study consistency with respect to *negation, adversarial distractors, multilinguality, paraphrasing* and *commonsense reasoning*. With our work on consistency, we show that LMs are prone to answer factual implications inconsistently and in a self-contradictory manner. As a result, it is sometimes hard to pin down what an LM actually "believes[3]", making them susceptible to inconsistent and/or irrational behavior and hard to debug errors. To address this, we embed the LM into a broader system with an 'symbolic executive' – a symbolic memory component in which we make the model's beliefs explicit and with a reasoning mechanism to resolve inconsistencies. This is significant as it is a first step towards LM-based architectures with a systematic notion of belief, enabling them to construct a more coherent picture of the world, and improve over time without model retraining.

Completeness: During pretraining, not every piece of knowledge an LM is exposed to is necessarily captured parametrically and after pretraining novel information arises constantly. In this context, we i) study what governs which knowledge is acquired and which is not, specifically we question whether LMs during pretraining are able to acquire factual knowledge beyond what they see explicitly and ii) build systems that integrate missing knowledge into the model. To study knowledge acquisition in LMs we focus on two mechanisms, reasoning and memorization, where we propose a synthetic framework that allows us to study the causal relationship between the facts present in the training data and the facts learned by the model. We build systems to integrate missing knowledge in the context of two of the most prominent knowledge-intensive NLP tasks, namely Q&A and Entity Linking (EL). Again, our approach is to embed the LM into a broader system consisting of an external knowledge store containing embedding-based representations of novel knowledge and a retrieval mechanism to integrate this information into the task specific system.

[2]Here, we refer to the model's factual opinions as "beliefs" instead of "knowledge" because a model's predictions may be wrong. In general, an agent can be said to believe statement s if it acts as if s was true (Schwitzgebel, 2024). We adopt a simple implementation of this, namely the LM generates s or answers "yes" to the question "s?". Note that other versions could be used.

[3]Here and in what follows, human concepts such as belief have been attributed to language models for ease of exposition. This anthropomorphism should not be taken too literally.

1.3 Consistency of Knowledge

Our central contribution to *knowledge consistency* is twofold. **Measuring Consistency**: We show that although LMs contain significant amounts of knowledge at the first glance, they are prone to inconsistent behaviour. As a result, it can be hard to identify what the model actually believes about the world, making it susceptible to inconsistent behaviour and simple errors. **Improving Consistency:** We introduce a new style of architecture to improve consistency where an LM is extended with a 'symbolic executive', an evolving, symbolic memory and reasoning mechanism. Such a component enables us to materialize belief dependencies and repair latent inconsistencies and therefore construct more coherent pictures of the world.

Measuring Consistency

Instead of following the standard Q&A setting, consisting of evaluating answers to factual queries in isolation, we define sets of implied factual statements and measure accuracy as well as consistency when querying an LM's belief about these statements. We define consistency sets for *negation, adversarial distractors, multilinguality, paraphrasing* and *commonsense reasoning*.

To query for LMs' beliefs, we follow two different approaches depending on the type of model. For early Masked LMs (refer to Section 1.7.4), we follow Petroni et al. (2019) and reformulate knowledge base (KB) (subject, relation, object) triplets into cloze-style natural language queries. To do so, we use natural language template relation (e.g., "X is the capital of Y") and insert Wikidata entities as subjects (X) and mask the object (Y), e.g., "Paris is the capital of [MASK]". This setup matches the input style of the model during pretraining and enables us to query the LM zero-shot (without any finetuning). For Causal LMs (refer to Section 1.7.4), we employ models finetuned for multiple-choice Q&A tasks.

To measure consistency, we define a consistency metric. Even though the exact formulation of our consistency metric varies in our papers, the underlying principle is the same. We measure consistency in terms of conditional violation (Li et al., 2019), namely the fraction of constraints whose condition is believed, but whose conclusion is not.

The following paragraphs summarize individual contributions to measuring knowledge consistency in detail:

In Chapter 2, building on Petroni et al. (2019)'s work quantifying the amount of knowledge acquired by LMs, we propose two new related tasks to analyze the factual knowledge stored in LMs. The first task, *negation*, involves automatically inserting negation markers into factual queries to create pairs of positive and negative questions. We find that LMs like Elmo and BERT are prone to generate both

factual statements ("Birds can fly") as well as their incorrect negation ("Birds cannot fly"). For some relation types, the overlap between the top-ranked completions for positive and negative queries exceeds 50%.

The second task, *mispriming*, draws inspiration from priming methods in human psychology. In an adversarial setting, we automatically insert misleading distractors into cloze questions, such as "Talk? Birds can [MASK]" which easily mislead LMs into incorrect responses, like filling the masked token with "talk" instead of factually correct fillers like "chirp". We can manually create more natural sounding misprimes, for example, "regent of Antioch" in "Tancred, regent of Antioch, played a role in the conquest of [MASK]" tricks BERT into choosing the filler "Antioch" (instead of "Jerusalem"). Our automatically generated misprimes are less natural, but they allow us to create a large misprime dataset for an initial study. We find that for BERT, factual accuracy drops significantly, depending on the type of relation and misprime, up to 100%.

Chapter 3 focuses on consistency with respect to *paraphrasing*. Again building on Petroni et al. (2019), we develop ParaRel, a high-quality collection of manually created paraphrases of relation templates. Through ParaRel, we show that LMs like BERT, RoBERTa (Liu et al., 2019) and ALBERT (Lan et al., 2019) exhibit poor consistency with respect to paraphrases, propose a finetuning approach to improve consistency with respect to paraphrases, and then experimentally demonstrate its effectiveness. Consistency improves by 5.8 percent points, which in turn also improves factual accuracy by 1.8 points.

Most work studying knowledge representations in LMs only considers English which limits accessibility of NLP based applications. To address this in the context of factual consistency, Chapter 4 extends to a *multilingual* setting. To this end, we automatically translate a set of factual queries into 53 languages and examine multilingual BERT (Devlin et al., 2019) (mBERT) for consistency across languages. We find that querying mBERT for factual knowledge yields inconsistent performance across languages. Notably, mBERT exhibits a language bias; e.g., when asked in French about locations, it tends to predict France or French cities. This language-bias can be mitigated by pooling predictions across languages by selecting the object predicted by the majority of languages. This approach leads to improved performance, even surpassing that of monolingual English BERT models.

Finally, in Chapter 5 and 6, we study inconsistency with respect to sets of factual queries implied by *commonsense reasoning*. In this context, we can distinguish between two notions of consistency: i) consistency with respect to human-defined implications of factual statements and ii) a model's self-consistency.

Chapter 5 studies consistency with respect to human-defined implications. For example, if the LM believes that "a poodle is a dog", we test if it also believes "a poodle has a tail" independent of the model knowing the implicating fact that

"dogs have tails". We collect these commonsense implications from the human-curated commonsense KB ConceptNet (Speer et al., 2017). We identify general concepts of interest, e.g., "mammal" and convert selected KB triples about them to implications. For example, the ConceptNet triple (dog, has a, tail) becomes "X is a dog" →"X has a tail". We then define a set of 85 entities of interest (animals and plants) which fill the placeholder X and query T5 large (Raffel et al., 2020). We find that T5 large answers only 70% of the queries consistently.

In Chapter 6, we study a model's self-consistency by examining its faithfulness to chains of reasoning it generates itself. Here, the model can be factually incorrect but still self-consistent. For instance, it might generate conditions like "birds have wings" and "a poodle is a bird" to justify the hypothesis "poodles have wings". In practice, we observe that these model generated reasoning chains are more fine-grained than the KB-based ones. To study self-consistency we deploy Entailer (Tafjord et al., 2022) which is a T5 based model finetuned to generate sets of candidate explanations based on a given hypothesis. Given a collection of commonsense, multiple-choice Q&A datasets, EntailmentBank (Dalvi et al., 2021), OBQA (Mihaylov et al., 2018), QuaRTz (Tafjord et al., 2019), we ask Entailer to explain why each candidate answer might be true, expressed as a set of sentences that entail the answer. We then add a self-verification step to check that the model itself believes those generations. For example, when we ask the model to explain "giraffes give live birth", the model generates [because] "mammals give live birth" [and] "a giraffe is a mammal". Self-querying then checks if the model actually believes its generations ("Do mammals give live birth?"). The answer ("yes"/"no") assigns a true/false value to each generation. This two step procedure of generating candidate explanations and self-verification is then applied recursively to the generated, supporting sentences. We call the resulting network of model beliefs and their dependencies a *belief graph* that enables us to materializes relevant model beliefs and their conflicts. Based on this graph, we find that T5 answers ∼87% of the queries self-consistently.

Improving Consistency

Beyond quantifying LM's consistency with respect to commonsense reasoning, Chapter 5 also introduces a new style of architecture which we call BeliefBank. This approach integrates an LM into a broader system that includes a 'symbolic executive', an evolving, symbolic memory of beliefs. This memory not only records the LM's answers but can also modify them. We outline two mechanisms to improve belief consistency in the overall system. i) A reasoning component that uses a weighted MaxSAT solver to revise beliefs that significantly conflict with others. This constraint solver has a global view on the memory, and thus can balance all beliefs seen so far with the provided constraints to minimize conflict.

ii) A feedback mechanism that retrieves past beliefs from memory and provides them as context when new queries are issued. This allows the model to prevent local conflicts by reflecting on beliefs immediately relevant to the query. We show that these two mechanisms significantly improve both the accuracy and consistency (by absolute $\sim$15% accuracy and $\sim$25% consistency). We also show how conflict resolution in early batches can be propagated to later test queries and therefore how answer accuracy improves over time.

In Chapter 6, we extend our work on BeliefBank and develop the REFLEX model to study self-consistency (see previous section). Given the *belief graph* and using the same global reasoning component as BeliefBank, we identify and minimize contradictions in the graph. As REFLEX relies on model generated constraints, it is prone to noise which makes conflict resolution while maintaining accuracy more challenging than in the BeliefBank setting. We find that REFLEX significantly improves consistency (by 8%-11% absolute) without harming overall answer accuracy, resulting in answers supported by faithful chains of reasoning drawn from a more consistent belief system.

Overall, this body of work is one of the first to challenge knowledge acquisition in LMs beyond the standard Q&A setup that queries individual facts in isolation and shows how brittle LMs' belief systems are. This brittleness is a major impediment in practical applications of LMs in critical domains such as medicine and law where properties of explainability, interpretability, and trust are a necessity. Our work on BeliefBank systems is a significant first step towards LM based architectures with a systematic notion of belief, enabling them to construct a more coherent picture of the world, and potentially improve over time without any additional training.

1.4 Completeness of Knowledge

Again, our contribution to *knowledge completeness* is twofold: First, we explore mechanisms through which LMs acquire factual knowledge during pretraining. More specifically reasoning mechanisms which could enable an LM to acquire knowledge beyond what it explicitly sees during training. Second, we build systems that are able to integrate knowledge from external natural language sources into the LM.

Study of knowledge acquisition

In Chapter 7, we explore how LMs acquire factual knowledge during pretraining by examining the causal relationship between the facts present in the training data and the facts learned by the model. We focus on two mechanisms: reasoning and

memorization.

Regarding reasoning, to conduct our study, we construct small, synthetic training corpora consisting of knowledge base (subject, relation, object) triplets that specifically target the factors we are interested in. We pretrain language models from scratch that follow BERT architecturally but reduce the number of parameters and layers significantly to account for training data size. To study reasoning, the synthetic corpus follows a set of six symbolic rules (equivalence, symmetry, inversion, composition, implication, negation). For each rule, we create a corpus that contains facts from which the rule can be learned. We test the model's ability to use the rule to infer unseen facts by holding out some facts in a test set.

We find that BERT is capable of learning some one-hop rules (equivalence and implication). For others, even though high test precision suggests successful learning, we observe that its application of these rules at inference time is flawed (symmetry, inversion and negation). We find that BERT struggles with two-hop rules (composition).

In the context of memorization, we aim to identify factors that contribute to the successful memorization of facts encountered during pretraining. Specifically, we examine two such factors: schema conformity and frequency. Schema conformity refers to facts that are consistently supported by other facts, e.g., "sparrows can fly" in a corpus with many similar facts about birds. In contrast, exceptions lack consistent support by other facts e.g., "penguins cannot fly". By frequency we mean the number of times a model is exposed to certain facts over training.

To study memorization, we ensure that the amount of factual knowledge contained in the training corpus surpasses the model's storage ability. We then control fact frequency and schema conformity, and measure their effect on memorization. We identify schema conformity and frequency as key factors for successful memorization whereas exceptions are more difficult to learn.

Prior work has mainly focused on quantifying the amount of knowledge that is acquired but little is understood about *how* this knowledge is acquired during pretraining and why. To the best of our knowledge, our work is the first to propose a synthetic framework to investigate the causal relation between facts present in training and facts learned by the LM, therefore enabling us to study whether LMs have the ability to acquire knowledge via reasoning which in turn would make models also more consistent.

Integrating novel knowledge into Language Models

To maintain stable performance over time, LMs need to keep in sync with our ever changing world. Moreover, we don't have compelling mechanisms which ensure that certain information seen during training is captured parametrically. Therefore, we need mechanisms to integrate new knowledge into LMs. Chapter 8 and

Chapter 9, address this problem in the context of two of the most prominent NLP tasks: Question-Answering and Entity Linking. For both tasks, we extend the LM with an external knowledge store containing embedding-based representations of novel knowledge and a retrieval mechanism to integrate this information into the task specific system.

In Chapter 8, we introduce BERT-kNN where we build on Khandelwal et al. (2020) and extend an LM with a retrieval component over Wikipedia articles. More specifically, we combine BERT with a two step retrieval process. First, using traditional Tf-idf-based information retrieval, we select a subset of Wikipedia articles that are relevant to the query. Second, we run a k-nearest-neighbour (kNN) search between the encoded query and encodings of the selected Wikipedia passages. We show that BERT-kNN outperforms BERT on cloze-style Q&A by large margins (by $\sim$10% points). Moreover, in contrast to BERT, BERT-kNN can handle facts not covered by BERT's training set without any further training and excels for rare facts. We also show that BERT often identifies the correct response category (e.g., US city), but only kNN recovers the factually correct answer (e.g., "New York").

In Chapter 9, we introduce the EDIN, the Unknown Entity Discovery and Indexing benchmark. Standard EL is the task of identifying mentions of named entities in natural language and connecting them to their corresponding entries in a reference knowledge base. This reference KB defines and represents the set of known entities. Usually, work on EL assumes that the reference KB is complete, and therefore all mentions can be linked. In practice this is hardly ever the case, as knowledge bases are incomplete and because novel concepts arise constantly. In EDIN, unknown entities, hence entities without a description in the knowledge base and without labelled mentions that could be used for training, have to be integrated into an existing EL system. Building on dense-retrieval based EL (Cucerzan, 2007), we introduce the end-to-end EDIN pipeline that detects, clusters, and indexes mentions of unknown entities in context by unifying the information of multiple mentions into one embedding per entity and adds them to the reference index. EDIN is able to link unseen entities without any additional training that standard EL systems would not be able to link without relying at least on manually crafted descriptions but its performance sill lacks behind that of seen entities ($\sim$28 percent points less recall).

Given the dynamic nature of the world, devising methods that keep LM-based applications up-to-date is essential. This portion of the thesis contributes to such research work. Concurrent and later work on retrieval-augmented LMs (summarized in Section 1.5), also building on Khandelwal et al. (2020), explores similar systems. Specifically, the work of Lewis et al. (2020) on retrieval augmented generation (RAG) finds wide spread application in industry, highlighting the practical viability of this hybrid approach relying on a combination of parametric and ex-

ternal knowledge.

1.5 Knowledge Representations in Language Models

Shortly after the introduction of pretrained LMs, research finds that their exposure to large amounts of textual information, combined with their training objective of predicting text continuations, indirectly facilitates the acquisition of factual and commonsense knowledge. Most prominently, Petroni et al. (2019) quantify the amount of relational knowledge these models acquire and pose the question whether pretrained LMs could act as alternatives to traditional knowledge bases and Bosselut et al. (2019); Davison et al. (2019) show how pretrained LMs' commonsense abilities can be harvested for KB completion.

These works initiated productive research activity around parametric knowledge representations in the context of LMs.

Advantages of Parametric Knowledge Representations

How to best represent factual knowledge has been a long standing problem in artificial intelligence research (Brachman and Levesque, 2004). Before the wide spread deployment of LMs, systems mainly relied on structured representations like KGs or retrieval over natural language corpora. These technologies are matured enough to constitute the backbone of knowledge systems in industry (Hogan et al., 2021) in the last decades.

Considering the proven utility of these traditional methods, a natural question is whether knowledge represented parametrically within an LM has advantages beyond what is possible with traditional methods.

First and foremost, language models demonstrate remarkable performance far beyond what was possible with prior specialized systems. Noteworthy is their few- and zero-shot performance on standard NLP benchmarks (Brown et al., 2020), and especially their open dialog performance (OpenAI, 2024; Gemini Team, 2023) in the context of closed-book question-answering (Roberts et al., 2020). In a few-shot setting, LMs are presented with only a few task-specific examples without further training to master a downstream task. In a zero-shot setting no task-specific samples are presented, just as in a closed-book Q&A setting where the model answers questions without access to external knowledge sources. In these scenarios, LMs have to rely solely on their parametric knowledge.

There are three aspects why parametric knowledge acquisition within LMs demonstrates great potential (Razniewski et al., 2021):

i) *Unsupervised Construction:* LMs possess the ability to capture knowledge through unsupervised learning. Consequently, they can autonomously acquire knowledge from vast amounts of textual data without the need for explicit human intervention or manual curation.

ii) *Domain Coverage:* LMs excel in covering a wide range of domains. By training on diverse textual sources, including books, websites, and articles, LMs acquire knowledge that spans multiple subject areas. Their expansive domain coverage enables them to provide contextually relevant responses and insights, making them valuable resources for addressing a broad range of queries and tasks.

iii) *Open Schema and Commonsense Knowledge:* Unlike traditional knowledge representations, which rely on structured triplets and predefined relations, LMs offer an open schema. This flexible framework allows LMs to capture and use knowledge beyond the limitations of rigid structured formats. Significantly, LMs hold promise in capturing and leveraging commonsense knowledge, which often eludes rigid formats.

Open Problems of Parametric Knowledge Representations

But with pretrained LM's neural nature also comes the challenge that LM's parametric knowledge is encoded in a diffused manner, making it difficult to access, interpret, control and maintain. As we lack methods to eliminate factual knowledge acquisition in LMs, we have to understand its advantages as well as open problems. Considerable research efforts have been devoted to advancing our understanding of parametric knowledge acquisition and improving its utility. This thesis contributes to such research work.

The following section gives an overview of related work. While it does not encompass the entirety of research conducted in this context, it highlights key research questions and mentions noteworthy papers concurrent to this thesis. Finally, it highlights the most practical approach to knowledge integration into LMs, namely hybrid systems that combine parametric knowledge representations within an LM with external knowledge sources like the BERT-kNN model introduced in Chapter 8.

Knowledge Access:

Petroni et al. (2019) quantify the amount of knowledge LMs acquire via zero-shot, cloze-style question answering relying on manually crafted fill-in the blank templates. However, this approach can only measure a lower bound on the amount of knowledge contained within a pretrained LM, as different templates could

have been more optimal for eliciting knowledge. Jiang et al. (2020) demonstrate this and show how template optimization can result in significantly better performance.

Moreover, zero-shot cloze-style extraction techniques also suffer from the problem that LMs can fill in the blank with natural sounding continuations that are not wrong but do not elicit the knowledge that was targeted, e.g., "Paris is the capital of fashion" instead of "Paris is the capital of France". Finetuning the LM on a dataset of text passages designed to elicit the desired information can address this issue. Two prevalent finetuning approaches in this context include finetuning for Q&A (Roberts et al., 2020) and for KG completion (Bosselut et al., 2019). In Chapters 2,3,4 we follow the cloze-style Q&A approach and in Chapters 5,6 we use finetuned models.

Knowledge Interpretation:

LMs' training objective is to predict the most probable continuation of a given textual input. This objective does not explicitly instruct an LM to refrain from generating responses in situations where they lack the necessary knowledge for accurate factual answers. The absence of such instructions makes LMs prone to attempt to provide a response even if the necessary information to answer a question was missing from the model's training data or was not latently memorized, potentially leading to inaccuracies (Varshney et al., 2022) and inconsistencies as highlighted in Chapters 2 till 6.

In this context, attribution techniques aiming to provide evidence snippets that support the text that an LM generates become important (Bohnet et al., 2023). These techniques enable a more thorough assessment of the credibility and accuracy of the generated knowledge which is instrumental for ensuring reliability of their responses. Attribution techniques are usually applied post-hoc. In our work studying knowledge acquisition during training (Chapter 7), we propose a framework that provides full control over the factual knowledge a model is exposed to enabling us to study (on a small scale) factual knowledge acquisition causally.

Knowledge Control:

The enormous amounts of training data, specifically vast amounts of internet data, coupled with the diffused nature of parametrically stored knowledge, make it hard to exert fine-grained control over the model's responses and to avoid unintended or undesirable outcomes. E.g., LMs have been shown to memorize specific information like personal identifiable information (Carlini et al., 2023) and more generally to amplify harmful societal biases (Blodgett et al., 2020). Research trying to align LMs' generations with human preferences is an active research thread,

e.g., spanning from careful dataset curation (Laurençon et al., 2022) to integration of human feedback during finetuning (Ouyang et al., 2022), to name a few.

Knowledge Maintenance:

Given the dynamic nature of the world, it is essential to ensure that the information stored within an LM remains accurate and relevant over time. Lazaridou et al. (2021) highlight this problem in more depth by showing how a model's performance generally degrades on future text corpora from beyond their training period. Specifically focusing on new events and KG facts, Livska et al. (2022); Dhingra et al. (2022) introduce new benchmarks to measure how LMs can be adapted to new information over time. With our work on EDIN (Chapter 9) we highlight and quantify this problem in the context of Entity Linking.

To keep LMs up-to-date and effective, various methods are explored. One strategy is to continuously train the model on novel data, e.g., Dhingra et al. (2022). Another approach augments LMs with external knowledge sources thereby enabling them to retrieve and integrate novel knowledge (Lewis et al., 2020; Izacard and Grave, 2021; Min et al., 2020). Chapters 8 (BERT-kNN) and 9 (EDIN) of this thesis fall into this category. Retrieval augmentation is discussed further in the next section. A third method, known as knowledge editing (Sinitsin et al., 2020; De Cao et al., 2021; Mitchell et al., 2022; Meng et al., 2022), aims for minimal weight updates to modify specific information within the model but is not a very practical solution to the problem.

Hybrid Systems:

Given the afore discussed challenges of knowledge access, interpretation, control and maintenance, a best-of-both-worlds approach, combining the strengths of LMs and external systems, is promising. Hybrid systems that integrate an LM with external knowledge sources, e.g., a retrieval component over natural text, offer such best-of-both-worlds solutions. Retrieval-augmented LMs usually rely on two steps: i) Retrieving textual passages relevant to the input prompt from a datastore and ii) integrating this information at inference time.

In the past, retrieval commonly used term-match or BM25 (Sparck Jones, 1972; Robertson, 1997). Now, dense-retrieval – relying on a bi-encoder architecture (Yih et al., 2011), that runs a nearest neighbor search between independently encoded input queries and a large-scale index of text passage encodings – is a popular approach. Relevance is computed using the inner product or Euclidean distance between the query and the text passage encodings. To integrate the retrieved information, the answer is usually generated using a sequence-to-sequence model conditioned on the retrieved documents (Lewis et al., 2020; Izacard and Grave,

2021; Min et al., 2020).

In the context of language modeling, Khandelwal et al. (2020) first integrates retrieval over a billions of token datastore into an LM relying directly on the pre-trained language model encoder. Our BERT-kNN model in Chapter 8 directly builds on this work introducing such hybrid system to Q&A and showing how it can integrate novel information not seen during training. RAG (Lewis et al., 2020) then jointly finetunes the retriever and language model by modelling documents as a latent variable, and minimizing the objective with gradient descent. In contrast to our work, RAG doesn't require a two-stage retrieval process that relies on a combination of Tf-idf and dense retrieval. REALM (Guu et al., 2020) extends this from finetuning to jointly training the retrieval system with the LM encoder in an end-to-end fashion. (Izacard et al., 2022) explores different contrastive learning methods to train retrievers to improve transfer to new applications. RETRO (Borgeaud et al., 2022) scales the retrieval datastore to trillions of tokens, and changes the model architecture to take retrieved documents as input. Atlas (Izacard et al., 2022) is a retrieval-augmented LM, designed with a focus on few-shot learning and sample efficiency that exhibits few-shot abilities that emerge at lower scale than in standard LMs.

Instead of dense-retrieval integration, another line of work proposes integration of search engines by generating text queries, and using the retrieved documents as additional context (Nakano et al., 2019; Thoppilan et al., 2022; Shuster et al., 2022; Lazaridou et al., 2022).

1.6 Foundations of Deep Learning for NLP

In this section, we introduce the basic concepts of Deep Learning tailored to NLP, aimed at introducing the basic building blocks underlying Deep Learning based LMs. We refer to Goodfellow et al. (2016) for a more thorough introduction to Deep Learning.

1.6.1 Neural Networks

A neural network is a non-linear function, parameterized by $\Theta \in \mathbb{R}^k$, that maps input $\mathbf{X}$ to output $\mathbf{Y}$ as $f_\Theta : \mathbf{X} \in \mathbb{R}^n \rightarrow \mathbf{Y} \in \mathbb{R}^m$. In this definition, given $\Theta \in \mathbb{R}^k$, f_Θ maps vectors to vectors. We can also extend its definition to input matrices $\mathbf{X} \in \mathbb{R}^{k \times n}$ by applying f_Θ row-wise, resulting in an output matrix $f_\Theta(\mathbf{X}) \in \mathbb{R}^{k \times m}$. Additionally, f_Θ is required to be differentiable with respect to Θ. Neural networks are typically a composition of such functions f_{i,Θ_i}:

$$f_\Theta(x) = f_{l,\Theta_l}(f_{l-1,\Theta_{l-1}}(\dots f_{1,\Theta_1}(x)) \dots)$$

where $l \in \mathbb{N}$ is the number of layers in the network.

In the following, we present specific layers commonly used to build neural LMs.

Encoder Layers

In NLP, the input space consists of sequences of words which have to be mapped to real-valued vectors as input to the neural network. This transformation takes place in the first layer of the neural network, called the embedding layer. First, we define a finite *vocabulary* set $\mathcal{V}$. Each item in the vocabulary space corresponds to a column in a identity matrix I_n where n corresponds to the number of words in the vocabulary. This sparse vector representation of the vocabulary is then mapped to a dense representation via a matrix product with an *embedding matrix* $\mathbf{E} \in \mathbb{R}^{m \times n}$ where each row corresponds to a vector representation, the *embedding*, of an item in the vocabulary and m is referred to as the *embedding dimensionality*.

Typically the set $\mathcal{V}$ corresponds to a set of words or subword tokens (Schuster and Nakajima, 2012). Subword tokens are beneficial because they lower the memory demands of $\mathbf{E}$ and allow the model to handle a wide range of text sequences.

Feed-Forward Layers

A feed-forward layer is a function that consists of a linear transformation followed by a non-linear one that is applied element-wise to an input vector, $\mathbf{x} \in \mathbb{R}^n$:

$$\text{Feed Forward}(\mathbf{x}) = g(\mathbf{W} \cdot \mathbf{x} + \mathbf{b})$$

where $\mathbf{W} \in \mathbb{R}^{m \times n}$ and $\mathbf{b} \in \mathbb{R}^m$ are the layer's parameters and $g : \mathbb{R}^m \to \mathbb{R}^m$ is the non-linear *activation function*. There are a number of different activation function to chose from. A common choice is the *rectified linear unit* (ReLU) $g(\mathbf{x}) = \max(0, \mathbf{x})$.

Softmax Layers

To transform the output of a neural network into a probability distribution, e.g., for classification tasks, the last layer of the neural network typically uses a *softmax function*:

$$\text{Softmax}(\mathbf{x})_i = \frac{e^{\mathbf{x}_i}}{\sum_{j=1}^{n} e_j^{\mathbf{x}}}$$

which is a parameter-free mapping where $\text{Softmax}(\mathbf{x})_i$ is interpreted as the probability that the model assigns to the ith output class.

Layer Normalization

Layer normalization (Ba et al., 2016) is a technique to normalize the inputs to a layer which stabilizes the learning process. With learnable parameters $\mathbf{w}, \mathbf{b} \in \mathbb{R}^n$, it is defined as:

$$\text{Layer Norm}(\mathbf{x}) = \mathbf{w} \odot \frac{\mathbf{x} - \mu(\mathbf{x})}{\sigma(\mathbf{x})} + \mathbf{b}$$

with $\mu(\mathbf{x})$, $\sigma(\mathbf{x})$ being the mean and the standard deviation of input $\mathbf{x}$ and $\odot$ denoting element-wise multiplication.

Recurrent Layers

For a sequential input $\mathbf{X} = \{\mathbf{x}_0, \ldots, \mathbf{x}_t\}$ like sequences of words (e.g., from a sentence), a recurrent layer recursively maps input $\mathbf{x}_t \in \mathbb{R}^n$ to output $\mathbf{y}_t \in \mathbb{R}^m$ by computing

$$\begin{aligned}\mathbf{h}_t &= g_h(\mathbf{U}_h \mathbf{x}_t + \mathbf{V}_h \mathbf{h}_{(t-1)} + \mathbf{b}_h) \\ \mathbf{y}_t &= g_y(\mathbf{U}_y \mathbf{h}_t + \mathbf{b}_y),\end{aligned}$$

where

$$\theta = (\mathbf{U}_h, \mathbf{V}_h \in \mathbb{R}_{n \times n}; \mathbf{b}_h \in \mathbb{R}_d; \mathbf{U}_y \in \mathbb{R}^{m \times n}; \mathbf{b}_y \in \mathbb{R}_m)$$

are the parameters of the model and g are activation functions.

RNNs can effectively capture temporal dependencies and sequential information by maintaining hidden vector representations $\mathbf{h}_t$ that evolve as they process each element of the input sequence. This sequential nature made RNNs suitable for language processing tasks.

Attention Layers

An attention layer can focus on different elements of an input sequence $\mathbf{X} = \{\mathbf{x}_0, \ldots, \mathbf{x}_t\}$, computing weighted sums based on their importance. To this end, we define $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ matrices, as so-called *query*, *key* and *value* representations of $\mathbf{X}$ where each column of the matrix represents a k-dimensional vector representation of $\mathbf{x}_t$. The output of the attention layer is a weighted sum of $\mathbf{V}$, where the weight assigned to each value is determined by the dot-product of the query with all the keys:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}(\mathbf{Q}\mathbf{K}^\top / \sqrt{k}) \cdot \mathbf{V}$$

Attention layers in NLP enable updating word representations based on their context words by *attending* to their representations (Bahdanau et al., 2015) and

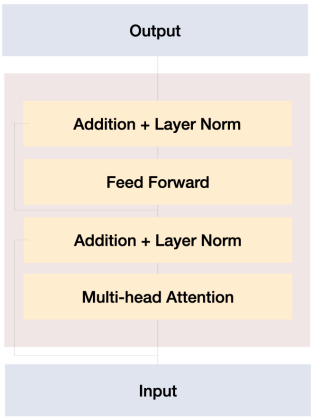


Figure 1.1 – *Transformer encoder block (adapted from Vaswani et al. (2017))*

are one of the core building blocks of the Transformer architecture (Vaswani et al., 2017). The Transformer architecture uses *multi-head self-attention*. *Self-attention* refers to the special case where $\mathbf{Q} = \mathbf{K} = \mathbf{V}$. In *multi-head attention*, attention is performed multiple times with different linear transformations of $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ where the resulting sequences are concatenated, and a final linear transformation is applied.

1.6.2 Transformer Models

In this section, we will assemble the individual layers we introduced in the preceding section to build the Transformer model (Vaswani et al., 2017). The Transformer is the core architecture that all modern LMs have in common.

Before the introduction of the Transformer model, recurrent neural networks (RNNs) that stack multiple recurrent layers, specifically the Long short-term memory (LSTM) network (Hochreiter and Schmidhuber, 1997) were commonly used in NLP.

RNN's sequential nature by definition seemed suitable for language. However, propagating information across long time spans turned out to be empirically difficult, e.g., as analyzed by Cho et al. (2014) for machine translation. Attention layers have been added to overcome this issue (Bahdanau et al., 2015). Vaswani et al. (2017) proposed a machine translation system relying on attention only. The architecture comprises an encoder responsible for transforming input sequences into contextualized vector representations via self-attention, and a decoder generating output sequences token-by-token by attending to both input representations and previously generated output.

Transformer Encoders (see Figure 1.1) are composed of stacked blocks, each consisting of self-attention layers followed by feed-forward layers, incorporating residual connections and layer normalization. These blocks transform input token embeddings into contextualized representations, with positional information added through learned positional embeddings or combinations of sine and cosine functions. The entire encoder comprises multiple such blocks.

Transformer Decoders are comprised of multiple blocks similar to the sequence of encoder layers but employ masked self-attention to prevent tokens from attending to future tokens during left-to-right text generation. Additionally, decoders incorporate attention layers allowing contextualized representations to attend to input representations from the encoder.

To obtain token predictions, the output of the last hidden layer $T(\mathbf{X})$ is once more transformed to obtain $S(\mathbf{X}) = \text{Layer Norm}(\sigma(T(\mathbf{X})\mathbf{W} + \mathbf{b}))$. Then the embeddings matrix $\mathbf{E}$ from the first layer is reused and prediction scores are obtained by $P(\mathbf{X}) = \text{Softmax}(S(\mathbf{X})\mathbf{E}^\intercal + \mathbf{b})$. Here, $P(\mathbf{X})_t$ can be interpreted as the probability that the model assigns to vocabulary tokens at step t.

1.6.3 Training

Initially, the network's parameters Θ are set randomly. We then train the neural network f_Θ to optimize a specified objective function formulated in terms of a *loss function* $\mathcal{L}$. For a specific training example x with label y, we define loss $\mathcal{L}(x, y)$ which we aim to minimize over the set of training examples $\mathcal{D}_{\text{train}}$:

$$\mathcal{L} = \sum_{(x,y)\in\mathcal{D}_{\text{train}}} \mathcal{L}(x, y).$$

A common loss function used in neural networks is the *cross-entropy loss*, which is defined as:

$$\mathcal{L} = -\log f_\Theta(x)_y.$$

This loss function is typical for classification tasks with k classes where the model generates a k-dimensional output vector $f_\Theta(x) \in [0, 1]^k$ for each input x. The final layer is a softmax layer, allowing us to interpret $f_\Theta(x)_i$ as the probability assigned to class i for input x.

In the context of language modeling, y represents the correct token, and $f_\Theta(x)$ is the output of the Transformer model. Here, $f_\Theta(x)_y$ is the probability assigned to the correct token y given the input x as shown in the prior section.

To train neural networks, we typically use *gradient descent*. The gradient of the loss, $\nabla\mathcal{L}$, is computed, and the parameters Θ are updated by taking a small step in the opposite direction of the gradient:

$$\Theta'_i = \Theta_i - \gamma \cdot (\nabla \mathcal{L})_i \,,$$

where $\gamma \in \mathbb{R}$ is the *learning rate*. This process is repeated multiple times until a predefined termination criterion is met.

To train deep neural networks successfully, modifications of this basic method exist.

1.7 Foundations of NLP for LMs

In this section, we introduce the basic concepts of representation learning for natural language and language modeling, and give an overview of popular language models published during the time of this thesis. For a more in-depth introduction to NLP, we refer the reader to Jurafsky and Martin (2009)[4].

Central to representation learning is the hypothesis that a word's meaning can be inferred by the contexts in which a word appears (Firth, 1957). Representation learning algorithms following this hypothesis base a word's representation on the contexts they appear in. Consequently, two words that occur in similar contexts, share similar representations as they are likely to have similar meaning.

1.7.1 Static Representations

In static representations, each unit in a vocabulary V with size n is represented by a single static vector given by $f : V \rightarrow \mathbb{R}^d$ with dimensionality $d \in \mathbb{N}$. Such representations could follow different formalization.

An early formalization of Firth's hypothesis is the idea of count-based approaches where a co-occurrence matrix tracks how often words appear together within a context window. Later neural networks approaches became more prevalent. Most prominently, Mikolov et al. (2013) introduced two methods to estimate word vectors. One method is skip-gram with negative sampling, commonly referred to as *word2vec*. The underlying idea is to predict, given a word v_i, whether another word v_j is likely to appear in the context window of v_i. This prediction is realized by a shallow, two-layer neural network where the dot product between embeddings of words should be large when occurring in the same context and small otherwise.

[4]For an updated version, see https://web.stanford.edu/ jurafsky/slp3/

1.7.2 Contextualized Representations.

With static word embeddings, ambiguous words, e.g., *bank*, that carry multiple meanings (e.g., *a river bank* or *the financial institution*) share a single static representation even though they appear in distinct contexts. Contextualized representations take the context in which a word appears explicitly into account. A word is represented by a vector that is dynamically conditioned on the current context it appears in. Hence the representations for the word *bank* in the above examples differ.

Among the first approaches to learn contextualized embeddings are Peters et al. (2017); McCann et al. (2017). The central idea is to learn a neural language model and use the hidden states of the model as contextualized embeddings.

1.7.3 Language Models

A language model is a statistical model of natural language. It models the probability that a sequence of tokens occurs, i.e., it models the probability $P(t_0, \ldots, t_t)$. Using the product rule of probability, we can decompose the probability of a sequence of tokens into conditional probabilities of each token given the previous context:

$$P(t_0, \ldots, t_t) = P(t_0)P(t_1|t_0)P(t_2|t_0, t_1) \ldots P(t_t|t_0, \ldots, t_{t-1}) = \prod_{i=0}^{t} P(u_i|u_{<i}).$$

A statistical model $P_\theta(t_i|t_0, \ldots, t_{i-1})$ can then be parameterized with, for example, an LSTM, e.g., Sundermeyer et al. (2012).

A significant benefit is that language models can be pretrained on extensive text data, such as internet sources, without requiring manual labeling. As a result, these pretrained models can then be applied to various downstream tasks without the need for additional labeled data.

Elmo: Embeddings from Language Models

Peters et al. (2017) introduced contextualized embeddings by training a language model with forward and backward LSTMs. The backward pass models previous tokens using future context, and the hidden states of the network, combined with static embeddings, serve as contextual input for downstream sequence tagging tasks.

Building on this, Peters et al. (2018) developed deep contextualized embeddings known as *Embeddings from Language Models* (*ELMo*), which at the time achieved state-of-the-art performance on various tasks. ELMo uses bidirectional

LSTMs for language modeling and allows the downstream model to learn a weighted mixture of hidden representations from different layers.

BERT: Bidirectional Encoder Representations from Transformers

Devlin et al. (2019) introduced *Bidirectional Encoder Representations from Transformers* (BERT) which is a Transformer encoder model pretrained on a modified, bi-directional, form of language modeling. BERT considers context words from both left and right sides simultaneously, rather than being limited to one direction. This version of language modeling is termed *masked language modeling* (*MLM*).

In masked language modeling, a random portion of the input tokens in a sentence is corrupted and the model objective is to predict the masked tokens of the original input. In BERT, 15% of the input tokens in a training corpus are sampled for learning. Of these, 80% are replaced with [MASK], 10% are replaced with randomly selected tokens, and the remaining 10% are left unchanged.

To adapt BERT for downstream tasks, we finetuning the model with task specifc data. For sequence labeling tasks, this involves adding a prediction head, which is a feed-forward layer, on top of the pretrained transformer. The output probabilities are computed as $P(\mathbf{X}) = \text{Softmax}(T(\mathbf{X})\mathbf{W} + \mathbf{b})$, where $\mathbf{W} \in \mathbb{R}^{d \times k}$ and $\mathbf{b} \in \mathbb{R}^k$ for k possible labels. The parameters from this task-specific prediction head, along with the parameters from the pretrained Transformer model, are jointly optimized via a loss function appropriate for the downstream task.

1.7.4 Overview of Neural Language Models

ELMo and BERT initiated the development of a large variety of LMs with different sizes, architectures, and training data. In Table 1.1, we list popular transformer-based language models published during the course of this thesis.

On the side of the training objective, we distinguish between masked language models (MLM) (described in the preceding section) and causal language models (CLM). As a MLM randomly masks tokens within an input sequence, it has the advantage of bidirectional context, allowing the model to consider both past and future tokens when making predictions. A CLM is an autoregressive model trained to predict the next token in a sequence given only the previous tokens, meaning they only consider the past and not the future context when generating predictions.

Architecturally, LMs can be differentiated in terms of their encoder/decoder structure. The original Transformer consists of two stacks: the encoder and decoder. The encoder is fed the sequence of input tokens and outputs a sequence of vectors of the same length as the input. Then, the decoder autoregressively predicts the target sequence, token by token, conditioned on the output of the encoder.

To achieve this conditioning, the decoder includes cross-attention layers in each of its blocks, allowing the decoder to also attend to the output of the encoder. The self-attention layers in the decoder utilize a causal masking pattern that prevents the model from attending to future tokens when predicting the output sequence.

BERT originally popularized the encoder-only architecture where only a Transformer encoder layer stack was used. A limitation of this architecture is the constraint that it can only produce the same number of tokens that it was fed as input.

Most recent LMs use a decoder-only architecture. These models can be trained as a traditional language model (i.e., to predict the next token in a sequence). Decoder-only models have no independent means of processing or representing the input sequence and target sequence differently, and conditioning is simply based on past tokens. On the one hand, this means that the representation for any conditioning text is inherently weaker; on the other hand, it yields a simpler architecture that is naturally suited to a standard autoregressive next-step-prediction pretraining objective.

Wang et al. (2022a) gives a comprehensive comparison between architectural and training objective differences.

In this work, we mostly focus on BERT (Devlin et al., 2019) and T5 (Raffel et al., 2020) but also make use of ELMo (Peters et al., 2017) , Transformer-XL (Dai et al., 2019), RoBERTa (Liu et al., 2019), ALBERT (Lan et al., 2019) and multilingual mBERT (Devlin et al., 2019).

Model	Architecture	Data	Objective	Sizes	Chapter
BERT Devlin et al. (2019)	Enc	Wikipedia, Books Corpus	CLM	base (110M), large (336M)	2,3,7,8,9
mBERT Devlin et al. (2019)	Enc	102 languages with largest Wikipedias	MLM, NSP	base (110M)	4
Transformer-XL Dai et al. (2019)	Enc	wikitext-103		large (257M)	2
RoBERTa Liu et al. (2019)	Enc	Wikipedia, Books Corpus, CC-News, OpenWeb-Text, Stories	MLM	base (125M), large (355M)	3
XLM-R Conneau et al. (2020)	Enc	Common Crawl	MLM	base (270M), large (550M)	-
ALBERT Lan et al. (2019)	Enc	Wikipedia, Books Corpus, CC-News, OpenWeb-Text, Stories	MLM, SOP	base (12M), large (18M), xlarge (60M), xxlarge (235M)	3
GPT-2 Radford et al. (2019)	Dec	WebText	CLM	small (117M), medium (345M), large (774M), XL (1.5B)	-
GPT-3 Brown et al. (2020)	Dec	Common Crawl, WebText2, Books1, Books2, Wikipedia	CLM	small (125M), medium (350M), large (760M), XL (1.3B), 2.7B, 6.7B, 13B, 175B	-
T5 Raffel et al. (2020)	Enc-Dec	C4	MLM	small (77M), base (250M), large (800M), XL (3B), XXL (11B)	5,6

Table 1.1 – *Overview of popular transformer-based LMs published during the time of this thesis (adapted from Schick (2023)). For each model, we list the underlying Transformer architecture, the pretraining data and objective, relevant model sizes and the chapters in which it is used.*

1.8 Limitations

Saying that the field of NLP is moving quickly would be an understatement. Its speed of progress is unprecedented. The scope of contributions of a single thesis is naturally limited. This thesis mostly studies early stage LMs, specifically BERT. Developments since November 2022 with the introduction of ChatGPT (OpenAI, 2024) are out of scope of this thesis. Moving forward, future work has to study how findings transfer to larger models where new capabilities could have emerged (Wei et al., 2022a).

On a technical level, the presented work on improving knowledge consistency and completeness embeds the LM into broader systems where improvements happen in this larger system outside of the LM itself. Propagating and generalizing information back into the parameters of the LM is out of scope of this thesis. Similarly, our studies of different consistency patterns highlight the problem at inference time but do not investigate the inner workings of LMs that could enable a model to construct consistent knowledge representations latently.

1.9 Outlook

GPT-3.5 and 4 (OpenAI, 2024) mark another significant breakthrough in the field of NLP, saturating many academic benchmarks and hence also constituting significant progress in the context of the shortcomings of LMs highlighted by this dissertation. These models are larger in size and have been trained on more data. Additionally, an emphasis has been put on aligning the model to human preferences and instructions (Ouyang et al., 2022). Nonetheless, despite these advancements, issues of consistency and completeness in knowledge acquisition are not entirely resolved.

As demonstrated in our transition from studying 770 million parameter models (Kassner et al., 2021) to 11 billion models (Kassner et al., 2023) where we moved to more challenging consistency sets, the nature of the problem appears to have shifted in granularity. While earlier models exhibited obvious and easily identifiable failures, current models tend to display more nuanced, subtle inconsistencies. Recent studies also confirm that inconsistencies, e.g., in the context of negation (Truong et al., 2023) and logical rules (Jang and Lukasiewicz, 2023; Berglund et al., 2024), persist, highlighting the relevance of this thesis also in the context of GPT-3.5/4.

In terms of completeness, we still lack a standard approach on how to integrate novel information. However, highly competitive and active model development ensures that up-to-date versions are frequently available. To enhance knowledge

integration, retrieval-augmentation proves to be a practical solution with RAG-like approaches (Lewis et al., 2020) gaining significant popularity in industry.

While practical solutions offer immediate benefits, our understanding of how LMs acquire and generalize knowledge remains insufficient. Research studying the inner workings of LMs, e.g., Geva et al. (2023); Wang et al. (2023a); Yang et al. (2024), holds promise. By enabling models to generalize knowledge beyond what they have explicitly seen during training and enhancing latent reasoning capabilities during both training and inference, such research holds potential for significant advancements. Ultimately, the goal is to develop models capable of generalizing any factual statement implicitly and integrating it seamlessly into a coherent belief system. For instance, in a multilingual context, given information in English, an ideal model would be able to automatically generalize across languages, enabling queries in any language of interest.

However, relying solely on latent knowledge and reasoning proves insufficient at present. Retrieval-augmented and prompting approaches like chain-of-thought-prompting (Wei et al., 2022b) and self-consistency (Wang et al., 2023b) or instruction following (Wang et al., 2022b) which trigger the LM to reason explicitly prove to be a viable alternative in making models more consistent and able to integrate external information – specifically given most recent introduction of models that can rely on millions of tokens of context (Gemini Team, 2024). Additionally, the development of LM agents (Weng, 2023), which are models embedded in larger systems capable of observing and taking actions in the real world via tools, heralds an exciting new chapter in language model capabilities. These agents leverage their ability to interact with external environments, allowing them to gather real-time information and make informed decisions, further enhancing the applicability and robustness of language models in practical scenarios.

Chapter 2

Negated and Misprimed Probes for Pretrained Language Models: Birds Can Talk, But Cannot Fly

Negated and Misprimed Probes for Pretrained Language Models: Birds Can Talk, But Cannot Fly

Nora Kassner, Hinrich Schütze
Center for Information and Language Processing (CIS)
LMU Munich, Germany
kassner@cis.lmu.de

Abstract

Building on Petroni et al. (2019), we propose two new probing tasks analyzing factual knowledge stored in Pretrained Language Models (PLMs). (1) *Negation.* We find that PLMs do not distinguish between negated ("Birds cannot [MASK]") and non-negated ("Birds can [MASK]") cloze questions. (2) *Mispriming.* Inspired by priming methods in human psychology, we add "misprimes" to cloze questions ("Talk? Birds can [MASK]"). We find that PLMs are easily distracted by misprimes. These results suggest that PLMs still have a long way to go to adequately learn human-like factual knowledge.

1 Introduction

PLMs like Transformer-XL (Dai et al., 2019), ELMo (Peters et al., 2018) and BERT (Devlin et al., 2019) have emerged as universal tools that capture a diverse range of linguistic and factual knowledge. Recently, Petroni et al. (2019) introduced LAMA (LAnguage Model Analysis) to investigate whether PLMs can recall factual knowledge that is part of their training corpus. Since the PLM training objective is to predict masked tokens, question answering (QA) tasks can be reformulated as cloze questions. For example, "Who wrote 'Dubliners'?" is reformulated as "[MASK] wrote 'Dubliners'." In this setup, Petroni et al. (2019) show that PLMs outperform automatically extracted knowledge bases on QA. In this paper, we investigate this capability of PLMs in the context of (1) *negation* and what we call (2) *mispriming*.

(1) Negation. To study the effect of negation on PLMs, we introduce the *negated LAMA dataset*. We insert negation elements (e.g., "not") in LAMA cloze questions (e.g., "The theory of relativity was *not* developed by [MASK].") – this gives us positive/negative pairs of cloze questions.

Querying PLMs with these pairs and comparing the predictions, we find that the predicted fillers have high overlap. **Models are equally prone to generate facts ("Birds can fly") and their incorrect negation ("Birds cannot fly").** We find that BERT handles negation best among PLMs, but it still fails badly on most negated probes. In a second experiment, we show that BERT can in principle memorize both positive and negative facts correctly if they occur in training, but that it poorly generalizes to unseen sentences (positive and negative). However, after finetuning, BERT does learn to correctly classify unseen facts as true/false.

(2) Mispriming. We use priming, a standard experimental method in human psychology (Tulving and Schacter, 1990) where a first stimulus (e.g., "dog") can influence the response to a second stimulus (e.g., "wolf" in response to "name an animal"). **Our novel idea is to use priming for probing PLMs**, specifically *mispriming*: we give automatically generated misprimes to PLMs that would not mislead humans. For example, we add "Talk? Birds can [MASK]" to LAMA where "Talk?" is the misprime. A human would ignore the misprime, stick to what she knows and produce a filler like "fly". We show that, in contrast, PLMs are misled and fill in "talk" for the mask.

We could have manually generated more natural misprimes. For example, misprime "regent of Antioch" in "Tancred, regent of Antioch, played a role in the conquest of [MASK]" tricks BERT into chosing the filler "Antioch" (instead of "Jerusalem"). Our automatic misprimes are less natural, but automatic generation allows us to create a large misprime dataset for this initial study.

Contribution. We show that PLMs' ability to learn factual knowledge is – in contrast to human capabilities – extremely brittle for negated sentences and for sentences preceded by distracting material (i.e., misprimes). Data and code will be

published.[1]

2 Data and Models

LAMA's cloze questions are generated from subject-relation-object triples from knowledge bases (KBs) and question-answer pairs. For KB triples, cloze questions are generated, for each relation, by a templatic statement that contains variables X and Y for subject and object (e.g, "X was born in Y"). We then substitute the subject for X and MASK for Y. In a question-answer pair, we MASK the answer.

LAMA is based on several sources: (i) Google-RE. 3 relations: "place of birth", "date of birth", "place of death". (ii) T-REx (Elsahar et al., 2018). Subset of Wikidata triples. 41 relations. (iii) ConceptNet (Li et al., 2016). 16 commonsense relations. The underlying corpus provides matching statements to query PLMs. (iv) SQuAD (Rajpurkar et al., 2016). Subset of 305 context-insensitive questions, reworded as cloze questions.

We use the source code provided by Petroni et al. (2019) and Wolf et al. (2019) to evaluate Transformer-XL large (Txl), ELMo original (Eb), ELMo 5.5B (E5B), BERT-base (Bb) and BERT-large (Bl).

Negated LAMA. We created negated LAMA by manually inserting a negation element in each template or question. For ConceptNet we only consider an easy-to-negate subset (see appendix).

Misprimed LAMA. We misprime LAMA by inserting an incorrect word and a question mark at the beginning of a statement; e.g., "Talk?" in "Talk? Birds can [MASK]." We only misprime questions that are answered correctly by BERT-large. To make sure the misprime is misleading, we manually remove correct primes for SQuAD and ConceptNet and automatically remove primes that are the correct filler for a different instance of the same relation for T-REx and ConceptNet. We create four versions of misprimed LAMA (A, B, C, D) as described in the caption of Table 3; Table 1 gives examples.

3 Results

Negated LAMA. Table 2 gives spearman rank correlation ρ and % overlap in rank 1 predictions between original and negated LAMA.

Our assumption is that the correct answers for a pair of positive question and negative question should *not* overlap, so high values indicate lack of understanding of negation. The two measures are complementary and yet agree very well. The correlation measure is sensitive in distinguishing cases where negation has a small effect from those where it has a larger effect.[2] % overlap is a measure that is direct and easy to interpret.

In most cases, $\rho > 85\%$; overlap in rank 1 predictions is also high. ConcepNet results are most strongly correlated but TREx 1-1 results are less overlapping. Table 4 gives examples (lines marked "N"). BERT has slightly better results. Google-RE date of birth is an outlier because the pattern "X (not born in [MASK])" rarely occurs in corpora and predictions are often nonsensical.

In summary, PLMs poorly distinguish positive and negative sentences.

We give two examples of the few cases where PLMs make correct predictions, i.e., they solve the cloze task as human subjects would. For "The capital of X is not Y" (TREX, 1-1) top ranked predictions are "listed", "known", "mentioned" (vs. cities for "The capital of X is Y"). This is appropriate since the predicted sentences are more common than sentences like "The capital of X is not Paris". For "X was born in Y", cities are predicted, but

Version	Query
A	Dinosaurs? Munich is located in [MASK] .
B	Somalia? Munich is located in [MASK] .
C	Prussia? Munich is located in [MASK] .
D	Prussia? "This is great". ... "What a surprise." "Good to know." ... Munich is located in [MASK] .

Table 1: Examples for different versions of misprimes: (A) are randomly chosen, (B) are randomly chosen from correct fillers of different instances of the relation, (C) were top-ranked fillers for the original cloze question but have at least a 30% lower prediction probability than the correct object. (D) is like (C) except that 20 short neutral sentences are inserted between misprime and MASK sentence.

[1] https://github.com/norakassner/LAMA_primed_negated

[2] A reviewer observes that spearman correlation is generally high and wonders whether high spearman correlation is really a reliable indicator of negation not changing the answer of the model. As a sanity check, we also randomly sampled, for each query correctly answered by BERT-large (e.g., "Einstein born in [MASK]"), another query with a different answer, but the same template relation (e.g., "Newton born in [MASK]") and computed the spearman correlation between the predictions for the two queries. In general, these positive-positive spearman correlations were significantly lower than those between positive ("Einstein born in [MASK]") and negative ("Einstein not born in [MASK]") queries (t-test, $p < 0.01$). There were two exceptions (not significantly lower): T-REx 1-1 and Google-RE birth-date.

		Facts	Rels	Txl		Eb		E5b		Bb		Bl	
				ρ	%	ρ	%	ρ	%	ρ	%	ρ	%
Google-RE	birth-place	2937	1	92.8	47.1	97.1	28.5	96.0	22.9	89.3	11.2	88.3	20.1
	birth-date	1825	1	87.8	21.9	92.5	1.5	90.7	7.5	70.4	0.1	56.8	0.3
	death-place	765	1	85.8	1.4	94.3	57.8	95.9	80.7	89.8	21.7	87.0	13.2
T-REx	1-1	937	2	89.7	88.7	95.0	28.6	93.0	56.5	71.5	35.7	47.2	22.7
	N-1	20006	23	90.6	46.6	96.2	78.6	96.3	89.4	87.4	52.1	84.8	45.0
	N-M	13096	16	92.4	44.2	95.5	71.1	96.2	80.5	91.9	58.8	88.9	54.2
ConceptNet	-	2996	16	91.1	32.0	96.8	63.5	96.2	53.5	89.9	34.9	88.6	31.3
SQuAD	-	305	-	91.8	46.9	97.1	62.0	96.4	53.1	89.5	42.9	86.5	41.9

Table 2: PLMs do not distinguish positive and negative sentences. Mean spearman rank correlation (ρ) and mean percentage of overlap in first ranked predictions (%) between the original and the negated queries for Transformer-XL large (Txl), ELMo original (Eb), ELMo 5.5B (E5B), BERT-base (Bb) and BERT-large (Bl).

for "X was not born in Y", sometimes countries are predicted. This also seems natural: for the positive sentence, cities are more informative, for the negative, countries.

Balanced corpus. Investigating this further, we train BERT-base from scratch on a synthetic corpus. Hyperparameters are listed in the appendix. The corpus contains as many positive sentences of form "x_j is a_n" as negative sentences of form "x_j is not a_n" where x_j is drawn from a set of 200 subjects $\mathcal{S}$ and a_n from a set of 20 adjectives $\mathcal{A}$. The 20 adjectives form 10 pairs of antonyms (e.g., "good"/"bad"). $\mathcal{S}$ is divided into 10 groups g_m of 20. Finally, there is an underlying KB that defines valid adjectives for groups. For example, assume that g_1 has property a_m = "good". Then for each $x_i \in g_1$, the sentences "x_i is good" and "x_i is not bad" are true. The training set is generated to contain all positive and negative sentences for 70% of the subjects. It also contains either only the positive sentences for the other 30% of subjects (in that case the negative sentences are added to test) or vice versa. Cloze questions are generated in the format "x_j is [MASK]"/"x_j is not [MASK]". We test whether (i) BERT memorizes positive and negative sentences seen during training, (ii) it generalizes to the test set. As an example, a correct generalization would be "x_i is not bad" if "x_i is good" was part of the training set. The question is: does BERT learn, based on the patterns of positive/negative sentences and within-group regularities, to distinguish facts from non-facts.

Table 5 ("pretrained BERT") shows that BERT memorizes positive and negative sentences, but poorly generalizes to the test set for both positive and negative. The learning curves (see appendix) show that this is not due to overfitting the training data. While the training loss rises, the test precision fluctuates around a plateau. However, if we finetune BERT ("finetuned BERT") on the task of classifying sentences as true/false, its test accuracy is 100%. (Recall that false sentences simply correspond to true sentence with a "not" inserted or removed.) So BERT easily learns negation if supervision is available, but fails without it. This experiment demonstrates the difficulty of learning negation through unsupervised pretraining. We suggest that the inability of pretrained BERT to distinguish true from false is a serious impediment to accurately handling factual knowledge.

Corpus	Relation	Facts	A	B	C	D
Google-RE	birth-place	386	11.7	44.7	99.5	98.4
	birth-date	25	72.0	91.7	100.0	88.0
	death-place	88	14.8	47.1	98.9	98.9
T-REx	1-1	661	12.7	20.6	30.1	28.1
	N-1	7034	22.1	48.3	59.9	41.2
	N-M	2774	26.6	55.3	58.7	43.9
ConceptNet	-	146	52.1	59.6	82.9	70.6
SQuAD	-	51	33.3	-	68.6	60.8

Table 3: Absolute precision drop (from 100%, lower better) when mispriming BERT-large for the LAMA subset that was answered correctly in its original form. We insert objects that (A) are randomly chosen, (B) are randomly chosen from correct fillers of different instances of the relation (not done for SQuAD as it is not organized in relations), (C) were top-ranked fillers for the original cloze question but have at least a 30% lower prediction probability than the correct object. (D) investigates the effect of distance, manipulating (C) further by inserting a concatenation of 20 neutral sentences (e.g., "Good to know.", see appendix) between misprime and cloze question.

Misprimed LAMA. Table 3 shows the effect of mispriming on BERT-large for questions answered correctly in original LAMA; recall that Table 1 gives examples of sentences constructed in modes A, B, C and D. In most cases, mispriming with a highly ranked incorrect object causes a precision drop of over 60% (C). Example predictions can be found in Table 4 (lines marked "M"). This sensi-

		cloze question	true	top 3 words generated with log probs
Google RE	O	Marcel Oopa died in the city of [MASK].	Paris	Paris (-2.3), Lausanne (-3.3), Brussels (-3.3)
	N	Marcel Oopa did not die in the city of [MASK].		Paris (-2.4), Helsinki (-3.5), Warsaw (-3.5)
	M	Yokohama? Marcel Oopa died in the city of [MASK].		Yokohama (-1.0), Tokyo (-2.5), Paris (-3.0)
	O	Anatoly Alexine was born in the city of [MASK].	Moscow	Moscow (-1.2), Kiev (-1.6), Odessa (-2.5)
	N	Anatoly Alexine was not born in the city of [MASK].		Moscow (-1.2), Kiev (-1.5), Novgorod (-2.5)
	M	Kiev? Anatoly Alexine was born in the city of [MASK].		Kiev (-0.0), Moscow (-6.1), Vilnius (-7.0)
TERx	O	Platonism is named after [MASK] .	Plato	Plato (-1.5), Aristotle (-3.5), Locke (-5.8)
	N	Platonism is not named after [MASK].		Plato (-0.24), Aristotle (-2.5), Locke (-5.7)
	M	Cicero? Platonism is named after [MASK].		Cicero (-2.3), Plato (-3.5), Aristotle (-5.1)
	O	Lexus is owned by [MASK] .	Toyota	Toyota (-1.4), Renault (-2.0), Nissan (-2.4)
	N	Lexus is not owned by [MASK].		Ferrari (-1.0), Fiat (-1.4), BMW (-3.7)
	M	Microsoft? Lexus is owned by [MASK] .		Microsoft (-1.2), Google (-2.1), Toyota (-2.6)
Concept Net	O	Birds can [MASK].	fly	fly (-0.5), sing (-2.3), talk (-2.8)
	N	Birds cannot [MASK].		fly (-0.3), sing (-3.6), speak (-4.1)
	M	Talk? Birds can [MASK].		talk (-0.2), fly (-2.5), speak (-3.9)
	O	A beagle is a type of [MASK].	dog	dog (-0.1), animal (-3.7), pigeon (-4.1)
	N	A beagle is not a type of [MASK].		dog (-0.2), horse (-3.8), animal (-4.1)
	M	Pigeon? A beagle is a type of [MASK].		dog (-1.3), pigeon (-1.4), bird (-2.2)
SQuAD	O	Quran is a [MASK] text.	religious	religious (-1.0), sacred (-1.8), Muslim (-3.2)
	N	Quran is not a [MASK] text.		religious (-1.1), sacred (-2.3), complete (-3.3)
	M	Secular? Quran is a [MASK] text.		religious (-1.5), banned (-2.8), secular (-3.0)
	O	Isaac's chains are made out of [MASK].	silver	silver (-1.9), gold (-2.1), iron (-2.2)
	N	Isaac's chains are not made out of [MASK].		iron (-1.2), metal (-2.1), gold (-2.1)
	M	Iron? Isaac's chains are made out of [MASK].		iron (-0.4), steel (-2.8), metal (-2.8)

Table 4: BERT-large examples for (O) original , (N) negated and (M) misprimed (Table 3 C) LAMA.

	train		**test**	
	pos	neg	pos	neg
pretrained BERT	0.9	0.9	0.2	0.2
finetuned BERT	1.0	1.0	1.0	1.0

Table 5: Accuracy of BERT on balanced corpus. Pretrained BERT does not model negation well, but finetuned BERT classifies sentences as true/false correctly.

tivity to misprimes still exists when the distance between misprime and cloze question is increased: the drop persists when 20 sentences are inserted (D). Striking are the results for Google-RE where the model recalls almost no facts (C). Table 4 (lines marked "M") shows predicted fillers for these misprimed sentences. BERT is less but still badly affected by misprimes that match selectional restrictions (B). The model is more robust against priming with random words (A): the precision drop is on average more than 35% lower than for (D). We included the baseline (A) as a sanity check for the precision drop measure. These baseline results show that the presence of a misprime per se does not confuse the model; a less distracting misprime (different type of entity or a completely implausible answer) often results in a correct answer by BERT.

4 Discussion

Whereas Petroni et al. (2019)'s results suggest that PLMs are able to memorize facts, our results indicate that PLMs largely do not learn the meaning of negation. They mostly seem to predict fillers based on co-occurrence of subject (e.g., "Quran") and filler ("religious") and to ignore negation.

A key problem is that in the LAMA setup, not answering (i.e., admitting ignorance) is not an option. While the prediction probability generally is somewhat lower in the negated compared to the positive answer, there is no threshold across cloze questions that could be used to distinguish valid positive from invalid negative answers (cf. Table 4).

We suspect that a possible explanation for PLMs' poor performance is that negated sentences occur much less frequently in training corpora. Our synthetic corpus study (Table 5) shows that BERT is able to memorize negative facts that occur in the corpus. However, the PLM objective encourages the model to predict fillers based on similar sentences in the training corpus – and if the most similar statement to a negative sentence is positive, then the filler is generally incorrect. However, after finetuning, BERT is able to classify truth/falseness correctly, demonstrating that negation can be learned through supervised training.

The mispriming experiment shows that BERT often handles random misprimes correctly (Table 3 A). There are also cases where BERT does the right thing for difficult misprimes, e.g., it robustly attributes "religious" to Quran (Table 4). In general, however, BERT is highly sensitive to misleading context (Table 3 C) that would not change human

behavior in QA. It is especially striking that a single word suffices to distract BERT. This may suggest that it is not knowledge that is learned by BERT, but that its performance is mainly based on similarity matching between the current context on the one hand and sentences in its training corpus and/or recent context on the other hand. Poerner et al. (2019) present a similar analysis.

Our work is a new way of analyzing differences between PLMs and human-level natural language understanding. We should aspire to develop PLMs that – like humans – can handle negation and are not easily distracted by misprimes.

5 Related Work

PLMs are top performers for many tasks, including QA (Kwiatkowski et al., 2019; Alberti et al., 2019). PLMs are usually finetuned (Liu et al., 2019; Devlin et al., 2019), but recent work has applied models without finetuning (Radford et al., 2019; Petroni et al., 2019). Bosselut et al. (2019) investigate PLMs' common sense knowledge, but do not consider negation explicitly or priming.

A wide range of literature analyzes linguistic knowledge stored in pretrained embeddings (Jumelet and Hupkes, 2018; Gulordava et al., 2018; Giulianelli et al., 2018; McCoy et al., 2019; Dasgupta et al., 2018; Marvin and Linzen, 2018; Warstadt and Bowman, 2019; Kann et al., 2019). Our work analyzes factual knowledge. McCoy et al. (2019) show that BERT finetuned to perform natural language inference heavily relies on syntactic heuristics, also suggesting that it is not able to adequately acquire common sense.

Warstadt et al. (2019) investigate BERT's understanding of how negative polarity items are licensed. Our work, focusing on factual knowledge stored in negated sentences, is complementary since grammaticality and factuality are mostly orthogonal properties. Kim et al. (2019) investigate understanding of negation particles when PLMs are finetuned. In contrast, our focus is on the interaction of negation and factual knowledge learned in pretraining. Ettinger (2019) defines and applies psycho-linguistic diagnostics for PLMs. Our use of priming is complementary. Their data consists of two sets of 72 and 16 sentences whereas we create 42,867 negated sentences covering a wide range of topics and relations.

Ribeiro et al. (2018) test for comprehension of minimally modified sentences in an adversarial setup while trying to keep the overall semantics the same. In contrast, we investigate large changes of meaning (negation) and context (mispriming). In contrast to adversarial work (e.g., (Wallace et al., 2019)), we do not focus on adversarial examples for a specific task, but on pretrained models' ability to robustly store factual knowledge.

6 Conclusion

Our results suggest that pretrained language models address open domain QA in datasets like LAMA by mechanisms that are more akin to relatively shallow pattern matching than the recall of learned factual knowledge and inference.

Implications for future work on pretrained language models. (i) Both factual knowledge and logic are discrete phenomena in the sense that sentences with similar representations in current pretrained language models differ sharply in factuality and truth value (e.g., "Newton was born in 1641" vs. "Newton was born in 1642"). Further architectural innovations in deep learning seem necessary to deal with such discrete phenomena. (ii) We found that PLMs have difficulty distinguishing "informed" best guesses (based on information extracted from training corpora) from "random" best guesses (made in the absence of any evidence in the training corpora). This implies that better confidence assessment of PLM predictions is needed. (iii) Our premise was that we should emulate human language processing and that therefore tasks that are easy for humans are good tests for NLP models. To the extent this is true, the two phenomena we have investigated in this paper – that PLMs seem to ignore negation in many cases and that they are easily confused by simple distractors – seem to be good vehicles for encouraging the development of PLMs whose performance on NLP tasks is closer to humans.

Acknowledgements. We thank the reviewers for their constructive criticism. This work was funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A and by the European Research Council (Grant No. 740516). The authors of this work take full responsibility for its content.

References

Chris Alberti, Kenton Lee, and Michael Collins. 2019. A BERT baseline for the natural questions. *ArXiv*, abs/1901.08634.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Ishita Dasgupta, Demi Guo, Andreas Stuhlmüller, Samuel J Gershman, and Noah D Goodman. 2018. Evaluating compositionality in sentence embeddings. *arXiv preprint arXiv:1802.04302*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Allyson Ettinger. 2019. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Mario Giulianelli, Jack Harding, Florian Mohnert, Dieuwke Hupkes, and Willem Zuidema. 2018. Under the hood: Using diagnostic classifiers to investigate and improve how language models track agreement information. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 240–248, Brussels, Belgium. Association for Computational Linguistics.

Kristina Gulordava, Piotr Bojanowski, Edouard Grave, Tal Linzen, and Marco Baroni. 2018. Colorless green recurrent networks dream hierarchically. 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 1195–1205, New Orleans, Louisiana. Association for Computational Linguistics.

Jaap Jumelet and Dieuwke Hupkes. 2018. Do language models understand anything? on the ability of LSTMs to understand negative polarity items. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 222–231, Brussels, Belgium. Association for Computational Linguistics.

Katharina Kann, Alex Warstadt, Adina Williams, and Samuel R. Bowman. 2019. Verb argument structure alternations in word and sentence embeddings. In *Proceedings of the Society for Computation in Linguistics (SCiL) 2019*, pages 287–297.

Najoung Kim, Roma Patel, Adam Poliak, Patrick Xia, Alex Wang, Tom McCoy, Ian Tenney, Alexis Ross, Tal Linzen, Benjamin Van Durme, Samuel R. Bowman, and Ellie Pavlick. 2019. Probing what different NLP tasks teach machines about function word comprehension. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics (*SEM 2019)*, pages 235–249, Minneapolis, Minnesota. Association for Computational Linguistics.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. 2016. Commonsense knowledge base completion. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1445–1455, Berlin, Germany. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association*

for Computational Linguistics, pages 3428–3448, Florence, Italy. 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, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. BERT is not a knowledge base (yet): Factual knowledge vs. name-based reasoning in unsupervised qa. *ArXiv*, abs/1911.03681.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865, Melbourne, Australia. Association for Computational Linguistics.

Endel Tulving and Daniel Schacter. 1990. Priming and human memory systems. *Science*, 247(4940):301–306.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing NLP. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2153–2162, Hong Kong, China. Association for Computational Linguistics.

Alex Warstadt and Samuel R. Bowman. 2019. Grammatical analysis of pretrained sentence encoders with acceptability judgments. *ArXiv*, abs/1901.03438.

Alex Warstadt, Yu Cao, Ioana Grosu, Wei Peng, Hagen Blix, Yining Nie, Anna Alsop, Shikha Bordia, Haokun Liu, Alicia Parrish, Sheng-Fu Wang, Jason Phang, Anhad Mohananey, Phu Mon Htut, Paloma Jeretic, and Samuel R. Bowman. 2019. Investigating BERT's knowledge of language: Five analysis methods with NPIs. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2877–2887, Hong Kong, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

A Appendix

A.1 Details on LAMA

We use source code provided by Petroni et al. (2019) [3]. T-REx, parts of ConceptNet and SQuAD allow multiple true answers (N-M). To ensure single true objects for Google-RE, we reformulate the templates asking for location to specifically ask for cities (e.g., "born in [MASK]" to "born in the city of [MASK]"). We do not change any other templates. T-REx still queries for "died in [MASK]".

A.1.1 Details on negated LAMA

For ConceptNet we extract an easy-to-negate subset. The final subset includes 2,996 of the 11,458 samples. We proceed as follows:

1. We only negate sentences of maximal token sequence length 4 or if we find a match with one of the following patterns: "is a type of ", "is made of", "is part of", "are made of.", "can be made of", "are a type of ", "are a part off".

2. The selected subset is automatically negated by a manually created verb negation dictionary.

A.1.2 Details on misprimed LAMA

To investigate the effect of distance between the prime and the cloze question, we insert a concatenation of up to 20 "neutral" sentences. The longest sequence has 89 byte pair encodings. The distance upon the full concatenation of all 20 sentences did not lessen the effect of the prime much. The used sentences are: "This is great.", "This is interesting.", "Hold this thought.", "What a surprise.", "Good to know.", "Pretty awesome stuff.", "Nice seeing you.", "Let's meet again soon.", "This is nice.",

[3] github.com/facebookresearch/LAMA

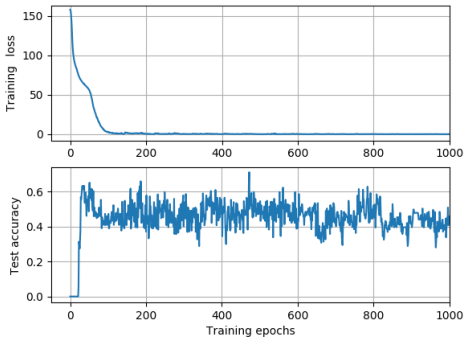


Figure 1: Training loss and test accuracy when pretraining BERT-base on a balanced corpus. The model is able to memorize positive and negative sentences seen during training but is not able to generalize to an unseen test set for both positive and negative sentences.

"Have a nice time.", "That is okay.", "Long time no see.", "What a day.", "Wonderful story.", "That's new to me.", "Very cool.", "Till next time.", "That's enough.", "This is amazing.", "I will think about it."

batch size	512
learning rate	6e-5
number of epochs	2000
max. sequence length	13

Table 6: Hyper-parameters for pretraining BERT-base on a balanced corpus of negative and positive sentences.

batch size	32
learning rate	4e-5
number of epochs	20
max. sequence length	7

Table 7: Hyper-parameters for finetuning on the task of classifying sentences as true/false.

A.2 Details on the balanced corpus

We pretrain BERT-base from scratch on a corpus on equally many negative and positive sentences. We concatenate multiples of the same training data into one training file to compensate for the little amount of data. Hyper-parameters for pretraining are listed in Table 6. The full vocabulary is 349 tokens long.

Figure 1 shows that training loss and test accuracy are uncorrelated. Test accuracy stagnates around 0.5 which is not more than random guessing as for each relation half of the adjectives hold.

We finetune on the task of classifying sentences as true/false. We concatenate multiples of the same training data into one training file to compensate for the little amount of data. Hyperparameters for finetuning are listed in Table 7.

We use source code provided by Wolf et al. (2019) [4].

[4] github.com/huggingface/transformers

Chapter 3

Measuring and Improving Consistency in Pretrained Language Models

Measuring and Improving Consistency in Pretrained Language Models

Yanai Elazar[1,2] **Nora Kassner**[3] **Shauli Ravfogel**[1,2] **Abhilasha Ravichander**[4]
Eduard Hovy[4] **Hinrich Schütze**[3] **Yoav Goldberg**[1,2]
[1]Computer Science Department, Bar Ilan University, Israel
[2]Allen Institute for Artificial Intelligence, United States
[3]Center for Information and Language Processing (CIS), LMU Munich, Germany
[4]Language Technologies Institute, Carnegie Mellon University, United States
{yanaiela,shauli.ravfogel,yoav.goldberg}@gmail.com
kassner@cis.lmu.de {aravicha,hovy}@cs.cmu.edu

Abstract

Consistency of a model—that is, the invariance of its behavior under meaning-preserving alternations in its input—is a highly desirable property in natural language processing. In this paper we study the question: Are Pretrained Language Models (PLMs) consistent with respect to factual knowledge? To this end, we create ParaRel🤘, a high-quality resource of cloze-style query English paraphrases. It contains a total of 328 paraphrases for 38 relations. Using ParaRel🤘, we show that the consistency of all PLMs we experiment with is poor—though with high variance between relations. Our analysis of the representational spaces of PLMs suggests that they have a poor structure and are currently not suitable for representing knowledge robustly. Finally, we propose a method for improving model consistency and experimentally demonstrate its effectiveness.[1]

1 Introduction

Pretrained Language Models (PLMs) are large neural networks that are used in a wide variety of NLP tasks. They operate under a pretrain-finetune paradigm: Models are first *pretrained* over a large text corpus and then *finetuned* on a downstream task. PLMs are thought of as good language encoders, supplying basic language understanding capabilities that can be used with ease for many downstream tasks.

A desirable property of a good language understanding model is *consistency*: the ability to make consistent decisions in semantically equivalent contexts, reflecting a systematic ability to generalize in the face of language variability.

Examples of consistency include: predicting the same answer in question answering and reading comprehension tasks regardless of paraphrase (Asai and Hajishirzi, 2020); making consistent assignments in coreference resolution (Denis and Baldridge, 2009; Chang et al., 2011); or making summaries factually consistent with the original document (Kryscinski et al., 2020). While consistency is important in many tasks, nothing in the training process explicitly targets it. One could hope that the unsupervised training signal from large corpora made available to PLMs such as BERT or RoBERTa (Devlin et al., 2019; Liu et al., 2019) is sufficient to induce consistency and transfer it to downstream tasks. In this paper, we show that this is not the case.

The recent rise of PLMs has sparked a discussion about whether these models can be used as Knowledge Bases (KBs) (Petroni et al., 2019; 2020; Davison et al., 2019; Peters et al., 2019; Jiang et al., 2020; Roberts et al., 2020). Consistency is a key property of KBs and is particularly important for automatically constructed KBs. One of the biggest appeals of using a PLM as a KB is that we can query it in natural language—instead of relying on a specific KB schema. The expectation is that PLMs abstract away from language and map queries in natural language into meaningful representations such that queries with identical intent but different language forms yield the same answer. For example, the query ''*Homeland* premiered on *[MASK]*'' should produce the same answer as ''*Homeland* originally aired on *[MASK]*''. Studying inconsistencies of PLM-KBs can also teach us about the organization of knowledge in the model, or lack thereof. Finally, failure to behave consistently may point to other representational issues such as the similarity between

[1]The code and resource are available at: https://github.com/yanaiela/pararel.

antonyms and synonyms (Nguyen et al., 2016), and overestimating events and actions (reporting bias) (Shwartz and Choi, 2020).

In this work, we study the consistency of factual knowledge in PLMs, specifically in Masked Language Models (MLMs)—these are PLMs trained with the MLM objective (Devlin et al., 2019; Liu et al., 2019), as opposed to other strategies such as standard language modeling (Radford et al., 2019) or text-to-text (Raffel et al., 2020). We ask: Is the factual information we extract from PLMs invariant to paraphrasing? We use zero-shot evaluation since we want to inspect models directly, without adding biases through finetuning. This allows us to assess how much consistency was acquired during pretraining and to compare the consistency of different models. Overall, we find that the consistency of the PLMs we consider is poor, although there is a high variance between relations.

We introduce PARAREL🤘, a new benchmark that enables us to measure consistency in PLMs (§3), by using factual knowledge that was found to be partially encoded in them (Petroni et al., 2019; Jiang et al., 2020). PARAREL🤘 is a manually curated resource that provides patterns—short textual prompts—that are paraphrases of one another, with 328 paraphrases describing 38 binary relations such as *X born-in Y*, *X works-for Y* (§4). We then test multiple PLMs for knowledge consistency, namely, whether a model predicts the same answer for all patterns of a relation. Figure 1 shows an overview of our approach. Using PARAREL🤘, we probe for consistency in four PLM types: BERT, BERT-whole-word-masking, RoBERTa, and ALBERT (§5). Our experiments with PARAREL🤘 show that current models have poor consistency, although with high variance between relations (§6).

Finally, we propose a method that improves model consistency by introducing a novel consistency loss (§8). We demonstrate that, trained with this loss, BERT achieves better consistency performance on unseen relations. However, more work is required to achieve fully consistent models.

2 Background

There has been significant interest in analyzing how well PLMs (Rogers et al., 2020) perform

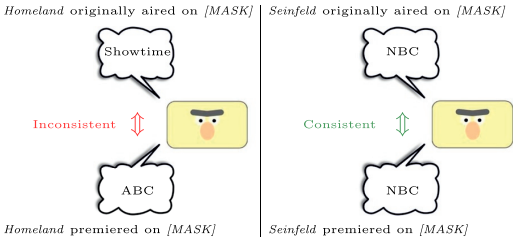


Figure 1: Overview of our approach. We expect that a consistent model would predict the same answer for two paraphrases. In this example, the model is inconsistent on the *Homeland* and consistent on the *Seinfeld* paraphrases.

on linguistic tasks (Goldberg, 2019; Hewitt and Manning, 2019; Tenney et al., 2019; Elazar et al., 2021), commonsense (Forbes et al., 2019; Da and Kasai, 2019; Zhang et al., 2020), and reasoning (Talmor et al., 2020; Kassner et al., 2020), usually assessed by measures of accuracy. However, accuracy is just one measure of PLM performance (Linzen, 2020). It is equally important that PLMs do not make contradictory predictions (cf. Figure 1), a type of error that humans rarely make. There has been relatively little research attention devoted to this question, that is, to analyze if models behave *consistently*. One example concerns negation: Ettinger (2020) and Kassner and Schütze (2020) show that models tend to generate facts and their negation, a type of inconsistent behavior. Ravichander et al. (2020) propose paired probes for evaluating consistency. Our work is broader in scope, examining the consistency of PLM behavior across a range of factual knowledge types and investigating how models can be made to behave more consistently.

Consistency has also been highlighted as a desirable property in automatically constructed KBs and downstream NLP tasks. We now briefly review work along these lines.

Consistency in knowledge bases (KBs) has been studied in theoretical frameworks in the context of the satisfiability problem and KB construction, and efficient algorithms for detecting inconsistencies in KBs have been proposed (Hansen and Jaumard, 2000; Andersen and Pretolani, 2001). Other work aims to quantify the degree to which KBs are inconsistent and detects inconsistent statements (Thimm, 2009, 2013; Muiño, 2011).

Consistency in question answering was studied by Ribeiro et al. (2019) in two tasks: visual question answering (Antol et al., 2015) and reading comprehension (Rajpurkar et al., 2016). They automatically generate questions to test the consistency of QA models. Their findings suggest that most models are not consistent in their predictions. In addition, they use data augmentation to create more robust models. Alberti et al. (2019) generate new questions conditioned on context and answer from a labeled dataset and by filtering answers that do not provide a consistent result with the original answer. They show that pretraining on these synthetic data improves QA results. Asai and Hajishirzi (2020) use data augmentation that complements questions with symmetricity and transitivity, as well as a regularizing loss that penalizes inconsistent predictions. Kassner et al. (2021b) propose a method to improve accuracy and consistency of QA models by augmenting a PLM with an evolving memory that records PLM answers and resolves inconsistency between answers.

Work on **consistency in other domains** includes Du et al. (2019) where prediction of consistency in procedural text is improved. Ribeiro et al. (2020) use consistency for more robust evaluation. Li et al. (2019) measure and mitigate inconsistency in natural language inference (NLI), and finally, Camburu et al. (2020) propose a method for measuring inconsistencies in NLI explanations (Camburu et al., 2018).

3 Probing PLMs for Consistency

In this section, we formally define consistency and describe our framework for probing consistency of PLMs.

3.1 Consistency

We define a model as *consistent* if, given two *cloze-phrases* such as ''*Seinfeld* originally aired on *[MASK]*'' and ''*Seinfeld* premiered on *[MASK]*'' that are *quasi-paraphrases*, it makes non-contradictory predictions[2] on N-1 relations over a large set of entities. A *quasi-paraphrase*—a concept introduced by Bhagat and Hovy (2013)—is a more fuzzy version of a paraphrase. The concept does not rely on the strict, logical definition of paraphrase and allows us to operationalize concrete uses of paraphrases. This definition is in the spirit of the RTE definition (Dagan et al., 2005), which similarly supports a more flexible use of the notion of entailment. For instance, a model that predicts *NBC* and *ABC* on the two aforementioned patterns, is not consistent, since these two facts are contradictory. We define a *cloze-pattern* as a cloze-phrase that expresses a relation between a subject and an object. Note that consistency does not require the answers to be factually correct. While correctness is also an important property for KBs, we view it as a separate objective and measure it independently. We use the terms *paraphrase* and *quasi-paraphrase* interchangeably.

Many-to-many (N-M) relations (e.g., *shares-border-with*) can be consistent even with different answers (given they are correct). For instance, two patterns that express the *shares-border-with* relation and predict *Albania* and *Bulgaria* for *Greece* are both correct. We do not consider such relations for measuring consistency. However, another requirement from a KB is *determinism*, that is, returning the results in the same order (when more than a single result exists). In this work, we focus on consistency, but also measure determinism of the models we inspect.

3.2 The Framework

An illustration of the framework is presented in Figure 2. Let D_i be a set of subject-object KB tuples (e.g., <*Homeland*, *Showtime*>) from some relation r_i (e.g., *originally-aired-on*), accompanied with a set of *quasi-paraphrases* cloze-patterns P_i (e.g., *X* originally aired on *Y*). Our goal is to test whether the model consistently predicts the same object (e.g., *Showtime*) for a particular subject (e.g., *Homeland*).[3] To this end, we substitute *X* with a subject from D_i and *Y* with *[MASK]* in all of the patterns P_i of that relation (e.g., *Homeland* originally aired on *[MASK]* and *Homeland* premiered on *[MASK]*). A consistent model must predict the same entity.

[2] We refer to *non-contradictory predictions* as predictions that, as the name suggest, do not contradict one another. For instance, predicting as the birth place of a person two difference cities is considered to be contradictory, but predicting a city and its country, is **not**.

[3] Although it is possible to also predict the subject from the object, in the cases of N-1 relations more than a single answer would be possible, thus converting the test from measuring consistency to measuring determinism instead.

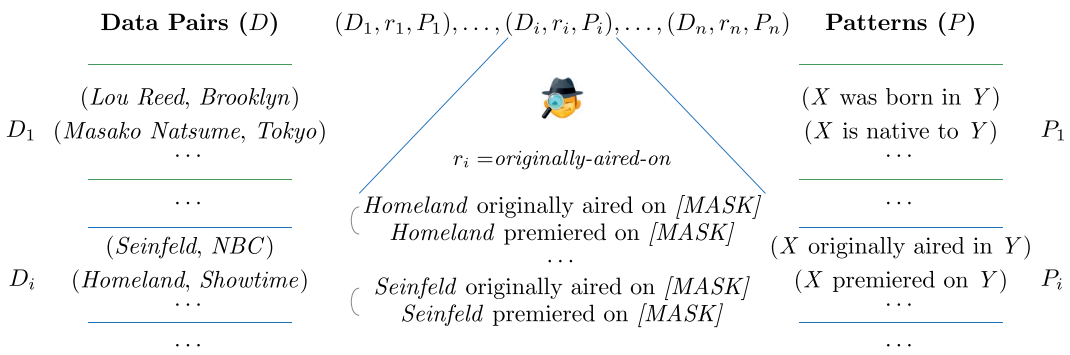


Figure 2: Overview of our framework for assessing model consistency. D_i (''Data Pairs (D)'' on the left) is a set of KB triplets of some relation r_i, which are coupled with a set of *quasi-paraphrase* cloze-patterns P_i (''Patterns (P)'' on the right) that describe that relation. We then populate the subjects from D_i as well as a mask token into all patterns P_i (shown in the middle) and expect a model to predict the same object across all pattern pairs.

Restricted Candidate Sets Since PLMs were not trained for serving as KBs, they often predict words that are not KB entities; for example, a PLM may predict, for the pattern ''*Showtime* originally aired on *[MASK]*'', the noun 'tv'—which is also a likely substitution for the language modeling objective, but not a valid KB fact completion. Therefore, following others (Xiong et al., 2020; Ravichander et al., 2020; Kassner et al., 2021a), we restrict the PLMs' output vocabulary to the set of possible gold objects for each relation from the underlining KB. For example, in the *born-in* relation, instead of inspecting the entire vocabulary of a model, we only keep objects from the KB, such as *Paris*, *London*, *Tokyo*, and so forth.

Note that this setup makes the task easier for the PLM, especially in the context of KBs. However, poor consistency in this setup strongly implies that consistency would be even lower without restricting candidates.

4 The ParaRel🤘 Resource

We now describe ParaRel🤘, a resource designed for our framework (cf. Section 3.2). ParaRel🤘 is curated by experts, with a high level of agreement. It contains patterns for 38 relations[4] from T-REx (Elsahar et al., 2018)—a large dataset containing KB triples aligned with Wikipedia abstracts—with an average of 8.63 patterns per relation. Table 1 gives statistics. We further analyze the paraphrases used in this resource, partly based on the types defined in Bhagat and Hovy (2013), and report this analysis in Appendix B.

Construction Method ParaRel🤘 was constructed in four steps. (1) We began with the patterns provided by LAMA (Petroni et al., 2019) (one pattern per relation, referred to as *base-pattern*). (2) We augmented each base-pattern with other patterns that are paraphrases from LPAQA (Jiang et al., 2020). However, since LPAQA was created automatically (either by back-translation or by extracting patterns from sentences that contain both subject and object), some LPAQA patterns are not correct paraphrases. We therefore only include the subset of correct paraphrases. (3) Using SPIKE (Shlain et al., 2020),[5] a search engine over Wikipedia sentences that supports syntax-based queries, we searched for additional patterns that appeared in Wikipedia and added them to ParaRel🤘. Specifically, we searched for Wikipedia sentences containing a subject-object tuple from T-REx and then manually extracted patterns from the sentences. (4) Lastly, we added additional paraphrases of the base-pattern using the annotators' linguistic expertise. Two additional experts went over all the patterns and corrected them, while engaging in a discussion until reaching agreement, discarding patterns they could not agree on.

Human Agreement To assess the quality of ParaRel🤘, we run a human annotation study. For

[4] Using the 41 relations from LAMA (Petroni et al., 2019), leaving out three relations that are poorly defined, or consist of mixed and heterogeneous entities.

[5] https://spike.apps.allenai.org/.

# Relations	38
# Patterns	328
Min # patterns per rel.	2
Max # patterns per rel.	20
Avg # patterns per rel.	8.63
Avg syntax	4.74
Avg lexical	6.03

Table 1: Statistics of PARAREL🤘. Last two rows: average number of unique syntactic/ lexical variations of patterns for a relation.

each relation, we sample up to five paraphrases, comparing each of the new patterns to the base-pattern from LAMA. That is, if relation r_i contains the following patterns: p_1, p_2, p_3, p_4, and p_1 is the base-pattern, then we compare the following pairs (p_1, p_2), (p_1, p_3), (p_1, p_4).

We populate the patterns with random subjects and objects pairs from T-REx (Elsahar et al., 2018) and ask annotators if these sentences are paraphrases. We also sample patterns from different relations to provide examples that are not paraphrases of each other, as a control. Each task contains five patterns that are thought to be paraphrases and two that are not.[6] Overall, we collect annotations for 156 paraphrase candidates and 61 controls.

We asked NLP graduate students to annotate the pairs and collected one answer per pair.[7] The agreement scores for the paraphrases and the controls are 95.5% and 98.3%, respectively, which is high and indicates PARAREL🤘's high quality. We also inspected the disagreements and fixed many additional problems to further improve quality.

5 Experimental Setup

5.1 Models and Data

We experiment with four PLMs: BERT, BERT whole-word-masking[8] (Devlin et al., 2019), RoBERTa (Liu et al., 2019), and ALBERT (Lan et al., 2019). For BERT, RoBERTa, and ALBERT, we use a base and a large version.[9] We also report a majority baseline that always predicts the most common object for a relation. By construction, this baseline is perfectly consistent.

We use knowledge graph data from T-REx (Elsahar et al., 2018).[10] To make the results comparable across models, we remove objects that are not represented as a single token in all models' vocabularies; 26,813 tuples remain.[11] We further split the data into N-M relations for which we report determinism results (seven relations) and N-1 relations for which we report consistency (31 relations).

5.2 Evaluation

Our consistency measure for a relation r_i (*Consistency*) is the percentage of consistent predictions of all the pattern pairs $p_k^i, p_l^i \in P_i$ of that relation, for all its KB tuples $d_j^i \in D_i$. Thus, for each KB tuple from a relation r_i that contains n patterns, we consider predictions for $n(n-1)/2$ pairs.

We also report *Accuracy*, that is, the acc@1 of a model in predicting the correct object, using the original patterns from Petroni et al. (2019). In contrast to Petroni et al. (2019), we define it as the accuracy of the top-ranked object from the candidate set of each relation. Finally, we report *Consistent-Acc*, a new measure that evaluates individual objects as correct only if *all* patterns of the corresponding relation predict the object correctly. *Consistent-Acc* is much stricter and combines the requirements of both consistency (*Consistency*) and factual correctness (*Accuracy*). We report the average over relations (i.e., macro average), but notice that the micro average produces similar results.

6 Experiments and Results

6.1 Knowledge Extraction through Different Patterns

We begin by assessing our patterns as well as the degree to which they extract the correct entities. These results are summarized in Table 2.

[6] The controls contain the same subjects and objects so that only the pattern (not its arguments) can be used to solve the task.

[7] We asked the annotators to re-annotate any mismatch with our initial label, to allow them to fix random mistakes.

[8] BERT whole-word-masking is BERT's version where words that are tokenized into multiple tokens are masked together.

[9] For ALBERT we use the smallest and largest versions.

[10] We discard three poorly defined relations from T-REx.

[11] In a few cases, we filter entities from certain relations that contain multiple fine-grained relations to make our patterns compatible with the data. For instance, most of the instances for the *genre* relation describes music genres, thus we remove some of the tuples were the objects include non-music genres such as 'satire', 'sitcom', and 'thriller'.

Model	Succ-Patt	Succ-Objs	Unk-Const	Know-Const
majority	97.3±7.3	23.2±21.0	100.0±0.0	100.0±0.0
BERT-base	**100.0**±0.0	63.0±19.9	46.5±21.7	63.8±24.5
BERT-large	**100.0**±0.0	**65.7**±22.1	48.1±20.2	65.2±23.8
BERT-large-wwm	**100.0**±0.0	64.9±20.3	**49.5**±20.1	**65.3**±25.1
RoBERTa-base	**100.0**±0.0	56.2±22.7	43.9±15.8	56.3±19.0
RoBERTa-large	**100.0**±0.0	60.1±22.3	46.8±18.0	60.5±21.1
ALBERT-base	**100.0**±0.0	45.8±23.7	41.4±17.3	56.3±22.0
ALBERT-xxlarge	**100.0**±0.0	58.8±23.8	40.5±16.4	57.5±23.8

Table 2: Extractability measures in the different models we inspect. Best model for each measure highlighted in bold.

Model	Accuracy	Consistency	Consistent-Acc
majority	23.1±21.0	100.0±0.0	23.1±21.0
BERT-base	45.8±25.6	58.5±24.2	27.0±23.8
BERT-large	48.1±26.1	**61.1**±23.0	**29.5**±26.6
BERT-large-wwm	**48.7**±25.0	60.9±24.2	29.3±26.9
RoBERTa-base	39.0±22.8	52.1±17.8	16.4±16.4
RoBERTa-large	43.2±24.7	56.3±20.4	22.5±21.1
ALBERT-base	29.8±22.8	49.8±20.1	16.7±20.3
ALBERT-xxlarge	41.7±24.9	52.1±22.4	23.8±24.8

Table 3: Knowledge and consistency results. Best model for each measure in bold.

First, we report *Succ-Patt*, the percentage of patterns that successfully predicted the right object at least once. A high score suggests that the patterns are of high quality and enable the models to extract the correct answers. All PLMs achieve a perfect score. Next, we report *Succ-Objs*, the percentage of entities that were predicted correctly by at least one of the patterns. *Succ-Objs* quantifies the degree to which the models ''have'' the required knowledge. We observe that some tuples are not predicted correctly by any of our patterns: The scores vary between 45.8% for ALBERT-base and 65.7% for BERT-large. With an average number of 8.63 patterns per relation, there are multiple ways to extract the knowledge, we thus interpret these results as evidence that a large part of T-REx knowledge is not stored in these models.

Finally, we measure *Unk-Const*, a consistency measure for the subset of tuples for which no pattern predicted the correct answer; and *Know-Const*, consistency for the subset where at least one of the patterns for a specific relation predicted the correct answer. This split into subsets is based on *Succ-Objs*. Overall, the results indicate that when the factual knowledge is successfully extracted, the model is also more consistent. For instance, for BERT-large, *Know-Const* is 65.2% and *Unk-Const* is 48.1%.

6.2 Consistency and Knowledge

In this section, we report the overall knowledge measure that was used in Petroni et al. (2019) (*Accuracy*), the consistency measure (*Consistency*), and *Consistent- Acc*, which combines knowledge and consistency (*Consistent-Acc*). The results are summarized in Table 3.

We begin with the *Accuracy* results. The results range between 29.8% (ALBERT-base) and 48.7% (BERT-large whole-word-masking). Notice that our numbers differ from Petroni et al. (2019) as we use a candidate set (§3) and only consider KB triples whose object is a single token in all the PLMs we consider (§5.1).

Next, we report *Consistency* (§5.2). The BERT models achieve the highest scores. There is a consistent improvement from base to large versions of each model. In contrast to previous work that observed quantitative and qualitative improvements of RoBERTa-based models over BERT (Liu et al., 2019; Talmor et al., 2020), in terms of consistency, BERT is more consistent than RoBERTa and ALBERT. Still, the overall results are low (61.1% for the best model), even more remarkably so because the restricted candidate set makes the task easier. We note that the results are highly variant between models (performance on *original-language* varies between 52% and 90%), and relations (BERT-large performance is 92% on *capital-of* and 44% on *owned-by*).

Finally, we report *Consistent-Acc*: the results are much lower than for *Accuracy*, as expected, but follow similar trends: RoBERTa-base performs worse (16.4%) and BERT-large best (29.5%).

Interestingly, we find strong correlations between Accuracy and Consistency, ranging from 67.3% for RoBERTa-base to 82.1% for BERT-large (all with small p-values $\ll 0.01$).

A striking result of the model comparison is the clear superiority of BERT, both in knowledge accuracy (which was also observed by Shin et al., 2020) and knowledge consistency. We hypothesize this result is caused by the different sources of training data: although Wikipedia is part of the training data for all models we consider, for BERT it is the main data source, but for RoBERTa and ALBERT it is only a small portion. Thus, when using additional data, some of the facts may be

Model	Acc	Consistency	Consistent-Acc
majority	23.1±21.0	100.0±0.0	23.1±21.0
RoBERTa-med-small-1M	11.2±9.4	37.1±11.0	2.8±4.0
RoBERTa-base-10M	17.3±15.8	29.8±12.7	3.2±5.1
RoBERTa-base-100M	22.1±17.1	31.5±13.0	3.7±5.3
RoBERTa-base-1B	**38.0**±23.4	**50.6**±19.8	**18.0**±16.0

Table 4: Knowledge and consistency results for different RoBERTas, trained on increasing amounts of data. Best model for each measure in bold.

forgotten, or contradicted in the other corpora; this can diminish knowledge and compromise consistency behavior. Thus, since Wikipedia is likely the largest unified source of factual knowledge that exists in unstructured data, giving it prominence in pretraining makes it more likely that the model will incorporate Wikipedia's factual knowledge well. These results may have a broader impact on models to come: Training bigger models with more data (such as GPT-3 [Brown et al., 2020]) is not always beneficial.

Determinism We also measure determinism for N-M relations—that is, we use the same measure as *Consistency*, but since difference predictions may be factually correct, these do not necessarily convey consistency violations, but indicate non-determinism. For brevity, we do not present all results, but the trend is similar to the consistency result (although not comparable, as the relations are different): 52.9% and 44.6% for BERT-large and RoBERTa-base, respectively.

Effect of Pretraining Corpus Size Next, we study the question of whether the number of tokens used during pretraining contributes to consistency. We use the pretrained RoBERTa models from Warstadt et al. (2020) and repeat the experiments on four additional models. These are RoBERTa-based models, trained on a sample of Wikipedia and the book corpus, with varying training sizes and parameters. We use one of the three published models for each configuration and report the average accuracy over the relations for each model in Table 4. Overall, *Accuracy* and *Consistent-Acc* improve with more training data. However, there is an interesting outlier to this trend: The model that was trained on one million tokens is more consistent than the models trained on ten and one-hundred million tokens. A potentially crucial difference is that this model has many fewer parameters than the rest (to avoid overfitting). It is nonetheless interesting that a model that is trained on significantly less data can achieve better consistency. On the other hand, its accuracy scores are lower, arguably due to the model being exposed to less factual knowledge during pretraining.

Model	Diff-Syntax	No-Change
majority	100.0±0.0	100.0±0.0
BERT-base	67.9±30.3	76.3±22.6
BERT-large	67.5±30.2	78.7±14.7
BERT-large-wwm	63.0±31.7	**81.1**±9.7
RoBERTa-base	66.9±10.1	80.7±5.2
RoBERTa-large	**69.7**±19.2	80.3±6.8
ALBERT-base	62.3±22.8	72.6±11.5
ALBERT-xxlarge	51.7±26.0	67.3±17.1

Table 5: Consistency and standard deviation when only syntax differs (*Diff-Syntax*) and when syntax and lexical choice are identical (*No-Change*). Best model for each metric is highlighted in bold.

6.3 Do PLMs Generalize Over Syntactic Configurations?

Many papers have found neural models (especially PLMs) to naturally encode syntax (Linzen et al., 2016; Belinkov et al., 2017; Marvin and Linzen, 2018; Belinkov and Glass, 2019; Goldberg, 2019; Hewitt and Manning, 2019). Does this mean that PLMs have successfully abstracted knowledge and can comprehend and produce it regardless of syntactic variation? We consider two scenarios. (1) Two patterns differ only in syntax. (2) Both syntax and lexical choice are the same. As a proxy, we define syntactic equivalence when the dependency path between the subject and object are identical. We parse all patterns from ParaRel🤘 using a dependency parser (Honnibal et al., 2020)[12] and retain the path between the entities. Success on (1) indicates that the model's knowledge processing is robust to syntactic variation. Success on (2) indicates that the model's knowledge processing is robust to variation in word order and tense.

Table 5 reports the results. While these and the main results on the entire dataset are not comparable as the pattern subsets are different, they are higher than the general results: 67.5% for BERT-large when only the syntax differs and 78.7% when

[12] https://spacy.io/.

#	Subject	Object	Pattern #1	Pattern #2	Pattern #3	Pred #1	Pred #2	Pred #3
1	Adriaan Pauw	Amsterdam	[X] was born in [Y].	[X] is native to [Y].	[X] is a [Y]-born person.	Amsterdam	Madagascar	Luxembourg
2	Nissan Livina Geniss	Nissan	[X] is produced by [Y].	[X] is created by [Y].	[X], created by [Y].	Nissan	Renault	Renault
3	Albania	Serbia	[X] shares border with [Y].	[Y] borders with [X].	[Y] shares the border with [X]	Greece	Turkey	Kosovo
4	iCloud	Apple	[X] is developed by [Y].	[X], created by [Y].	[X] was created by [Y]	Microsoft	Google	Sony
5	Yahoo! Messenger	Yahoo	[X], a product created by [Y]	[X], a product developed by [Y]	[Y], that developed [X]	Microsoft	Microsoft	Microsoft
6	Wales	Cardiff	The capital of [X] is [Y] .	[X]'s capital, [Y].	[X]'s capital city, [Y].	Cardiff	Cardiff	Cardiff

Table 6: Predictions of BERT-large-cased. ''Subject'' and ''Object'' are from T-REx (Elsahar et al., 2018). ''Pattern #i'' / ''Pred #i'': three different patterns from our resource and their predictions. The predictions are colored in blue if the model predicted correctly (out of the candidate list), and in red otherwise. If there is more than a single erroneous prediction, it is colored by a different red.

the syntax is identical. This demonstrates that while PLMs have impressive syntactic abilities, they struggle to extract factual knowledge in the face of tense, word-order, and syntactic variation.

McCoy et al. (2019) show that supervised models trained on MNLI (Williams et al., 2018), an NLI dataset (Bowman et al., 2015), use superficial syntactic heuristics rather than more generalizable properties of the data. Our results indicate that PLMs have problems along the same lines: They are not robust to surface variation.

7 Analysis

7.1 Qualitative Analysis

To better understand the factors affecting consistent predictions, we inspect the predictions of BERT-large on the patterns shown in Table 6. We highlight several cases: The predictions in Example #1 are inconsistent, and correct for the first pattern (*Amsterdam*), but not for the other two (*Madagascar* and *Luxembourg*). The predictions in Example #2 also show a single pattern that predicted the right object; however, the two other patterns, which are lexically similar, predicted the same, wrong answer—*Renault*. Next, the patterns of Example #3 produced two factually correct answers out of three (*Greece*, *Kosovo*), but simply do not correspond to the gold object in T-REx (*Albania*), since this is an M-N relation. Note that this relation is not part of the consistency evaluation, but the determinism evaluation. The three different predictions in example #4 are all incorrect. Finally, the two last predictions demonstrate consistent predictions: Example #5 is consistent but factually incorrect (even though the correct answer is a substring of the subject), and finally, Example #6 is consistent and factual.

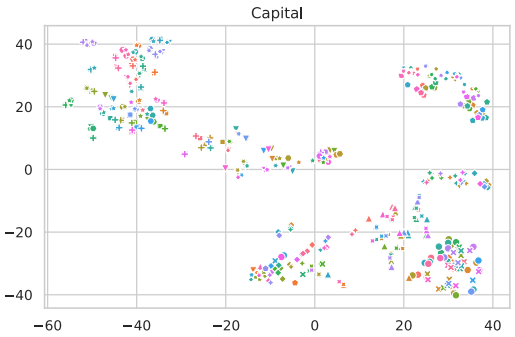


Figure 3: t-SNE of the encoded patterns from the *capital* relation. The colors represent the different subjects, while the shapes represent patterns. A knowledge-focused representation should cluster based on identical subjects (color), but instead the clustering is according to identical patterns (shape).

7.2 Representation Analysis

To provide insights on the models' representations, we inspect these after encoding the patterns.

Motivated by previous work that found that words with the same syntactic structure cluster together (Chi et al., 2020; Ravfogel et al., 2020) we perform a similar experiment to test if this behavior replicates with respect to knowledge: We encode the patterns, after filling the placeholders with subjects and masked tokens and inspect the last layer representations in the masked token position. When plotting the results using t-SNE (Maaten and Hinton, 2008) we mainly observe clustering based on the patterns, which suggests that encoding of knowledge of the entity is not the main component of the representations. Figure 3 demonstrates this for BERT-large encodings of the *capital* relation, which is highly consistent.[13] To provide a more quantitative assessment of this

[13] While some patterns are clustered based on the subjects (upper-left part), most of them are clustered based on patterns.

phenomenon, we also cluster the representations and set the number of centroids based on:[14] (1) the number of patterns in each relation, which aims to capture pattern-based clusters, and (2) the number of subjects in each relation, which aims to capture entity-based clusters. This would allow for a perfect clustering, in the case of perfect alignment between the representation and the inspected property. We measure the purity of these clusters using V-measure and observe that the clusters are mostly grouped by the patterns, rather than the subjects. Finally, we compute the Spearman correlation between the consistency scores and the V-measure of the representations. However, the correlation between these variables is close to zero,[15] therefore not explaining the models' behavior. We repeated these experiments while inspecting the objects instead of the subjects, and found similar trends. This finding is interesting since it means that (1) these representations are not knowledge-focused, i.e., their main component does not relate to knowledge, and (2) the representation by its entirety does not explain the behavior of the model, and thus only a subset of the representation does. This finding is consistent with previous work that observed similar trends for linguistic tasks (Elazar et al., 2021). We hypothesize that this disparity between the representation and the behavior of the model may be explained by a situation where the distance between representations largely does not reflect the distance between predictions, but rather other, behaviorally irrelevant factors of a sentence.

8 Improving Consistency in PLMs

In the previous sections, we showed PLMs are generally not consistent in their predictions, and previous works have noticed the lack of this property in a variety of downstream tasks. An ideal model would exhibit the consistency property after pretraining, and would then be able to transfer it to different downstream tasks. We therefore ask: Can we enhance current PLMs and make them more consistent?

8.1 Consistency Improved PLMs

We propose to improve the consistency of PLMs by continuing the pretraining step with a novel consistency loss. We make use of the T-REx tuples and the paraphrases from ParaRel🤘.

For each relation r_i, we have a set of paraphrased patterns P_i describing that relation. We use a PLM to encode all patterns in P_i, after populating a subject that corresponds to the relation r_i and a mask token. We expect the model to make the same prediction for the masked token for all patterns.

Consistency Loss Function As we evaluate the model using acc@1, the straight-forward consistency loss would require these predictions to be identical:

$$\min_{\theta} \text{sim}(\arg\max_{i} f_\theta(P_n)[i], \arg\max_{j} f_\theta(P_m)[j])$$

where $f_\theta(P_n)$ is the output of an encoding function (e.g., BERT) parameterized by θ (a vector) over input P_n, and $f_\theta(P_n)[i]$ is the score of the ith vocabulary item of the model.

However, this objective contains a comparison between the output of two argmax operations, making it discrete and discontinuous, and hard to optimize in a gradient-based framework. We instead relax the objective, and require that the predicted *distributions* $Q_n = \text{softmax}(f_\theta(P_n))$, rather than the top-1 prediction, be identical to each other. We use two-sided KL Divergence to measure similarity between distributions: $D_{KL}(Q_n^{r_i} || Q_m^{r_i}) + D_{KL}(Q_m^{r_i} || Q_n^{r_i})$ where $Q_n^{r_i}$ is the predicted distribution for pattern P_n of relation r_i.

As most of the vocabulary is not relevant for the predictions, we filter it down to the k tokens from the candidate set of each relation (§3.2). We want to maintain the original capabilities of the model—focusing on the candidate set helps to achieve this goal since most of the vocabulary is not affected by our new loss.

To encourage a more general solution, we make use of all the paraphrases together, and enforce all predictions to be as close as possible. Thus, the consistency loss for all pattern pairs for a particular relation r^i is:

$$\mathcal{L}_c = \sum_{n=1}^{k} \sum_{m=n+1}^{k} D_{KL}(Q_n^{r_i} || Q_m^{r_i}) + D_{KL}(Q_n^{r_i} || Q_m^{r_i})$$

MLM Loss Since the consistency loss is different from the Cross-Entropy loss the PLM is trained on, we find it important to continue the

[14] Using the KMeans algorithm.

[15] Except for BERT-large whole-word-masking, where the correlation is 39.5 ($p < 0.05$).

MLM loss on text data, similar to previous work (Geva et al., 2020).

We consider two alternatives for continuing the pretraining objective: (1) MLM on Wikipedia and (2) MLM on the patterns of the relations used for the consistency loss. We found that the latter works better. We denote this loss by $\mathcal{L}_{MLM}$.

Consistency Guided MLM Continual Training Combining our novel consistency loss with the regular MLM loss, we continue the PLM training by combining the two losses. The combination of the two losses is determined by a hyperparameter λ, resulting in the following final loss function:

$$\mathcal{L} = \lambda \mathcal{L}_c + \mathcal{L}_{MLM}$$

This loss is computed per relation, for one KB tuple. We have many of these instances, which we require to behave similarly. Therefore, we batch together $l = 8$ tuples from the same relation and apply the consistency loss function to all of them.

8.2 Setup

Since we evaluate our method on unseen relations, we also split train and test by relation type (e.g., location-based relations, which are very common in T-REx). Moreover, our method is aimed to be simple, effective, and to require only minimal supervision. Therefore, we opt to use only three relations: *original-language*, *named-after*, and *original-network*; these were chosen randomly, out of the non-location related relations.[16] For validation, we randomly pick three relations of the remaining relations and use the remaining 25 for testing.

We perform minimal tuning of the parameters ($\lambda \in 0.1, 0.5, 1$) to pick the best model, train for three epochs, and select the best model based on *Consistent-Acc* on the validation set. For efficiency reasons, we use the base version of BERT.

8.3 Improved Consistency Results

The results are presented in Table 7. We report aggregated results for the 25 relations in the test. We again report macro average (mean over relations) and standard deviation. We report the results of the majority baseline (first row), BERT-base (second row), and our new model (BERT-ft, third row).

[16]Many relations are location-based—not training on them prevents train-test leakage.

Model	Accuracy	Consistency	Consistent-Acc
majority	24.4±22.5	100.0±0.0	24.4±22.5
BERT-base	45.6±27.6	58.2±23.9	27.3±24.8
BERT-ft	**47.4**±27.3	**64.0**±22.9	**33.2**±27.0
-consistency	46.9±27.6	60.9±22.6	30.9±26.3
-typed	46.5±27.1	62.0±21.2	31.1±25.2
-MLM	16.9±21.1	80.8±27.1	9.1±11.5

Table 7: Knowledge and consistency results for the baseline, BERT base, and our model. The results are averaged over the 25 test relations. Underlined: best performance overall, including ablations. Bold: Best performance for BERT-ft and the two baselines (BERT-base, majority).

First, we note that our model significantly improves consistency: 64.0% (compared with 58.2% for BERT-base, an increase of 5.8 points). *Accuracy* also improves compared to BERT-base, from 45.6% to 47.4%. Finally, and most importantly, we see an increase of 5.9 points in *Consistent-Acc*, which is achieved due to the improved consistency of the model. Notably, these improvements arise from training on merely three relations, meaning that the model improved its consistency ability and generalized to new relations. We measure the statistical significance of our method compared to the BERT baseline, using McNemar's test (following Dror et al. [2018, 2020]) and find all results to be significant ($p \ll 0.01$).

We also perform an ablation study to quantify the utility of the different components. First, we report on the finetuned model without the consistency loss (-consistency). Interestingly, it does improve over the baseline (BERT-base), but it lags behind our finetuned model. Second, applying our loss on the candidate set rather than on the entire vocabulary is beneficial (-typed). Finally, by not performing the MLM training on the generated patterns (-MLM), the consistency results improve significantly (80.8%); however, this also hurts *Accuracy* and *Consistent-Acc*. MLM training seems to serve as a regularizer that prevents catastrophic forgetting.

Our ultimate goal is to improve consistency in PLMs for better performance on downstream tasks. Therefore, we also experiment with finetuning on SQuAD (Rajpurkar et al., 2016), and evaluating on paraphrased questions from SQuAD (Gan and Ng, 2019) using our consistency model. However, the results perform on par with the baseline model, both on SQuAD and the paraphrase

questions. More research is required to show that consistent PLMs can also benefit downstream tasks.

9 Discussion

Consistency for Downstream Tasks The rise of PLMs has improved many tasks but has also brought a lot of expectations. The standard usage of these models is pretraining on a large corpus of unstructured text and then finetuning on a task of interest. The first step is thought of as providing a good language-understanding component, whereas the second step is used to teach the format and the nuances of a downstream task.

As discussed earlier, consistency is a crucial component of many NLP systems (Du et al., 2019; Asai and Hajishirzi, 2020; Denis and Baldridge, 2009; Kryscinski et al., 2020) and obtaining this skill from a pretrained model would be extremely beneficial and has the potential to make specialized consistency solutions in downstream tasks redundant. Indeed, there is an ongoing discussion about the ability to acquire ''meaning'' from raw text signal alone (Bender and Koller, 2020). Our new benchmark makes it possible to track the progress of consistency in pretrained models.

Broader Sense of Consistency In this work we focus on one type of consistency, that is, consistency in the face of paraphrasing; however, consistency is a broader concept. For instance, previous work has studied the effect of negation on factual statements, which can also be seen as consistency (Ettinger, 2020; Kassner and Schütze, 2020). A consistent model is expected to return different answers to the prompts: ''*Birds* can *[MASK]*'' and ''*Birds* cannot *[MASK]*''. The inability to do so, as was shown in these works, also shows the lack of model consistency.

Usage of PLMs as KBs Our work follows the setup of Petroni et al. (2019) and Jiang et al. (2020), where PLMs are being tested as KBs. While it is an interesting setup for probing models for knowledge and consistency, it lacks important properties of standard KBs: (1) the ability to return more than a single answer and (2) the ability to return no answer. Although some heuristics can be used for allowing a PLM to do so, for example, using a threshold on the probabilities, it is not the way that the model was trained, and thus may not be optimal. Newer approaches that propose to use PLMs as a starting point to more complex systems have promising results and address these problems (Thorne et al., 2020).

In another approach, Shin et al. (2020) suggest using AutoPrompt to automatically generate prompts, or patterns, instead of creating them manually. This approach is superior to manual patterns (Petroni et al., 2019), or aggregation of patterns that were collected automatically (Jiang et al., 2020).

Brittleness of Neural Models Our work also relates to the problem of the brittleness of neural networks. One example of this brittleness is the vulnerability to adversarial attacks (Szegedy et al., 2014; Jia and Liang, 2017). The other problem, closer to the problem we explore in this work, is the poor generalization to paraphrases. For example, Gan and Ng (2019) created a paraphrase version for a subset of SQuAD (Rajpurkar et al., 2016), and showed that model performance drops significantly. Ribeiro et al. (2018) proposed another method for creating paraphrases and performed a similar analysis for visual question answering and sentiment analysis. Recently, Ribeiro et al. (2020) proposed CheckList, a system that tests a model's vulnerability to several linguistic perturbations.

ParaRel🤘 enables us to study the brittleness of PLMs, and separate facts that are robustly encoded in the model from mere 'guesses', which may arise from some heuristic or spurious correlations with certain patterns (Poerner et al., 2020). We showed that PLMs are susceptible to small perturbations, and thus, finetuning on a downstream task—given that training datasets are typically not large and do not contain equivalent examples—is not likely to perform better.

Can We Expect from LMs to Be Consistent? The typical training procedure of an LM does not encourage consistency. The standard training solely tries to minimize the log-likelihood of an unseen token, and this objective is not always aligned with consistency of knowledge. Consider for example the case of Wikipedia texts, as opposed to Reddit; their texts and styles may be very different and they may even describe contradictory facts. An LM can exploit the styles of each text to best fit the probabilities given to an unseen word, even if the resulting generations contradict each other.

Since the pretraining-finetuning procedure is currently the dominating one in our field, a great amount of the language capabilities that were learned during pre-training also propagates to the fine-tuned models. As such, we believe it is important to measure and improve consistency already in the pretrained models.

Reasons Behind the (In)Consistency Since LMs are not expected to be consistent, what are the reasons behind their predictions, when being consistent, or inconsistent?

In this work, we presented the predictions of multiple queries, and the representation space of one of the inspected models. However, this does not point to the origins of such behavior. In future work, we aim to inspect this question more closely.

10 Conclusion

In this work, we study the consistency of PLMs with regard to their ability to extract knowledge. We build a high-quality resource named ParaRel🤘 that contains 328 high-quality patterns for 38 relations. Using ParaRel🤘, we measure consistency in multiple PLMs, including BERT, RoBERTa, and ALBERT, and show that although the latter two are superior to BERT in other tasks, they fall short in terms of consistency. However, the consistency of these models is generally low. We release ParaRel🤘 along with data tuples from T-REx as a new benchmark to track knowledge consistency of NLP models. Finally, we propose a new simple method to improve model consistency, by continuing the pretraining with a novel loss. We show this method to be effective and to improve both the consistency of models as well as their ability to extract the correct facts.

Acknowledgments

We would like to thank Tomer Wolfson, Ido Dagan, Amit Moryossef, and Victoria Basmov for their helpful comments and discussions, and Alon Jacovi, Ori Shapira, Arie Cattan, Elron Bandel, Philipp Dufter, Masoud Jalili Sabet, Marina Speranskaya, Antonis Maronikolakis, Aakanksha Naik, Aishwarya Ravichander, and Aditya Potukuchi for the help with the annotations. We also thank the anonymous reviewers and the action editor, George Foster, for their valuable suggestions.

Yanai Elazar is grateful to be supported by the PBC fellowship for outstanding PhD candidates in Data Science and the Google PhD fellowship. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme, grant agreement no. 802774 (iEXTRACT). This work has been funded by the European Research Council (#740516) and by the German Federal Ministry of Education and Research (BMBF) under grant no. 01IS18036A. The authors of this work take full responsibility for its content. This research was also supported in part by grants from the National Science Foundation Secure and Trustworthy Computing program (CNS-1330596, CNS15-13957, CNS-1801316, CNS-1914486), and a DARPA Brandeis grant (FA8750-15-2-0277). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the NSF, DARPA, or the US Government.

References

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA corpora generation with roundtrip consistency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6168–6173. https://doi.org/10.18653/v1/P19-1620

Kim Allan Andersen and Daniele Pretolani. 2001. Easy cases of probabilistic satisfiability. *Annals of Mathematics and Artificial Intelligence*, 33(1):69–91. https://doi.org/10.1023/A:1012332915908

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. 2015. VQA: Visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2425–2433. https://doi.org/10.1109/ICCV.2015.279

Akari Asai and Hannaneh Hajishirzi. 2020. Logic-guided data augmentation and regularization for consistent question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational*

Linguistics, pages 5642–5650, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.499

Yonatan Belinkov, Nadir Durrani, Fahim Dalvi, Hassan Sajjad, and James Glass. 2017. What do neural machine translation models learn about morphology? In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 861–872. https://doi.org/10.18653/v1/P17-1080

Yonatan Belinkov and James Glass. 2019. Analysis methods in neural language processing: A survey. *Transactions of the Association for Computational Linguistics*, 7:49–72. https://doi.org/10.1162/tacl_a_00254

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198. https://doi.org/10.18653/v1/2020.acl-main.463

Rahul Bhagat and Eduard Hovy. 2013. What is a paraphrase? *Computational Linguistics*, 39(3):463–472. https://doi.org/10.1162/COLI_a_00166

Lukas Biewald. 2020. Experiment tracking with weights and biases. Software available from wandb.com.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics. https://doi.org/10.18653/v1/D15-1075

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Oana-Maria Camburu, Tim Rocktäschel, Thomas Lukasiewicz, and Phil Blunsom. 2018. E-SNLI: Natural language inference with natural language explanations. In *NeurIPS*.

Oana-Maria Camburu, Brendan Shillingford, Pasquale Minervini, Thomas Lukasiewicz, and Phil Blunsom. 2020. Make up your mind! Adversarial generation of inconsistent natural language explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4157–4165. https://doi.org/10.18653/v1/2020.acl-main.382

Kai-Wei Chang, Rajhans Samdani, Alla Rozovskaya, Nick Rizzolo, Mark Sammons, and Dan Roth. 2011. Inference protocols for coreference resolution. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task*, pages 40–44.

Ethan A. Chi, John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577, Online. Association for Computational Linguistics.

Jeff Da and Jungo Kasai. 2019. Cracking the contextual commonsense code: Understanding commonsense reasoning aptitude of deep contextual representations. In *Proceedings of the First Workshop on Commonsense Inference in Natural Language Processing*, pages 1–12, Hong Kong, China. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The Pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*, pages 177–190. Springer. https://doi.org/10.1007/11736790_9

Joe Davison, Joshua Feldman, and Alexander M. Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1173–1178. https://doi.org/10.18653/v1/D19-1109

Pascal Denis and Jason Baldridge. 2009. Global joint models for coreference resolution and named entity classification. *Procesamiento del Lenguaje Natural*, 42.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1383–1392. https://doi.org/10.18653/v1/P18-1128

Rotem Dror, Lotem Peled-Cohen, Segev Shlomov, and Roi Reichart. 2020. Statistical significance testing for natural language processing. *Synthesis Lectures on Human Language Technologies*, 13(2):1–116. https://doi.org/10.2200/S00994ED1V01Y202002HLT045

Xinya Du, Bhavana Dalvi, Niket Tandon, Antoine Bosselut, Wen-tau Yih, Peter Clark, and Claire Cardie. 2019. Be consistent! Improving procedural text comprehension using label consistency. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2347–2356.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2021. Amnesic Probing: Behavioral Explanation with Amnesic Counterfactuals. *Transactions of the Association for Computational Linguistics*, 9:160–175. https://doi.org/10.1162/tacl_a_00359

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48. https://doi.org/10.1162/tacl_a_00298

Maxwell Forbes, Ari Holtzman, and Yejin Choi. 2019. Do neural language representations learn physical commonsense? In *CogSci*.

Wee Chung Gan and Hwee Tou Ng. 2019. Improving the robustness of question answering systems to question paraphrasing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6065–6075.

Mor Geva, Ankit Gupta, and Jonathan Berant. 2020. Injecting numerical reasoning skills into language models. In *Association for Computational Linguistics (ACL)*. https://doi.org/10.18653/v1/2020.acl-main.89

Yoav Goldberg. 2019. Assessing BERT's syntactic abilities. *arXiv preprint arXiv:1901.05287*.

Pierre Hansen and Brigitte Jaumard. 2000. Probabilistic satisfiability. In *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, pages 321–367, Springer. https://doi.org/10.1007/978-94-017-1737-3_8

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *North American Chapter*

of the Association for Computational Linguistics: Human Language Technologies (NAACL). Association for Computational Linguistics.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spaCy: Industrial-strength Natural Language Processing in Python. https://doi.org/10.5281/zenodo.1212303.

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031. https://doi.org/10.18653/v1/D17-1215

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438. https://doi.org/10.1162/tacl_a_00324

Nora Kassner, Philipp Dufter, and Hinrich Schütze. 2021a. Multilingual lama: Investigating knowledge in multilingual pretrained language models.

Nora Kassner, Benno Krojer, and Hinrich Schütze. 2020. Are pretrained language models symbolic reasoners over knowledge? In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 552–564, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.conll-1.45

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.698

Nora Kassner, Oyvind Tafjord, Hinrich Schütze, and Peter Clark. 2021b. Enriching a model's notion of belief using a persistent memory. *CoRR*, abs/2104.08401.

Wojciech Kryscinski, Bryan McCann, Caiming Xiong, and Richard Socher. 2020. Evaluating the factual consistency of abstractive text summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9332–9346. https://doi.org/10.18653/v1/2020.emnlp-main.750

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Tao Li, Vivek Gupta, Maitrey Mehta, and Vivek Srikumar. 2019. A logic-driven framework for consistency of neural models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3924–3935, Hong Kong, China. Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1405

Tal Linzen. 2020. How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5210–5217. https://doi.org/10.18653/v1/2020.acl-main.465

Tal Linzen, Emmanuel Dupoux, and Yoav Goldberg. 2016. Assessing the ability of LSTMs to learn syntax-sensitive dependencies. *Transactions of the Association for Computational Linguistics*, 4:521–535. https://doi.org/10.1162/tacl_a_00115

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605.

Rebecca Marvin and Tal Linzen. 2018. Targeted syntactic evaluation of language models. In *Proceedings of the 2018 Conference on*

Empirical Methods in Natural Language Processing, pages 1192–1202, Brussels, Belgium. Association for Computational Linguistics. https://doi.org/10.18653/v1/D18-1151

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448. https://doi.org/10.18653/v1/P19-1334

David Picado Muiño. 2011. Measuring and repairing inconsistency in probabilistic knowledge bases. *International Journal of Approximate Reasoning*, 52(6):828–840. https://doi.org/10.1016/j.ijar.2011.02.003

Kim Anh Nguyen, Sabine Schulte im Walde, and Ngoc Thang Vu. 2016. Integrating distributional lexical contrast into word embeddings for antonym-synonym distinction. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 454–459. https://doi.org/10.18653/v1/P16-2074

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54. https://doi.org/10.18653/v1/D19-1005

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. In *Automated Knowledge Base Construction*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics. https://doi.org/10.18653/v1/D19-1250

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: Efficient-yet-effective entity embeddings for bert. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 803–818. https://doi.org/10.18653/v1/2020.findings-emnlp.71

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392. https://doi.org/10.18653/v1/D16-1264

Shauli Ravfogel, Yanai Elazar, Jacob Goldberger, and Yoav Goldberg. 2020. Unsupervised distillation of syntactic information from contextualized word representations. In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 91–106, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.blackboxnlp-1.9

Abhilasha Ravichander, Eduard Hovy, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit

Cheung. 2020. On the systematicity of probing contextualized word representations: The case of hypernymy in BERT. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*, pages 88–102, Barcelona, Spain (Online). Association for Computational Linguistics.

Marco Tulio Ribeiro, Carlos Guestrin, and Sameer Singh. 2019. Are red roses red? Evaluating consistency of question-answering models. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6174–6184, Florence, Italy. Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1621

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging NLP models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865. https://doi.org/10.18653/v1/P18-1079

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of NLP models with CheckList. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.442

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426. https://doi.org/10.18653/v1/2020.emnlp-main.437

Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. A primer in bertology: What we know about how bert works. *Transactions of the Association for Computational Linguistics*, 8:842–866. https://doi.org/10.1162/tacl_a_00349

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting knowledge from language models with automatically generated prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.346

Micah Shlain, Hillel Taub-Tabib, Shoval Sadde, and Yoav Goldberg. 2020. Syntactic search by example. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 17–23, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-demos.3

Vered Shwartz and Yejin Choi. 2020. Do neural language models overcome reporting bias? In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6863–6870. https://doi.org/10.18653/v1/2020.coling-main.605

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. 2014. Intriguing properties of neural networks. In *International Conference on Learning Representations*.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2020. oLMpics—on what language model pre-training captures. *Transactions of the Association for Computational Linguistics*, 8:743–758. https://doi.org/10.1162/tacl_a_00342

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. Bert rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601. https://doi.org/10.18653/v1/P19-1452

Matthias Thimm. 2009. Measuring inconsistency in probabilistic knowledge bases. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI'09)*, pages 530–537. AUAI Press.

Matthias Thimm. 2013. Inconsistency measures for probabilistic logics. *Artificial Intelligence*, 197:1–24. https://doi.org/10.1016/j.artint.2013.02.001

James Thorne, Majid Yazdani, Marzieh Saeidi, Fabrizio Silvestri, Sebastian Riedel, and Alon Halevy. 2020. Neural databases. *arXiv preprint arXiv:2010.06973*.

Alex Warstadt, Yian Zhang, Xiaocheng Li, Haokun Liu, and Samuel R. Bowman. 2020. Learning which features matter: RoBERTa acquires a preference for linguistic generalizations (eventually). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 217–235, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-main.16

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122. Association for Computational Linguistics. https://doi.org/10.18653/v1/N18-1101

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-demos.6

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2020. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model. In *8th International Conference on Learning Representations, ICLR* 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net.

Xikun Zhang, Deepak Ramachandran, Ian Tenney, Yanai Elazar, and Dan Roth. 2020. Do language embeddings capture scales? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 4889–4896. https://doi.org/10.18653/v1/2020.findings-emnlp.439

A Implementation Details

We heavily rely on Hugging Face's Transformers library (Wolf et al., 2020) for all experiments involving the PLMs. We used Weights & Biases for tracking and logging the experiments (Biewald, 2020). Finally, we used sklearn (Pedregosa et al., 2011) for other ML-related experiments.

B Paraphrases Analysis

We provide a characterization of the paraphrase types included in our dataset.

We analyze the type of paraphrases in ParaRel🤘. We sample 100 paraphrase pairs from the agreement evaluation that were labeled as paraphrases and annotate the paraphrase type. Notice that the paraphrases can be complex; as such, multiple transformations can be annotated for each pair. We mainly use a subset of paraphrase types categorized by Bhagat and Hovy (2013), but also define new types that were not covered by that work. We begin by briefly defining the types of paraphrases found in ParaRel🤘 from Bhagat and Hovy (2013) (more thorough definitions can be found in their paper), and then define the new types we observed.

1. Synonym substitution: Replacing a word/phrase by a synonymous word/phrase, in the appropriate context.

2. Function word variations: Changing the function words in a sentence/phrase without affecting its semantics, in the appropriate context.

3. Converse substitution: Replacing a word/phrase with its converse and inverting the

relationship between the constituents of a sentence/phrase, in the appropriate context, presenting the situation from the converse perspective.

4. Change of tense: Changing the tense of a verb, in the appropriate context.

5. Change of voice: Changing a verb from its active to passive form and vice versa results in a paraphrase of the original sentence/phrase.

6. Verb/Noun conversion: Replacing a verb by its corresponding nominalized noun form and vice versa, in the appropriate context.

7. External knowledge: Replacing a word/phrase by another word/phrase based on extra-linguistic (world) knowledge, in the appropriate context.

8. Noun/Adjective conversion: Replacing a verb by its corresponding adjective form and vice versa, in the appropriate context.

9. Change of aspect: Changing the aspect of a verb, in the appropriate context.

We also define several other types of paraphrases not covered in Bhagat and Hovy (2013) (potentially because they did not occur in the corpora they have inspected).

a. Irrelevant addition: Addition or removal of a word or phrase, that does not affect the meaning of the sentence (as far as the relation of interest is concerned), and can be inferred from the context independently.

b. Topicalization transformation: A transformation from or to a topicalization construction. Topicalization is a construction in which a clause is moved to the beginning of its enclosing clause.

c. Apposition transformation: A transformation from or to an apposition construction. In an apposition construction, two noun phrases where one identifies the other are placed one next to each other.

d. Other syntactic movements: Includes other types of syntactic transformations that are not part of the other categories. This includes cases such as moving an element from a coordinate construction to the subject position as in the last example in Table 8. Another type of transformation is in the following paraphrase: ‘‘[X] plays in [Y] position.’’ and ‘‘[X] plays in the position of [Y].’’ where a compound noun-phrase is replaced with a prepositional phrase.

We report the percentage of each type, along with examples of paraphrases in Table 8. The most common paraphrase is the ‘synonym substitution’, following ‘function words variations’ which occurred 41 and 16 times, respectively. The least common paraphrase is ‘change of aspect’, which occurred only once in the sample.

The full ParaRel🤘 resource can be found at: https://github.com/yanaiela/pararel/tree/main/data/pattern_data/graphs_json.

Paraphrase Type	Pattern #1	Pattern #2	Relation	N.
Synonym substitution	[X] died in [Y].	[X] expired at [Y].	place of death	41
Function words variations	[X] is [Y] citizen.	[X], who is a citizen of [Y].	country of citizenship	16
Converse substitution	[X] maintains diplomatic relations with [Y].	[Y] maintains diplomatic relations with [X].	diplomatic relation	10
Change of tense	[X] is developed by [Y].	[X] was developed by [Y].	developer	10
Change of voice	[X] is owned by [Y].	[Y] owns [X].	owned by	7
Verb/Noun conversion	The headquarter of [X] is in [Y].	[X] is headquartered in [Y].	headquarters location	7
External knowledge	[X] is represented by music label [Y].	[X], that is represented by [Y].	record label	3
Noun/Adjective conversion	The official language of [X] is [Y].	The official language of [X] is the [Y] language.	official language	2
Change of aspect	[X] plays in [Y] position.	playing as an [X], [Y]	position played on team	1
Irrelevant addition	[X] shares border with [Y].	[X] shares a common border with [Y].	shares border with	11
Topicalization transformation	[X] plays in [Y] position.	playing as a [Y], [X]	position played on team	8
Apposition transformation	[X] is the capital of [Y].	[Y]'s capital, [X].	capital of	4
Other syntactic movements	[X] and [Y] are twin cities.	[X] is a twin city of [Y].	twinned administrative body	10

Table 8: Different types of paraphrases in PARAREL🤘. We report examples from each paraphrase type, along with the type of relation, and the number of examples from the specific transformation from a random subset of 100 pairs. Each pair can be classified into more than a single transformation (we report one for brevity), thus the sum of transformation is more than 100.

Chapter 4

Multilingual LAMA: Investigating Knowledge in Multilingual Pretrained Language Models

Multilingual LAMA: Investigating Knowledge in Multilingual Pretrained Language Models

Nora Kassner*, **Philipp Dufter***, **Hinrich Schütze**
Center for Information and Language Processing (CIS), LMU Munich, Germany
{kassner,philipp}@cis.lmu.de

Abstract

Recently, it has been found that monolingual English language models can be used as knowledge bases. Instead of structural knowledge base queries, masked sentences such as "Paris is the capital of [MASK]" are used as probes. We translate the established benchmarks TREx and GoogleRE into 53 languages. Working with mBERT, we investigate three questions. (i) Can mBERT be used as a multilingual knowledge base? Most prior work only considers English. Extending research to multiple languages is important for diversity and accessibility. (ii) Is mBERT's performance as knowledge base language-independent or does it vary from language to language? (iii) A multilingual model is trained on more text, e.g., mBERT is trained on 104 Wikipedias. Can mBERT leverage this for better performance? We find that using mBERT as a knowledge base yields varying performance across languages and pooling predictions across languages improves performance. Conversely, mBERT exhibits a language bias; e.g., when queried in Italian, it tends to predict Italy as the country of origin.

1 Introduction

Pretrained language models (LMs) (Peters et al., 2018; Howard and Ruder, 2018; Devlin et al., 2019) can be finetuned to a variety of natural language processing (NLP) tasks and generally yield high performance. Increasingly, these models and their generative variants are used to solve tasks by simple text generation, without any finetuning (Brown et al., 2020). This motivated research on how much knowledge is contained in LMs: Petroni et al. (2019) used models pretrained with masked language to answer fill-in-the-blank templates such as "Paris is the capital of [MASK]."

* Equal contribution - random order.

	Query	Two most frequent predictions
en	X was created in MASK.	[Japan (170), Italy (56), . . .]
de	X wurde in MASK erstellt.	[Deutschland (217), Japan (70), . . .]
it	X è stato creato in MASK.	[Italia (167), Giappone (92), . . .]
nl	X is gemaakt in MASK.	[Nederland (172), Italië (50), . . .]
en	X has the position of MASK.	[bishop (468), God (68), ...]
de	X hat die Position MASK.	[WW (261), Ratsherr (108), ...]
it	X ha la posizione di MASK.	[pastore (289), papa (138), ...]
nl	X heeft de positie van MASK.	[burgemeester (400), bisschop (276) , ...]

Table 1: Language bias when querying (TyQ) mBERT. Top: For an Italian cloze question, Italy is favored as country of origin. Bottom: There is no overlap between the top-ranked predictions, demonstrating the influence of language – even though the facts are the same: the same set of triples is evaluated across languages. Table 3 shows that pooling predictions across languages addresses bias and improves performance. WW = "Wirtschaftswissenschaftler".

This research so far has been exclusively on English. In this paper, we focus on using *multilingual* pretrained LMs as knowledge bases. Working with mBERT, we investigate three questions. (i) Can mBERT be used as a multilingual knowledge base? Most prior work only considers English. Extending research to multiple languages is important for diversity and accessibility. (ii) Is mBERT's performance as knowledge base language-independent or does it vary from language to language? To answer these questions, we translate English datasets and analyze mBERT for 53 languages. (iii) A multilingual model is trained on more text, e.g., BERT's training data contains the English Wikipedia, but mBERT is trained on 104 Wikipedias. Can mBERT leverage this fact? Indeed, we show that pooling across languages helps performance.

In summary our contributions are: **i)** We automatically create a multilingual version of TREx and GoogleRE covering 53 languages. **ii)** We use an alternative to fill-in-the-blank querying – ranking entities of the type required by the template (e.g., cities) – and show that it is a better tool

to investigate knowledge captured by pretrained LMs. **iii)** We show that mBERT answers queries across languages with varying performance: it works reasonably for 21 and worse for 32 languages. **iv)** We give evidence that the query language affects results: a query formulated in Italian is more likely to produce Italian entities (see Table 1). **v)** Pooling predictions across languages improves performance by large margins and even outperforms monolingual English BERT. Code and data are available online (https://github.com/norakassner/mlama).

2 Data

2.1 LAMA

We follow the LAMA setup introduced by Petroni et al. (2019). More specifically, we use data from TREx (Elsahar et al., 2018) and GoogleRE.[1] Both consist of triples of the form (object, relation, subject). The underlying idea of LAMA is to query knowledge from pretrained LMs using templates without any finetuning: the triple (Paris, capital-of, France) is queried with the template "Paris is the capital of [MASK]." In LAMA, TREx has 34,039 triples across 41 relations, GoogleRE 5528 triples and 3 relations. Templates for each relation have been manually created by Petroni et al. (2019). We call all triples from TREx and GoogleRE together *LAMA*.

LAMA has been found to contain many "easy-to-guess" triples; e.g., it is easy to guess that a person with an Italian sounding name is born in Italy. *LAMA-UHN* is a subset of triples that are hard to guess introduced by Poerner et al. (2020).

2.2 Translation

We translate both entities and templates. We use Google Translate to translate templates in the form "[X] is the capital of [Y]". After translation, all templates were checked for validity (i.e., whether they contain "[X]", "[Y]" exactly once) and corrected if necessary. In addition, German, Hindi and Japanese templates were checked by native speakers to assess translation quality (see Table 2). To translate the entity names, we used Wikidata and Google knowledge graphs.

mBERT covers 104 languages. Google Translate covers 77 of these. Wikidata and Google Knowledge Graph do not provide entity translations for all languages and not all entities are contained in the knowledge graphs. For English we can find a total of 37,498 triples which we use from now on. On average, 34% of triples could be translated (macro average over languages). We only consider languages with a coverage above 20%, resulting in the final number of languages we include in our study: 53. The macro average of translated triples in these 53 languages is 43%. Figure 1 gives statistics. We call the translated dataset *mLAMA*.

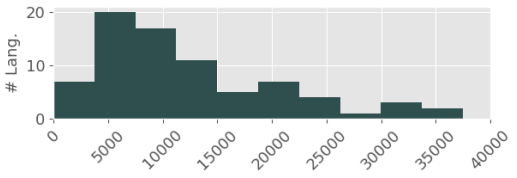


Figure 1: x-axis is the number of translated triples, y-axis the number of languages. There are 39,567 triples in the original LAMA (TREx and GoogleRE).

3 Experiments

3.1 Model

We work with mBERT (Devlin et al., 2019), a model pretrained on the 104 largest Wikipedias. We denote mBERT queried in language x as mBERT[x]. As comparison we use the English BERT-Base model and refer to it as BERT. In initial experiments with XLM-R (Conneau et al., 2020) we observed worse performance, similar to Jiang et al. (2020a). Thus, for simplicity we only report results on mBERT.

3.2 Typed and Untyped Querying

Petroni et al. (2019) use templates like "Paris is the capital of [MASK]" and give $\arg\max_{w \in \mathcal{V}} p(w|t)$ as answer where V is the vocabulary of the LM and $p(w|t)$ is the (log-)probability that word w gets predicted in the template t. Thus the object of a triple must be contained in the vocabulary of the language model. This has two drawbacks: it reduces the number of triples that can be considered drastically and hinders performance comparisons across LMs with different vocabularies. We refer to this procedure as *UnTyQ*.

We propose to use typed querying, *TyQ*: for each relation a candidate set $\mathcal{C}$ is created and the prediction becomes $\arg\max_{c \in \mathcal{C}} p(c|t)$. For templates like "[X] was born in [MASK]", we know which entity type to expect, in this case cities. We observed that (English-only) BERT-base predicts city

[1] code.google.com/archive/p/relation-extraction-corpus/

names for MASK whereas mBERT predicts years for the same template. TyQ prevents this.

We choose as $\mathcal{C}$ the set of objects across all triples for a single relation. The candidate set could also be obtained from an entity typing system (e.g., (Yaghoobzadeh and Schütze, 2016)), but this is beyond the scope of this paper. Variants of TyQ have been used before (Xiong et al., 2020).

3.3 Singletoken vs. Multitoken Objects

Assuming that objects are in the vocabulary (Petroni et al., 2019) is a restrictive assumption, even more in the multilingual case as e.g., "Hamburg" is in the mBERT vocabulary, but French "Hambourg" is tokenized to ["Ham", "##bourg"]. We consider multitoken objects by including multiple [MASK] tokens in the templates. For both TyQ and UnTyQ we compute the score that a multitoken object is predicted by taking the average of the log probabilities for its individual tokens.

Given a template t (e.g., "[X] was born in [Y].") let t_1 be the template with one mask token, (i.e., "[X] was born in [MASK].") and t_k be the template with k mask tokens (i.e., "[X] was born in [MASK] [MASK] ... [MASK]."). We denote the log probability that the token $w \in V$ is predicted at ith mask token as $p(m_i = w|t_k)$, where V is the vocabulary of the LM. To compute $p(e|t)$ for an entity e that is tokenized into l tokens $\epsilon_1, \epsilon_2, \ldots, \epsilon_l$ we simply average the log probabilities across tokens:

$$p(e|t) = \frac{1}{l} \sum_{i=1}^{l} p(m_i = \epsilon_i|t_l).$$

If k is the maximum number of tokens of any entity $e \in \mathcal{C}$ gets split into, we consider all templates $t_1, \ldots, t_k$, with C being the candidate set. The prediction is then the word with the highest average log probability across all templates $t_1, \ldots, t_k$.

Note that for UnTyQ the space of possible predictions is $V \times V \times \cdots \times V$ whereas for TyQ it is the candidate set $\mathcal{C}$.

3.4 Evaluation

We compute precision at one for each relation, i.e., $1/|T| \sum_{t \in T} \mathbb{1}\{\hat{t}_{object} = t_{object}\}$ where T is the set of all triples and $\hat{t}_{object}$ is the object predicted by TyQ or UnTyQ. Note that T is different for each language. Our final measure (p1) is then the precision at one averaged over relations (i.e., macro average). Results for multiple languages are the macro average p1 across languages.

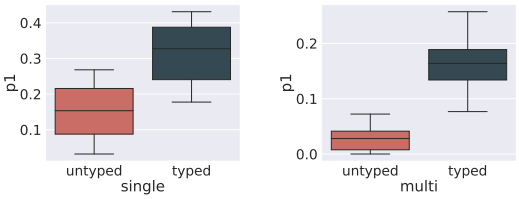


Figure 2: Distribution of p1 scores for 53 languages in UnTyQ vs. TyQ. Left: singletoken (object = 1 token). Right: multitoken (object > 1 token).

4 Results and Discussion

We first investigate TyQ and UnTyQ and find that TyQ is better suited for investigating knowledge in LMs. After exploring the translation quality, we use TyQ on mLAMA and observe rather stable performance for 21 and poor performance for 32 languages. When investigating the languages more closely, we find that prediction results highly depend on the language. Finally, we validate our initial hypothesis that mBERT can leverage its multilinguality by pooling predictions: pooling indeed performs better.

4.1 UnTyQ vs. TyQ

Figure 2 shows the distribution of p1 scores for single and multitoken objects. As expected, TyQ works better, both for single and multitoken objects. With UnTyQ, performance not only depends on the model's knowledge, but on at least three extraneous factors: **(i)** Does the model understand the type constraints of the template (e.g., in "X is the capital of Y", Y must be a country)? **(ii)** How "fluent" a substitution is an object under linguistic constraints (e.g., morphology) that can be viewed as orthogonal to knowledge? Many English templates cannot be translated into a single template in many languages, e.g., "in X" (with X a country) has different translations in French: "à Chypre", "au Mexique", "en Inde". But the LAMA setup requires a single template. By enforcing the type, we reduce the number of errors that are due to surface fluency. **(iii)** The inadequacy of the original LAMA setup for multitoken answers. Figure 2 (right) shows that the original UnTyQ struggles with multitokens (mean p1 .03 vs. .17 for TyQ).

Overall, TyQ allows us to focus the evaluation on the core question: what knowledge is contained in LMs? From now on, we report numbers in the TyQ setting.

Manual template tuning or automatic template

	machine translated	manually corrected		manually paraphrased	
de	18.1	19.4	(6)	20.9	(18)
hi	5.4	6.2	(14)	6.2	(1)
ja	0.4	0.4	(14)	0.7	(5)

Table 2: Effect of manual template modification on UnTyQ. Shown is p1, number of templates modified (in brackets). Templates are modified to correct mistakes from machine translation and paraphrased to achieve the correct object type. Manual template correction has a small effect on UnTyQ.

mining (Jiang et al., 2020b) has been investigated in the literature to approach the typing problem. We had native speakers check templates for German, Hindi and Japanese, correct mistakes in the automatic translation and paraphrase the template to obtain predictions with the correct type. Table 2 shows that corrections do not yield strong improvements. We conclude that template modifications are not an effective solution for the typing problem.

4.2 Translation Quality

Contemporaneous work by Jiang et al. (2020a) provides manual translations of LAMA templates for 23 languages respecting grammatical gender and inflection constraints. We evaluate our machine translated templates by comparing performance on a common subset of 14 languages using TyQ querying on the TREx subset. Surprisingly, we find a performance difference of 1 percentage points (0.23 vs. 0.24, p1 averaged over languages) in favor of the machine translated templates. This indicates that the machine translated templates in combination with TyQ exhibit comparable performance but come with the benefit of larger language coverage (53 vs. 23 languages).

4.3 Multilingual Performance

In mLAMA, not all triples are available in all languages. Thus absolute numbers are not comparable across languages and we adopt a relative performance comparison: we report p1 of a model-language combination divided by p1 of mBERT's performance in English (mBERT[en]) on the exact same set of triples and call this *rel-p1*. A rel-p1 score of 0.5 for mBERT[fi] means that p1 of mBERT on Finnish is half of mBERT[en]'s performance on the same triples. rel-p1 of English BERT is usually greater than 1 as monolingual BERT tends to outperform mBERT[en].

Figure 3 shows that mBERT performs reasonably well for 21 languages, but for 32 languages

	LAMA	LAMA-UHN
BERT	38.5	29.0
mBERT[en]	35.0	25.7
mBERT[pooled]	**41.1**	**32.1**

Table 3: p1 for BERT, mBERT queried in English, mBERT pooled on LAMA and LAMA-UHN.

rel-p1 is less than 0.6 (i.e., their p1 is 60% of English's p1). We conclude that mBERT does not exhibit a stable performance across languages. The variable performance (from 20% to almost 100% rel-p1) indicates that mBERT has no common representation for, say, "Paris" across languages, i.e., mBERT representations are language-dependent.

4.4 Bias

If mBERT captured knowledge independent of language, we should get similar answers across languages for the same relation. However, Table 1 shows that mBERT exhibits language-specific biases; e.g., when queried in Italian, it tends to predict Italy as the country of origin. This effect occurs for several relations: Table 4 in the supplementary presents data for ten relations and four languages.

4.5 Pooling

We investigate pooling of predictions across languages by picking the object predicted by the majority of languages. Table 3 shows that pooled mBERT outperforms mBERT[en] by 6 percentage points on LAMA, presumably in part because the language-specific bias is eliminated. mBERT[pooled] even outperforms BERT by 3 percentage points on LAMA-UHN. This indicates that mBERT can leverage the fact that it is trained on 104 Wikipedias vs. just one and even outperforms the much stronger model BERT.

5 Related Work

Petroni et al. (2019) first asked the question: can pretrained LMs function as knowledge bases? Subsequent analyses focused on different aspects, such as negation (Kassner and Schütze, 2020), easy to guess names (Poerner et al., 2020), integrating adapters (Wang et al., 2020) or finding alternatives to a "fill-in-the-blank" approach with single-token answers (Bouraoui et al., 2020; Heinzerling and Inui, 2020; Jiang et al., 2020b). Other work combines pretrained LM with information retrieval (Guu et al., 2020; Lewis et al., 2020a; Izacard and Grave, 2020; Kassner and Schütze, 2020; Petroni

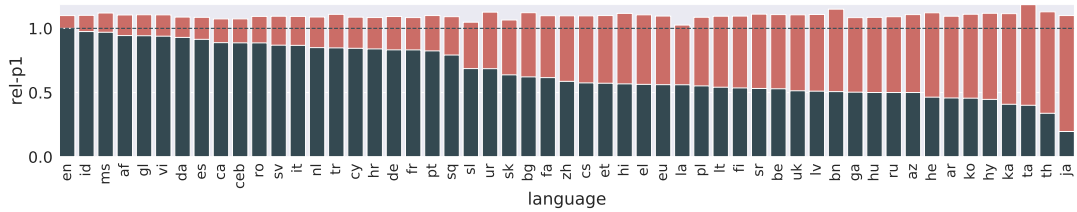


Figure 3: p1 of BERT (red) vs mBERT[x] (blue) divided by p1 of mBERT[en] on the same set of triples in each language x. mBERT captures less factual knowledge than monolingual English BERT. While performance is reasonable for 21 languages, it is below 60% for 32 languages. Dashed line is rel-p1 of mBERT[en] (by definition equal to 1.0). Performance of BERT varies slightly as the set of triples is different for each language. Note that the Wikipedia of Cebuano (ceb) consists mostly of machine translated articles.

et al., 2020). None of this work addresses languages other than English.

Multilingual models like mBERT (Devlin et al., 2019) and XLM-R (Conneau et al., 2020) perform well for zero-shot crosslingual transfer (Hu et al., 2020). However, we are not aware of any prior work that analyzed to what degree pretrained multilingual models can be used as knowledge bases. There are many multilingual question answering datasets such as XQuAD (Artetxe et al., 2020), TiDy (Clark et al., 2020), MKQA (Longpre et al., 2020) and MLQA (Lewis et al., 2020b). Usually, multilingual models are finetuned to solve such tasks. Our goal is not to improve question answering or create an alternative multilingual question answering dataset, but instead to investigate which knowledge is contained in pretrained multilingual LMs without any kind of supervised finetuning.

There is a range of alternative multilingual knowledge bases that could be used for evaluation. Those include ConceptNet (Speer et al., 2017) or BabelNet (Navigli and Ponzetto, 2010). We decided to provide a translated versions of TREx and GoogleRE for the sake of comparability across languages. By translating manually created templates and entities we can ensure comparability across languages. This is not possible for crowd-sourced databases like ConceptNet.

In contemporaneous work, Jiang et al. (2020a) create and investigate a multilingual version of LAMA. They provide human template translations for 23 languages, propose several methods for multitoken decoding and code-switching, and experiment with a number of PLMs. In contrast to their work, we investigate typed querying, focus on comparabiliy and pooling across languages, and explore language biases.

6 Conclusion

We presented mLAMA, a dataset to investigate knowledge in language models (LMs) in a multilingual setting covering 53 languages. While our results suggest that correct entities can be retrieved for many languages, there is a clear performance gap between English and, e.g., Japanese and Thai. This suggests that mBERT is not storing entity knowledge in a language-independent way. Experiments investigating language bias confirm this finding. We hope that this paper and the dataset we publish will stimulate research on investigating knowledge in LMs *multilingually* rather than just in English.

Acknowledgements

This work was supported by the European Research Council (# 740516) and the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibility for its content. The second author was supported by the Bavarian research institute for digital transformation (bidt) through their fellowship program. We thank Yannick Couzinié and Karan Tiwana for correcting the Japanese and Hindi templates. We thank the anonymous reviewers for valuable comments.

References

Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. 2020. On the cross-lingual transferability of monolingual representations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4623–4637, Online. Association for Computational Linguistics.

Zied Bouraoui, José Camacho-Collados, and Steven Schockaert. 2020. Inducing relational knowledge from BERT. In *The Thirty-Fourth AAAI Conference*

on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020, pages 7456–7463. AAAI Press.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Jonathan H. Clark, Jennimaria Palomaki, Vitaly Nikolaev, Eunsol Choi, Dan Garrette, Michael Collins, and Tom Kwiatkowski. 2020. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC-2018)*, Miyazaki, Japan. European Languages Resources Association (ELRA).

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: retrieval-augmented language model pre-training. *Computing Research Repository*, arXiv:2002.08909.

Benjamin Heinzerling and Kentaro Inui. 2020. Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. *Computing Research Repository*, arXiv:2008.09036.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 328–339, Melbourne, Australia. Association for Computational Linguistics.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. XTREME: A massively multilingual multi-task benchmark for evaluating cross-lingual generalisation. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 4411–4421. PMLR.

Gautier Izacard and E. Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *ArXiv*, abs/2007.01282.

Zhengbao Jiang, Antonios Anastasopoulos, Jun Araki, Haibo Ding, and Graham Neubig. 2020a. X-FACTR: Multilingual factual knowledge retrieval from pretrained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5943–5959, Online. Association for Computational Linguistics.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020b. How can we know what language models know. *Transactions of the Association for Computational Linguistics*, 8:423–438.

Nora Kassner and Hinrich Schütze. 2020. BERT-kNN: Adding a kNN search component to pretrained language models for better QA. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*, pages 3424–3430. Association for Computational Linguistics.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7811–7818, Online. Association for Computational Linguistics.

Mike Lewis, Marjan Ghazvininejad, Gargi Ghosh, Armen Aghajanyan, Sida Wang, and Luke Zettlemoyer. 2020a. Pre-training via paraphrasing. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Patrick Lewis, Barlas Oguz, Ruty Rinott, Sebastian Riedel, and Holger Schwenk. 2020b. MLQA: Evaluating cross-lingual extractive question answering. In

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7315–7330, Online. Association for Computational Linguistics.

Shayne Longpre, Yi Lu, and Joachim Daiber. 2020. MKQA: A linguistically diverse benchmark for multilingual open domain question answering. *CoRR*, abs/2007.15207.

Roberto Navigli and Simone Paolo Ponzetto. 2010. Babelnet: Building a very large multilingual semantic network. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*, pages 216–225. The Association for Computer 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, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. In *Automated Knowledge Base Construction*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander H. Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 2463–2473. Association for Computational Linguistics.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2020. E-BERT: Efficient-yet-effective entity embeddings for BERT. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 803–818, Online. Association for Computational Linguistics.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 4444–4451. AAAI Press.

Ruize Wang, Duyu Tang, Nan Duan, Zhongyu Wei, Xuanjing Huang, Jianshu Ji, Guihong Cao, Daxin Jiang, and Ming Zhou. 2020. K-adapter: Infusing knowledge into pre-trained models with adapters. *Computing Research Repository*, arXiv:2002.01808.

Wenhan Xiong, Jingfei Du, William Yang Wang, and Veselin Stoyanov. 2020. Pretrained encyclopedia: Weakly supervised knowledge-pretrained language model. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Yadollah Yaghoobzadeh and Hinrich Schütze. 2016. Intrinsic subspace evaluation of word embedding representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 236–246, Berlin, Germany. Association for Computational Linguistics.

A Language Bias

Table 4 shows the language bias for 10 relations. For each relation we aggregated the predictions across all triples and show the most common two predicted entities together with its count (in brackets). The querying language clearly affects results. The effect is drastic for relations that ask for a country (e.g., P495 or P1001). P39 yields very different results without exhibiting a clear pattern. Other relations such as P463 or P178 are rather stable.

B Data Samples

Table 4 and Table 5 show randomly sampled entries from the data.

C Pretraining Data

We investigate whether performance across languages is correlated with the amount of pretraining data for each language. To this end we investigate the number of articles per language as of January 2021 [2] and p1 for TyQ in Figure 6. We do not have access to the original pretraining data of mBERT. Thus, the number of articles we consider in the analysis might be different to the actual data used to train mBERT.

af	Microsoft Word word ontwikkel deur Microsoft.
	Aix-en-Provence is die hoofstad van Provence.
	Die hoofstad van Alpes-Maritimes is Nice.
ar	مات جون دبورانت بريفال في باريس.
	يتم إنتاج كيندل فاير بواسطة أمازون.
	اللغة الرسمية لـ كوسوفو هي الصربية.
az	Madrid və Sofiya əkiz şəhərlərdir.
	İbn ən-Nədim islam dinə bağlıdır.
	Baş qərargah Belavia Minsk -dədir.
be	Афіцыйнай мовай Фінляндская праваслаўная царква з'яўляецца фінская мова.
	Сталіцай Правінцыя Цзянсу з'яўляецца Нанкін.
	Артур Гардэн нарадзіўся ў Манчэстэр.
bg	Ник Филип е роден в Лондон.
	Наситена мазнина се състои от въглерод.
	Официалният език на Организация на обединените нации е руски език.
bn	মন্টানা আইডাহো এর সাথে সীমানা ভাগ করে দেয়।
	মৌরিতানিয়া মালি এর সাথে সীমানা ভাগ করে দেয়।
	থার্ডম এবং কায়রো হ'ল দুটি শহর।
ca	Rich Gannon juga en la posició quarterback.
	La llengua oficial de Nastola és finès.
	Nova York i Jerusalem són ciutats bessones.
ceb	Ang Mario Aldo Montano usa ka lungsuranon sa Italya.
	Ang opisyal nga sinultian sa Belhika mao ang Inolandes.
	Ang Praga ug Berlin mga kaluha nga lungsod.
cs	Maháráďža je právní termín v Indie.
	Felix Magath hraje v poloze záložník.
	Philipp Eduard Anton von Lenard pracuje v oblasti fyzika.
cy	Mae Maes Awyr Rhyngwladol Des Moines wedi'i leoli yn Iowa.
	Iaith swyddogol Aragón yw Sbaeneg.
	Enwir Flins-sur-Seine ar ôl Afon Seine.
da	Dylan Taite blev født i Liverpool.
	Johann Andreas Schmeller døde i München.
	Hovedstaden i Yolo County er Davis.
de	The Bill wurde in Englisch geschrieben.
	Buenaventura Sitjar starb in Kalifornien.
	Mai Jones funktioniert für British Broadcasting Corporation.
el	Ρεϊνάλντο Αν παίζει μουσική όπερα.
	Το Μπόσνα Σαράγεβο βρίσκεται στο Σαράγεβο.
	Το Μεταδιθειώδες καλιο αποτελείται από θείο.
en	Vienna bread is a subclass of bread .
	Stevie Wonder is represented by music label Motown .
	Kinji Fukasaku is Japan citizen .
es	Luxemburgo comparte frontera con Francia.
	Charles Schreiber nació en Colchester.
	Polonia comparte frontera con Alemania.
et	Kohtumõistjate raamat on osa Piibel -st.
	Iisrael hoiab diplomaatilisi suhteid Jordaania -ga.
	Sambia jagab piiri Angola -ga.
eu	Amerikетako Estatu Batuak -ek Albania -ekin harreman diplomatikoak mantentzen ditu.
	Tajikistango presidente legezko terminoa da Tajikistan -n.
	Carla Bruni Paris -n lan egiteko erabiltzen zen.
fa	استان آنتورپ به آنتورپ نامگذاری شده است.
	سوزوکی گِرَنشی توسط سوزوکی تولید می شود.
	رئیسجمهور زیر کلاس سیاستمدار است.
fi	Pääoman Genovan maakunta pääoma on Genova.
	Edward Joseph Kelly syntyi Chicago.
	The Home Depot perustettiin Atlanta.
fr	Mel Charles est pays de Galles citoyen.
	Audi A5 est produit par Audi.
	Honda XR est produit par Honda.
ga	Fuair Walter Gay bás i Páras.
	Tá Ollscoil Concordia suite i Montréal.
	Is cúpla cathair iad Vín agus An Bhrataslàiv.
gl	Hannover e Bristol son cidades xemelgas.
	O idioma oficial de Mercia é Lingua latina.
	Afloramento de algas está composto por Alga.
he	אוניברסיטת וושינגטון נמצא ב- סיאטל.
	ויקיפדיה הצ'כית נכתב ב- צ'כית.
	תיירות מין היא תת-סוג של תיירות.
hi	कीनिया युगाण्डा के साथ राजनयिक संबंध बनाए रखता है।
	कावेरी नदी एशिया में स्थित है।
	फ़िलीपीन्स का ध्वज फ़िलीपीन्स में एक कानूनी शब्द है।
hr	Istanbul i Bukurešt su gradovi blizanci.
	Gro Harlem Brundtland se nekada radila u Oslo.
	Izvorni jezik Ćelava pjevačica je francuski jezik.
hu	Gilles Grimandi Gap -ben született.
	Edo-kor elnevezése Edo.
	Joseph-Marie, comte Portalis anyanyelve francia.
hy	Իրան գտնվում է Ասիա –ում:
	Բարսելոնա և Դուբլին զույգ քաղաքներ են:
	Աղրբեջան -ի մայրաքաղաքն է Բաքու:

Figure 4: Three randomly sampled data entries from mLAMA per language. Due to the automatic generation of the dataset not all of them are fully correct.

[2] https://meta.wikimedia.org/wiki/List_of_Wikipedias

	en	de	nl	it
P495: “[X] was created in [Y]”	Japan (170), Italy (56)	Deutschland (217), Japan (70)	Nederland (172), Italië (50)	Italia (167), Giappone (92)
P101: “[X] works in the field of [Y]”	art (205), science (135)	Kunst (384), Film (64)	psychologie (263), kunst (120)	fisiologia (168), caccia (135)
P106: “[X] is [Y] by profession”	politician (423), composer (80)	Politiker (323), Journalist (128)	politicus (339), acteur (247)	giornalista (420), giurista (257)
P1001: “[X] is a legal term in [Y]”	India (12), Germany (11)	Deutschland (36), Russland (9)	Nederland (22), België (12)	Italia (31), Germania (16)
P39: “[X] has the position of [Y]”	bishop (468), God (68)	WW (261), Ratsherr (108)	burgemeester (400), bisschop (276)	pastore (289), papa (138)
P527 “[X] consists of [Y]”	sodium (125), carbon (88)	Wasserstof (398), C (49)	vet (216), aluminium (130)	calcio (165), atomo (96)
P1303 “[X] plays [Y]”	guitar (431), piano (165)	Gitarre (312), Klavier (204)	piano (581), harp (42)	arpa (188), pianoforte (139)
P178 “[X] is developed by [Y]”	Microsoft (177), IBM (55)	Microsoft (153), Apple (99)	Microsoft (200), Nintendo (69)	Microsoft (217), Apple (49)
P264 “[X] is represented by music label [Y]”	EMI (267), Swan (32)	EMI (202), Paramount Records (59)	EMI (225), Swan (50)	EMI (217), Swan (99)
P463 “[X] is a member of [Y]”	FIFA (126), NATO (33)	FIFA (118), NATO (38)	FIFA (157), WWE (16)	FIFA (121), NATO (36)

Table 4: Most frequent object predictions (TyQ) in different languages. Some relations exhibit language specific biases. WW = “Wirtschaftswissenschaftler”.

	Landskrona BoIS terletak di Swedia.
id	Roy Hargrove memainkan Trompet.
	Bahasa resmi Ossetia Selatan adalah Rusia.
	Federazione calcistica di Vanuatu è un membro di FIFA.
it	Mikhail Gromov funziona nel campo di geometria.
	letteratura fantasy è una sottoclasse di fantasy.
	ボルドーとカサブランカは双子の都市です。
ja	アーロン・マレーはクォーターバック位置で再生されます。
	エンツォ・フェラーリはイタリア市民です。
	ვიტამინი F შედგება ნაჯერი ცხიმ.
ka	ანრი რუსო გარდაიცვალა პარიზი -ში.
	კატალონია მდებარეობს ესპანეთი -ში.
	바하칼리포르니아주는 캘리포니아주와 (과) 국경을 공유합니다.
ko	레오 10세의 위치는 교황입니다.
	캐나다는 이탈리아와의 외교 관계를 유지합니다.
	Prophetia Michaeae est pars Biblia.
la	Carentonium Cum shares terminus Lutetia.
	Carolus Bildt in communicate ad lingua Suecica.
	Pensilvanija dalijasi riba su Merilandas.
lt	Viskonsinas sostinė yra Madisonas.
	Park Chan-wook naudojamas bendrauti Korėjiečių kalba.
	Altaja Republika oficiālā valoda ir krievu valoda.
lv	Francs Kafka dzimtā valoda ir vācu valoda.
	Audi Allroad Quattro ražo Audi.
	Tour de France dinamakan Perancis.
ms	Goa berkongsi sempadan dengan Maharashtra.
	Modal Amerika British ialah London.
	allmennaksjeselskap is een juridische term in Noorwegen.
nl	San Marinese voetbalbond is lid van FIFA.
	De oorspronkelijke taal van Nouvelle Star is Frans.
	Armand Marrast pracował w Paryż.
pl	Irlandia nosi imię Irlandia.
	Oficjalnym językiem Kanton Jura jest język francuski.
	Denver e Nairóbi são cidades gêmeas.
pt	Arábia Saudita mantém relações diplomáticas com México.
	Nyepi está localizado em Bali.
	Districtul Darnah este localizat în Libia.
ro	Sediul central al Toyota este în Toyota.
	Roy Orbison folosit pentru a comunica în limba engleză.
	Венанций Фортунат имеет позицию епископ.
ru	Renault 21 производится Renault.
	Пиччотто, Ги играет гитара.
	Austrália udržiava diplomatické vzťahy s Nórsko.
sk	Alanín pozostáva z dusík.
	Optický ďalekohľad je podtrieda teleskop.
	Fergus Morton se je rodil v Glasgow.
sl	Nemčija je članica NATO.
	Cenk Renda se je rodil v Turčija.
	Nic Chagall interpreton Trance muzikë.
sq	Marie Liljedahl është Suedia qytetar.
	Pallati i Fontainebleau është në pronësi të Francë.
	Нортхемптоншир дели границу са Бакингемшир.
sr	периапсис је део орбита.
	Патрик Дотерс је рођен у Беркли.
	Det officiella språket för Savukoski är finska.
sv	The Upsetters spelar reggae musik.
	Arakidonsyra består av kol.
	வெனிசுவேலா செருமனி உடன் இராஜதந்திர உறவுகளைப் பேணுகிறது.
ta	**தையால் கந்தகம் ஐக் கொண்டுள்ளது.**
	எக்ச.கோட் ஆப்பிள் நிறுவனம் ஆல் உருவாக்கப்பட்டது.
	ประเทศซูดาน อยู่ใน ทวีปแอฟริกา
th	ภาษาราชการของ วัฒโอโอวา คือ ภาษาอังกฤษ
	กรุงเทพมหานคร เป็นเมืองหลวงของ ประเทศไทย
	Markus Feldmann Bern 'da çalışırdı.
tr	Graduate Institute of International and Development Studies şirketinin genel merkezi Cenevre dedir.
	Afganistan Demokratik Cumhuriyeti 'un başkenti Kâbil' dir.
	Столиця Сирія - Дамаск.
uk	Комплекс Наполеона названий на честь Наполеон I Бонапарт.
	Нижня Канада ділиться межею з Вермонт.
	مملکت سربیا کا دارالحکومت بلغراد ہے۔
ur	میکسیکو فوس فٹ بال ٹیم فیفا کا ممبر ہے۔
	یورپی اتحاد بیلجیم کے ساتھ سفارتی تعلقات برقرار رکھے ہوئے ہے۔
	Ngôn ngữ chính thức của Lampung là tiếng Indonesia.
vi	Ngôn ngữ chính thức của Viitasaari là tiếng Phần Lan.
	Á Rập Saudi duy trì quan hệ ngoại giao với Yemen.
	蒙马特圣伯多禄堂以西門彼得命名。
zh	Sun Media集團的总部位于多伦多中。
	多米尼克·杜卡管经在布拉格中工作。

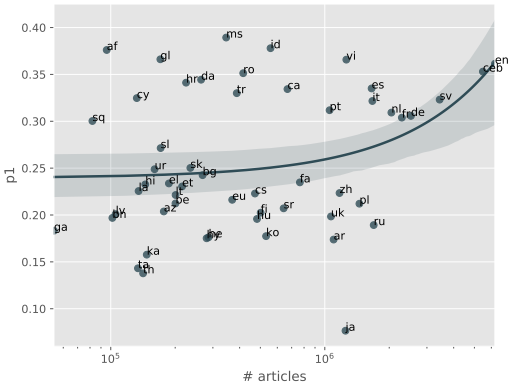


Figure 6: Scatter plot of p1 TyQ and number of articles in the corresponding Wikipedia. There is no clear trend visible.

Chapter 5

BeliefBank: Adding Memory to a Pre-Trained Language Model for a Systematic Notion of Belief

BeliefBank: Adding Memory to a Pre-Trained Language Model for a Systematic Notion of Belief

Nora Kassner[1,2], Oyvind Tafjord[1], Hinrich Schütze[2], Peter Clark[1]
[1]Allen Institute for AI, Seattle, WA
[2]Center for Information and Language Processing, LMU Munich, Germany
kassner@cis.lmu.de
{oyvindt,peterc}@allenai.org

Abstract

Although pretrained language models (PTLMs) contain significant amounts of world knowledge, they can still produce inconsistent answers to questions when probed, even after specialized training. As a result, it can be hard to identify what the model actually "believes" about the world, making it susceptible to inconsistent behavior and simple errors. Our goal is to reduce these problems. Our approach is to embed a PTLM in a broader system that also includes an evolving, symbolic memory of beliefs – a BeliefBank – that records but then may modify the raw PTLM answers. We describe two mechanisms to improve belief consistency in the overall system. First, a reasoning component – a weighted MaxSAT solver – revises beliefs that significantly clash with others. Second, a feedback component issues future queries to the PTLM using known beliefs as context. We show that, in a controlled experimental setting, these two mechanisms result in more consistent beliefs in the overall system, improving both the accuracy and consistency of its answers over time. This is significant as it is a first step towards PTLM-based architectures with a systematic notion of belief, enabling them to construct a more coherent picture of the world, and improve over time without model retraining.

1 Introduction

Intelligent agents are typically considered to have beliefs about the world – propositions that they take as true (Genin and Huber, 2021). In general, a system can be said to (appear to) believe a proposition p, e.g., "eagles are birds", if it produces answers consistent with p (and its other beliefs). Pragmatically, we expect the system to (a) give a consistent answer to different paraphrases of the question "p?" ("Are eagles birds?", "Is an eagle a type of bird?", ...), and (b) give correct answers about implications of p ("Eagles lay eggs", "Eagles have feathers", ...), conditional on its other knowledge and reasoning abilities.

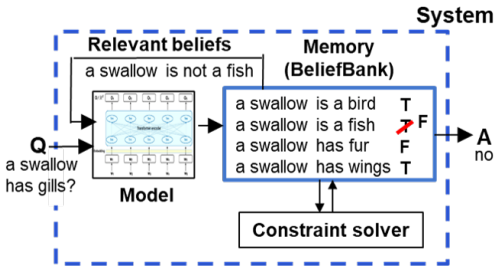


Figure 1: The proposed architecture. The model's raw answers are stored in a persistent, symbolic memory (BeliefBank), and two mechanisms attempt to improve them: (a) A **constraint solver** flips beliefs (e.g., the belief that "a swallow is a fish") that clash significantly with others. (b) A **feedback** mechanism poses new questions using existing, relevant beliefs (e.g., "a swallow is not a fish") as the query context. We find that both consistency and accuracy of the overall system improve. Example: The model M shown in the figure incorrectly answers "yes", when asked "a swallow has gills?". But (as shown above) if reminded of its previous answer "a swallow is not a fish", M correctly answers "no".

Maintaining a consistent set of beliefs (a "belief system") is a key facet of intelligence, as it can help debug errors and encourage rational behavior. However, although PTLMs contain substantial world knowledge (Petroni et al., 2019), their answers to probing questions can be inconsistent (Elazar et al., 2021; Kassner and Schütze, 2020), even after specialized training to reduce inconsistency (Ribeiro et al., 2019; Li et al., 2019). As a result, it is sometimes hard to pin down what a PTLM actually "believes", making them susceptible to inconsistent and/or irrational behavior. Our goal is a first step to avoid these problems by embedding a PTLM in a broader system with a clearer notion of belief (see Figure 1).

Prior work in AI, including in formal logic (Genesereth and Nilsson, 1987), belief mainte-

nance (De Kleer, 1986; Dechter and Dechter, 1988), and uncertainty (Pearl, 1986), offers models for how beliefs can be managed. Most importantly, it posits that creating a coherent set of beliefs – a kind of "mental model" of the world (Johnson-Laird, 1983) - is a constructive process requiring explicit representation of beliefs, and inference about their dependencies. Based on this, our approach is to embed a PTLM in a broader system with an evolving, symbolic memory of beliefs - a BeliefBank - along with two mechanisms to improve belief consistency of the overall system. First a reasoning component – a weighted MaxSAT (satisfiability) solver – reasons about belief dependencies and revises beliefs that significantly clash with others. Second, a feedback component poses future queries to the model using known beliefs as context, aiming for more accurate and consistent answers from the PTLM itself. The BeliefBank represents the overall system's beliefs (a "mental model") about the world, constructed by deliberating over the noisy output of a raw PTLM.

We explore this in a controlled experimental setting where both candidate beliefs and constraints between them are provided. Candidate facts are simple sentences that may be true or false, e.g., "An eagle is a bird" (T), "An eagle is a mammal" (F). Constraints are between (variabilized) facts, e.g., "X is a bird $\rightarrow$ X has wings". These allow us to both probe and measure improvement in the consistency and accuracy of a system's beliefs, compared with an original PTLM.

In contrast to prior work, this system does not rely on fine-tuning the PTLM. Fine-tuning requires expensive training data, and risks destabilizing the model's performance on other tasks outside the scope of training. Instead, our system functions without training data, explicitly reasoning about beliefs using an external mechanism, thus allowing both controllability and interpretability. Most significantly, we find that improving consistency in this way improves accuracy, while earlier fine-tuning-based approaches report either no accuracy gains (Ribeiro et al., 2019; Minervini and Riedel, 2018; Li et al., 2019) or only slight gains (Asai and Hajishirzi, 2020).

We make the following contributions:

1. We show that a PTLM-based system can be given a consistent notion of belief by augmenting the PTLM with a global memory – the BeliefBank – that can be deliberated over. Specifically, we show that two mechanisms – constraint reasoning and feedback – improve the overall system's accuracy and consistency over time.
2. We contribute a targeted dataset to measure a system's consistency against given constraints.
3. We provide an analysis of the failure modes and directions for future work.

This work is significant as it is a first step towards PTLM-based architectures that have a systematic notion of belief, allowing them to construct a more coherent picture of the world, and improve over time without model retraining.[1]

2 Related work

The idea that agents should have a belief system dates back to the earliest years of AI, e.g., McCarthy (1959) envisioned representing a system's beliefs as formal propositions along with a reasoning process to identify what follows. Multiple subfields of AI have explored ways of representing and updating beliefs, including in formal logic (Genesereth and Nilsson, 1987; Moore, 1983), belief revision (De Kleer, 1986; Dechter and Dechter, 1988), and uncertainty (Pearl, 1986). Similarly, work in cognitive science has promoted mental models – coherent, constructed representations of the way the world is believed to be – as central to understanding and communication (Johnson-Laird, 1983; Gentner and Stevens, 1983; Hilton, 1996). We draw on these ideas, proposing how they can be layered on top of PTLMs, here representing beliefs as NL statements rather than formal structures.

Although PTLMs contain extensive world knowledge (Petroni et al., 2019; Roberts et al., 2020), they can be inconsistent in their answers to probing questions (Ettinger, 2020; Davison et al., 2019; Ravichander et al., 2020; Elazar et al., 2021; Subramanian et al., 2020), making their "world model" unclear. Although various approaches have improved answer consistency, mainly through modified model training, e.g., (Ribeiro et al., 2019; Minervini and Riedel, 2018; Li et al., 2019; Asai and Hajishirzi, 2020), they have not solved the problem. Current PTLMs still behave as a source of noisy knowledge, rather than projecting a coherent picture of the world (Kassner and Schütze, 2020).

A close analogy to our task is in knowledge graph (KG) construction. Pujara et al. (2013)

[1] Dataset is available at https://allenai.org/data/beliefbank

define "knowledge graph identification" as the task of building a maximally consistent KG given noisy candidate facts and their extraction confidences, and constraints between them. They develop a solution using probabilistic soft logic (PSL) (Broecheler et al., 2010) as their constraint reasoner. Our reasoning component follows similar ideas, but applied to the noisy predictions of a PTLM. On the face of it, it is not clear how to plug a constraint solver into a PTLM, given their very different natures. Our solution introduces a global persistent memory, making this novel combination of technologies possible. To our knowledge this has not been done before.

Our work presents a broader system architecture in which a PTLM is embedded, along with a dynamic, persistent memory. While there are prior neural architectures that include an associated memory, e.g., (Henaff et al., 2016; Sukhbaatar et al., 2015; Graves et al., 2016), these components typically play the role of a short-term working memory to help computation. In contrast, our BeliefBank is a persistent, long-term memory, and we treat the PTLM as a component within a larger system. Our work also differs from retrieval-augmented architectures, e.g., RAG (Lewis et al., 2020), REALM (Guu et al., 2020), that augment model input with external retrievals. Rather, our memory is *reflective*, built from model *outputs* and reasoned over.

Our feedback mechanism uses old answers to help answer new questions. This builds on prior work such as Self-Talk (Shwartz et al., 2020), where a model asks itself related questions to help with new answers, and can be seen as a form of dynamic prompt engineering (Liu et al., 2021). In our case, feedback is selected from a global BeliefBank, rather than generated with templated subqueries, potentially allowing more control over feedback selection.

Finally, our system performs a kind of continual learning (Parisi et al., 2019; Carlson et al., 2010). While recent work in this area has focused on dynamic update of model parameters, e.g., (Ebrahimi et al., 2021), our work leaves the model fixed, and seeks improvement in the broader system in which the model is embedded, exploring an alternative and potentially more interpretable architecture towards this goal.

3 Task

Our goal is to ascribe a clearer notion of "belief" to a system that includes a model M, by improving, over time, the consistency and accuracy of its answers (compared with M) to a stream of questions. We measure this with true/false probing, where we are also given a set of constraints between answers:

Given:

- a stream of **sentences** Q, interpreted as questions to the model
- a set of **constraints** $C(S)$ between (the truth values of) sentences in Q, each annotated with a weight w_i (A penalty w_i is applied if $c_i \in C(S)$ is violated)
- a **model** M that takes as input a question $q \in Q$ and optionally an (NL) context X (consisting of answers to previously posed questions), and predicts a True/False answer A with confidence score F

Accumulate:

- the True/False labels for Q predicted by M, optionally corrected by the constraint solver, so as to maximally improve accuracy (with respect to gold labels) and consistency (minimize total penalties of constraint violations)

4 Approach

Our approach adds a memory layer, called the **BeliefBank**, on top of the model to globally track beliefs. Two mechanisms are then used to manage the BeliefBank beliefs, namely constraint reasoning and feedback, as we now describe.

4.1 Definitions

Let

- a **belief** b_i be a triple (s_i,l_i,w_i), where
  - s_i is a sentence $\in S$
  - label $l_i \in$ {T,F} denotes the system's True/False belief about the truth of s_i[2]
  - weight w_i is a number $\in [0, 1]$ representing the system's strength of that belief

  For example:

  ("a poodle is a dog", T, 0.9)

  denotes the belief (strength 0.9) that "a poodle is a dog" is a true statement (T).

[2]Strictly, the label F denotes the belief that the negation of s_i is true, e.g., ("a poodle is a bird",F,...) denotes the belief "a poodle is not a bird".

- a **BeliefBank** $B(S)$ = a set of beliefs over sentences $S = s_1, ..., s_n$
- a **constraint** c_i = a 5-tuple of the form $(s_i.l_i \rightarrow s_j.l_j, w_i)$ where
  - s_i, s_j are sentences $\in S$,
  - $l_i, l_j \in$ {T,F}. If s_i has truth value l_i, denoted $s_i.l_i$, then s_j is expected to have truth value l_j, denoted $s_j.l_j$.
  - w_i denotes the strength of that expectation (a penalty w_i is applied if violated).

  For convenience, a shared variable X can be used in s_i, s_j, allowing a set of grounded constraints to be expressed in one statement, e.g.,
  *("X is a dog".*T $\rightarrow$ *"X has a tail".*T, 0.8)
  expresses that if something is a dog, then it should (T) have a tail, with a penalty of 0.8 applied if it does not. Mutual exclusivity is expressed using two rules, e.g., that fish and birds are mutually exclusive is expressed:
  *("X is a bird".*T $\rightarrow$ *"X is a fish".*F, 1.0)
  *("X is a fish".*T $\rightarrow$ *"X is a bird".*F, 1.0)
  where "F" indicates the conclusion should be *false* if the condition here is true (T).
- a **constraint graph** $C(S)$ = a set of constraints c_i over sentences S

Given a set of beliefs $B(S)$ about S and a set of constraints $C(S)$, we measure **consistency** using (the complement of) Li et al. (2019)'s conditional constraint violation (τ) metric, namely the fraction of constraints whose *condition* $s_i.l_i$ is believed, but whose *conclusion* (that s_j has truth value l_j) is not. In other words, over all constraints $c_i \in C(S)$, inconsistency τ is

$$\tau = |\{\ c_i \mid \neg(s_i.l_i \rightarrow s_j.l_j)\ \}|\ /\ |\{\ c_i \mid s_i.l_i\ \}|$$

i.e., the size of the set of *violated* constraints ($s_i.l_i \rightarrow s_j.s_j$ is false) divided by the size of the set of *applicable* constraints. We then define:

$$\text{consistency} = 1 - \tau$$

4.2 Methods

We consider our system in a dynamic setting, where it receives a stream of questions and gradually builds up a BeliefBank of answers (including revising earlier answers). We evaluate two methods for improving the BeliefBank's accuracy and consistency over time:

Constraint solving: Given a model M's raw answers (with confidences), a constraint solver seeks to reduce constraint violations by potentially flipping answers that maximally clash with other answers.

Feedback: Given a new question q, selected beliefs in the BeliefBank are provided as context to M to help it answer q correctly.

Figure 1 shows these components.

4.2.1 Constraint Solving

Given a set of beliefs and constraints, the constraint solver has two competing objectives: (a) flip beliefs so as to minimize constraint violations (b) don't flip beliefs, so as to preserve the model's raw answers, i.e., minimize conflict between the model and BeliefBank. To implement this tradeoff, the model's answers are themselves treated as just another constraint, e.g., the answer that "a poodle is a dog" is true (confidence 0.9) is treated as a constraint "a poodle is a dog", with penalty 0.9 if it is violated. To balance the two objectives (a) and (b), the model confidences are scaled by a learned hyper-parameter λ, trained on a calibration part of our dataset, disjoint from the data then used in experiments (Section 5).

To implement constraint solving, we translate the task into a *weighted MaxSAT* (satisfiability) problem P, for which efficient algorithms with guarantees exist. Each belief becomes a weighted assertion in P, e.g., the belief ("a poodle is a dog", T, 0.9) is expressed in SAT syntax:

0.9 "a poodle is a dog"[3]

while the constraint ("a poodle is a dog".T $\rightarrow$ "a poodle has a tail".T, 0.8) is expressed:

0.8 "a poodle has a tail" **-***"a poodle is a dog"*

(literally: *"a poodle has a tail"* OR NOT ("**-**") *"a poodle is a dog"*). We then apply the solver Z3 (De Moura and Bjørner, 2008) to P, which outputs a set of truth assignments for all individual sentences in P so as to minimize the weighted sum of violations. If the truth of any sentence has changed, the BeliefBank is correspondingly updated.

4.2.2 Feedback

Feedback involves asking the model a question, but with the benefit of knowing answers to prior, related questions. To use these answers in the query, selected beliefs are added to the query context before asking the model. (Note that the selected beliefs are not guaranteed to be correct, of course). Our conjecture is that if the model is explicitly reminded of relevant beliefs when answering a new question, it will answer the question more accu-

[3] In practice, strings are replaced with numeric identifiers in SAT syntax, but for clarity we leave them as strings here.

rately and consistently. For example, in Figure 1, when asked "Do swallows have gills?", our model M incorrectly answers "yes". But if reminded that swallows are not fish, by asking: "CONTEXT Swallows are not fish. QUERY Do swallows have gills?" the model now correctly answers "no".

We evaluate two policies for choosing which beliefs to feed back to M when asking question q about entity e:

1. **on topic** beliefs, namely current beliefs about entity e, randomly selected from the BeliefBank
2. most **relevant** on topic beliefs (i.e., again about e), using the constraint graph to identify relevance. As the constraint graph captures potential clashes that the answer to q could cause, we use the graph to identify beliefs that would be most affected by that answer. For example, if the current query is: "Is a poodle an animal?", the constraint graph identifies potential clashes that would occur if the model answered "yes", and also clashes if it answered "no". Here, if the model answered "no", the resulting belief ("a poodle is *not* an animal") would strongly clash with other beliefs "A poodle is a dog." and "A poodle is a mammal.", so these two are strong candidates for the context. We select the three strongest clashing beliefs found in this way, considering both "yes" and "no" answers to q. If no relevant fact is present, we use a randomly selected topic belief instead.

In both cases, three beliefs are selected, this number was empirically found to be most effective.

5 Dataset

We create a dataset[4] to test our approach in a controlled way, allowing us to perform systematic experiments to evaluate behavior. The dataset contains two parts, *constraints* and *facts*, defined over simple sentences such as "a swallow is a bird."

5.1 Constraints

The dataset contains two kinds of constraints:

positive implications: conclusion truth value l_j = T (true), e.g.,
"X is a dog.T → "X has a tail.".T

mutual exclusivities: expressed as a pair of constraints with l_j = F (false), e.g.,
"X is a dog".T → "X is a bird.".F
"X is a bird".T → "X is a dog.".F
expresses that an entity cannot be both a dog and a bird at the same time.

Positive implications were manually gathered from ConceptNet (Speer et al., 2017). First, we identified 121 general concepts of interest, e.g., "mammal", then converted selected triples about them to constraints (Details of the selection process are in Appendix A). For example, the ConceptNet triple (dog,HasA,tail) becomes the constraint "X is a dog" → "X has a tail". We also add weaker, disjunctive constraints in the backward direction, e.g., "X has a tail" → "X is a dog" *OR* "X is a cat" *OR* .... for all entities with tails. Mutual exclusivities were gathered from the "isa" taxonomies in ConceptNet and WordNet (Fellbaum, 2005), using the approximation that siblings in the noun hierarchy are mutually exclusive. Thus, for any pair of siblings, we add a mutual exclusivity constraint (using two constraint rules).

We collected 2612 constraints in this fashion (1836 forward implications, 2*388 bidirectional mutual exclusivities).

5.2 Constraint Weights

Constraint weights need to be set appropriately to mix well with the model's confidences inside the weighted SAT solver. We use a development set of 1072 facts about seven entities to set one constraint weight for the forward direction of the implications and the mutual exclusivity rules and a second one for the backward direction of the implications. To do this we perform a grid search over these parameters, finding the values that result in the highest F1 (accuracy) after running the constraint solver over the raw model's beliefs about these facts.

5.3 Facts

We also collect a set of truth-labeled facts about different entities, relevant to the constraints. To do this, we select a new entity, e.g., "poodle", that is a member of one of our general concepts, e.g., "dog", then instantiate the constraint graph with that entity (i.e., set X = "poodle"). We then identify the leaf (source) nodes of that graph, just considering forward implication rules, i.e., finding facts not implied by other facts in the graph, and manually annotate their True/False labels. We then use the implications and mutual exclusivities to infer other True/False labels for other sentences, i.e., we propagate the annotated labels through the graph. This

[4] Dataset is available at https://allenai.org/data/beliefbank

provides "silver" labels for sentences reachable in this way (a subset of all the sentences in the graph) – silver because the implications are soft, hence not guaranteed to hold for all entities.

We repeat this for 85 entities (animals and plants), resulting in a final dataset containing 12,525 "silver" facts (sentences + True/False labels). Note that this data is purely for evaluation. There is no training phase or training data. The system does not have access to any labeled data besides the constraint rules.

6 Model

The fixed model M that we use for our experiments is Macaw (Tafjord and Clark, 2021), a state-of-the-art T5 QA model fine-tuned on ≈400k QA pairs. To query the model, we pose the query (optionally with a textual context), and let the model choose between the two answer options "yes" and "no". The model also outputs an answer confidence, used as the belief weight.

We use the T5-large version of this model. Note that we do not retrain the model for this work; rather, it is used as a black-box QA module in the broader system (Figure 1). Other models could equally have been used.

7 Experiments

We evaluate our system in a dynamic setting in which it receives a stream of questions, building up and revising a BeliefBank. To simplify the evaluation, we consider questions to arrive in *batches*, and evaluate the BeliefBank after each batch, measuring accuracy (F1)[5] and consistency (1-τ, Section 4.1) of the BeliefBank so far, comparing with the gold labels. We evaluate four configurations:

Raw model: The BeliefBank simply records the raw model's answers [6]

Constraint-Solving: After each batch, the constraint solver is run over all the (raw) model answers so far, and the BeliefBank updated accordingly.

Feedback: Questions in batch n are posed to the model using a context selected from the beliefs already in the BeliefBank (batches 1 to $n-1$). We evaluate two selection strategies:

Feedback (on-topic): Random beliefs about the entity e being queried about

Feedback (relevant): On-topic beliefs (i.e., again about e) that are most relevant to the query, as defined in Section 4.2.2

Feedback + Constraint-Solving: A combination of the two.

These configurations are illustrated in Figure 2.

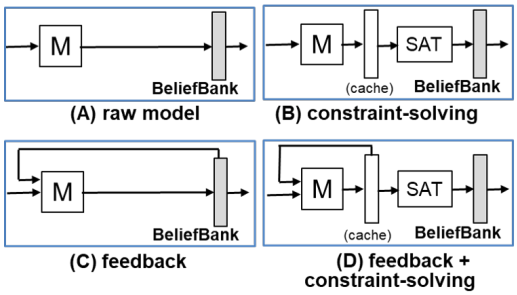


Figure 2: The four configurations we evaluate. In (B), the contraint-solver (SAT solver) is run over all model M answers so far. In (C), current beliefs are fed back as context for new questions. (D) combines the two.

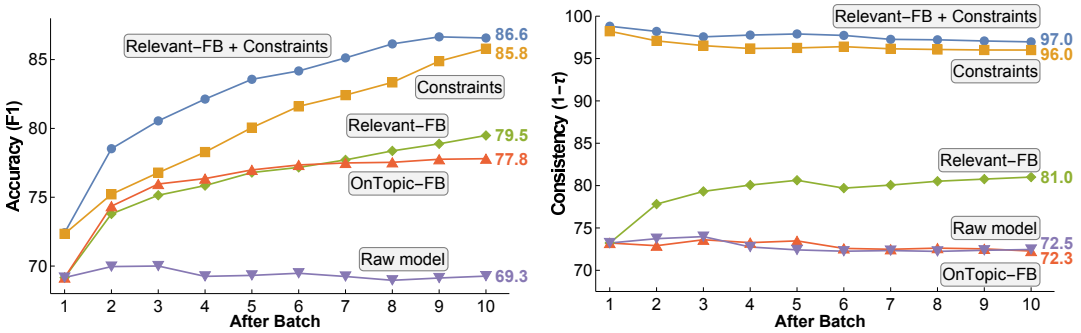
7.1 Results

The results are shown in Figure 3, showing the changing accuracy and consistency of the growing BeliefBank with time, for different configurations. Each time-step (batch) represents another 10% of the test questions being posed to the system. (The same data is presented in tabular form in Appendix B). Several conclusions can be drawn:

- Use of feedback, constraint-checking, or both, all result in a **continually improving accuracy over time**. This is a significant result, showing a larger system can continually improve even if its internal PTLM component is fixed. (The raw accuracy of the PTLM itself is necessarily constant).
- **Use of the constraint-solver results in very high (~95%) consistency,** indicating that it is doing its job well, and **also improving accuracy** substantially (+17% over the raw model). The constraint-solver has a global view of the BeliefBank, and thus can balance all beliefs seen so far with the provided constraints to make a decision.
- **Relevant feedback results in significant consistency gains** compared with just on-topic feedback. As relevant beliefs are exactly those that may clash with the answer to the current question (Section 4.2.2), this encourages the model to

[5] We measure accuracy with F1 (on the True class) rather than % correct because the True/False distribution in our dataset is unbalanced, with significantly fewer True than False answers. F1 avoids scores being dominated by negative answers.

[6] To the best of our knowledge there are no other baseline models to compare to as consistency based Q&A does not go beyond paraphrases and relies on finetuning (Elazar et al., 2021).



OnTopic-FB = using (randomly selected) **on-topic** feedback from old answers for new queries.
Relevant-FB = using most **relevant** on-topic feedback for new queries.
Constraints = running the constraint-solver after each batch.

Figure 3: Accuracy (left) and consistency (right) of the growing BeliefBank, as the system answers incrementally more questions (each batch = 10% of the queries). Relevant feedback, constraint-solving, and both, all help improve both F1 and Consistency.

answer consistently with those beliefs, promoting consistency. Of course, this could hurt accuracy if those relevant beliefs were wrong, amplifying the errors. In fact, the overall accuracy remains about the same as with on-topic feedback, and significantly better than the model's raw answers.

- The **greatest gains are for feedback and constraint-solving combined**, resulting in +18% F1 (absolute) over the raw model accuracy. This suggests that feedback and constraints can work together in a positive way.

- As a sanity check we also tried using random beliefs about *other* entities (off-topic) as feedback, but (as expected) the results did not significantly differ from no feedback (raw model), ending at ≈72% F1 and ≈74% consistency after batch 10.

We also evaluated a **non-incremental**, "omniscient" version of the BeliefBank: Given the raw model answers to *all* questions, re-ask every question using feedback selected (using relevance) from all the other answers. The resulting accuracy was 74.5%, substantially lower than for the on-topic incremental approaches. Interestingly, this approach's built-in advantage (that every question has access to answers for *all* other questions) does not outweigh the built-in disadvantage (that those are the raw, rather than incrementally corrected, answers). This is a significant result demonstrating that **the positive feedback loop of the incremental approaches can be advantageous**, where feedback feeds more accurate beliefs into the BeliefBank, improving future feedback, etc.

7.2 Failure Analysis

We now provide some examples of good and bad flips to better understand the behavior of the model.

First, as an **illustration of desired behavior**, the raw model incorrectly believes that a pine is both a plant (correct) and a vertebrate (incorrect), when queried. However, this violates a mutual exclusivity rule, so the constraint-solver considers flipping one of these. Flipping "pine is a plant" from T to F would result in numerous other violations, e.g., "pine is a tree" (which the model also believes) would be violated. As a result, it prefers to (correctly) disbelieve "pine is a vertebrate", improving both accuracy and consistency.

From an analysis of the data, we see that the **majority of the raw model errors are false positives** – the raw model generally answers (almost) all the positive facts correctly (recall is ≈98%), but mistakenly thinks many negative facts are also true (precision is ≈54%). These false positives can be rather unusual facts, e.g., "A poodle is a bathroom." (model's answer: True). It is unsurprising that the model knows most of the positive facts, as they are simple statements about common entities ("eagles can fly"), likely seen in pre-training. However, the fact that the model makes (what a person would view as) catastrophic errors when asked more unusual questions, e.g., believing that "a poodle is plant", reveals that the PTLM's grasp of the world is still incomplete and problematic. The constraint mechanism proposed here essentially asks the model to think about its answers and their consequences, so that it can spot problems that the PTLM alone does not see, and repair them.

The constraint reasoner can also make mistakes, flipping things the wrong way so as to improve consistency, at the expense of accuracy. For example, the raw model correctly believes that "a rat is not a cat". However, the constraint solver then (incorrectly) flips this to "a rat is a cat", because multiple constraints weakly suggest rats are cats given other beliefs ("rats catch mice", "rats have tails",...), which together add up, causing the (incorrect) flip, including overwhelming the strong (but not infinitely strong) constraint that "a rat is not a feline." This illustrates that the **constraint mechanism is sensitive to the number of and weights on constraints**, even with automatic hyperparameter tuning (Section 5.2).

Similarly, **the feedback mechanism is sensitive to question order**, especially if the model's early answers are wrong, as the feedback mechanism causes the model to pay extra (sometime disproportionate) attention to earlier context (Kassner and Schütze, 2020). For example, the bad context "A poodle is not a mammal" (from an earlier bad answer) undesirably causes the model to change its answer for "A poodle is a dog" from true (raw model) to false.

Finally, we can only speculate why feedback improves results, in particular, since the feedback consists of facts that came from the model itself (i.e., that it already knows). One explanation is that **feedback may help the model focus attention** on important facts, e.g., reminding the model that "a swallow is not a fish" should help it realize that "a swallow has gills" is False (Figure 1). In addition, feedback may possibly help resolve some ambiguities, e.g., the feedback "a swallow has wings" helps identify the bird sense of "swallow". Similar advantageous use of feedback was observed in the SelfTalk experiments (Shwartz et al., 2020).

In future work, the feedback mechanism can be improved further by training it to respond more systematically to feedback (similar to (Clark et al., 2020)) and to better balance implicit and explicit knowledge (Talmor et al., 2020), ideally incorporating different levels of confidence.

8 Future Work

8.1 Human in the Loop

Although our system is autonomous, its incremental setting combined with the explicit representation of beliefs makes it amenable to a human in the loop. In this setting, a human might spot an egregious bad belief in the BeliefBank, and forcibly correct it. Then, ideally, this strong positive datapoint would *also* improve the model's accuracy on other beliefs, both in the BeliefBank and for future questions. As a brief test of this, we allowed a human to correct all bad beliefs (average 6) in the BeliefBank after just the first batch (10%) of questions, and then continued as before to completion, using the constraint-solving approach. We find that these limited interventions increased both the final F1 and Consistency each by 2% (absolute) *on top of* the gains produced by the corrected beliefs themselves. Although preliminary, this suggests that our architecture may have value in an interactive "machine teaching" setting, where the user is supervising and correcting the system, and it continually improves as a result (Zhu, 2015).

8.2 Towards Deployment

Although our work has been in a constrained setting (targeted set of relations, entities and constraints), there is a clear development path to deployment in real QA systems to reduce the kind of irrational behavior we have described, such as in this (real) transcript:

(1) Is oxygen colorless? yes
(2) What color is oxygen? blue
(3) What gas do plants produce? oxygen
(4) What color is the gas plants produce? green

The basic components of our architecture provide a framework to help avoid such irrationality. First, (declarative versions of) questions and model answers would be persistently stored in a BeliefBank. Second, on-topic feedback could be selected to help answer new questions using information retrieval over the BeliefBank. Third, given a source of constraints, e.g., a general rule of taxonomic inheritance,[7] constraint solving could be applied to spot and reduce clashes. This would require a mechanism to identify when a belief satisfies a constraint's condition or conclusion, e.g., a state-of-the-art textual entailment engine such as CA-MTL (Pilault et al., 2021). A variant of our system could also work without the distinction of model beliefs and constraints: Instead of providing constraints externally we could treat them as beliefs, e.g., query the model for mutual exclusivities "Can an entity be an animal and a plant?" or implications: "Do dogs have tails?" directly. This would run the risk

[7] I.e., that the properties of an entity type usually apply to all its subtypes also.

of adding extra noise, but would eliminate the manual effort involved in generating the constraint set, and therefore improve scalability. Together, such developments would pave the way to real-world QA systems that are more consistent and improve over time, rather than remain static.

8.3 The Broader Research Agenda

This work only touches on a broader research agenda, namely how to expand work on PTLMs to encompass the cognitive skills of world modeling and deliberative reasoning ("thinking, fast and slow" (Kahneman, 2011)). In this broader agenda, intelligence is not just about opaque question-answering, but also about constructing mental models that describe how (some aspect of) the world works (Gentner and Stevens, 1983). Although mental models are abstractions (hence are approximate), they add a powerful, systematic component to understanding that should expand its capabilities.

The BeliefBank can be seen as a simple illustration of this broader agenda. A wider pursuit would include a richer notion of a model, perhaps with more structure to model elements than just sentences; more sophisticated means of model construction than just accumulating and resolving answers; and the generation of explanations to convey the deliberative component's behavior, and ultimately interact with a user. Such mechanisms may be symbolic or neural in nature, e.g., (Talmor et al., 2020). Although these issues are beyond the scope of this paper, our work points to this interesting, larger goal for PTLM research, as well as offering a specific mechanism for belief consistency.

9 Conclusion

PTLMs can be inconsistent in their answers to probing questions, and can still give (what to a person appear as) naively wrong answers. This work is a first step towards alleviating these problems. By embedding a PTLM within a larger system with a persistent, global memory – the BeliefBank –, a constraint-solver and feedback mechanism, we have shown that the overall system's behavior is more coherent, both in terms of consistency and accuracy. The additional memory layer can loosely be seen as the system's "mental model", a representation constructed from the PTLM's raw answers.

Our experiments were conducted in a restricted (small set of relations, entities and constraints), controlled setting, and further development is needed to scale to larger and more complex tasks. Nevertheless, the work here is significant as it is a first step towards PTLM-based architectures with a globally consistent notion of belief, allowing them to construct a more coherent picture of the world, and continually improve with time.

Acknowledgements

This work has been funded by the Allen Institute for AI and the European Research Council (#740516) and by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibility for its content.

References

Akari Asai and Hannaneh Hajishirzi. 2020. Logic-guided data augmentation and regularization for consistent question answering. *ArXiv*, abs/2004.10157.

Matthias Broecheler, Lilyana Mihalkova, and L. Getoor. 2010. Probabilistic similarity logic. In *UAI*.

Andrew Carlson, J. Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka, and Tom Michael Mitchell. 2010. Toward an architecture for never-ending language learning. In *AAAI*.

P. Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. In *IJCAI*.

Joe Davison, Joshua Feldman, and Alexander Rush. 2019. Commonsense knowledge mining from pretrained models. In *EMNLP*.

Johan De Kleer. 1986. An assumption-based tms. *Artificial intelligence*, 28(2):127–162.

Leonardo De Moura and Nikolaj Bjørner. 2008. Z3: An efficient SMT solver. In *International conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer.

Rina Dechter and Avi Dechter. 1988. *Belief maintenance in dynamic constraint networks*. University of California, Computer Science Department.

Sayna Ebrahimi, Suzanne Petryk, Akash Gokul, William Gan, J. Gonzalez, Marcus Rohrbach, and Trevor Darrell. 2021. Remembering for the right reasons: Explanations reduce catastrophic forgetting. *ICLR*.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, E. Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *ArXiv*, abs/2102.01017.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Christiane Fellbaum. 2005. Wordnet and wordnets. In Keith et al. Brown, editor, *Encyclopedia of Language and Linguistics, Second Edition*. Elsevier, Oxford.

M. Genesereth and N. Nilsson. 1987. *Logical foundations of artificial intelligence*. Kaufmann.

Konstantin Genin and Franz Huber. 2021. Formal Representations of Belief. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*, spring 2021 edition. Metaphysics Research Lab, Stanford University.

Dedre Gentner and Albert L. Stevens. 1983. *Mental Models*. Lawrence Erlbaum Associates.

A. Graves, Greg Wayne, M. Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwinska, Sergio Gomez Colmenarejo, Edward Grefenstette, Tiago Ramalho, J. Agapiou, Adrià Puigdomènech Badia, K. Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, C. Summerfield, P. Blunsom, K. Kavukcuoglu, and D. Hassabis. 2016. Hybrid computing using a neural network with dynamic external memory. *Nature*, 538:471–476.

Kelvin Guu, Kenton Lee, Z. Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *ArXiv*, abs/2002.08909.

Mikael Henaff, J. Weston, Arthur D. Szlam, Antoine Bordes, and Y. LeCun. 2016. Tracking the world state with recurrent entity networks. In *ICLR*.

D. Hilton. 1996. Mental models and causal explanation: Judgements of probable cause and explanatory relevance. *Thinking & Reasoning*, 2:273–308.

P. Johnson-Laird. 1983. *Mental Models : Towards a Cognitive Science of Language*. Harvard University Press.

D. Kahneman. 2011. Thinking, fast and slow. Farrar, Straus, and Giroux.

Nora Kassner and H. Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *ACL*.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, V. Karpukhin, Naman Goyal, Heinrich Kuttler, M. Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *NeurIPS*.

Tao Li, Vivek Gupta, Maitrey Mehta, and Vivek Srikumar. 2019. A logic-driven framework for consistency of neural models. In *EMNLP*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, L. Carin, and Weizhu Chen. 2021. What makes good in-context examples for GPT-3? *ArXiv*, abs/2101.06804.

John W. McCarthy. 1959. Programs with common sense. In *Proc. Tedding Conf. on the Mechanization of Thought Processes*, pages 75–91.

Pasquale Minervini and S. Riedel. 2018. Adversarially regularising neural NLI models to integrate logical background knowledge. In *CoNLL*.

R. Moore. 1983. Semantical considerations on nonmonotonic logic. In *IJCAI*.

G. I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and S. Wermter. 2019. Continual lifelong learning with neural networks: A review. *Neural networks : the official journal of the International Neural Network Society*, 113:54–71.

J. Pearl. 1986. Fusion, propagation, and structuring in belief networks. *Artif. Intell.*, 29:241–288.

F. Petroni, Tim Rocktäschel, Patrick Lewis, A. Bakhtin, Yuxiang Wu, Alexander H. Miller, and S. Riedel. 2019. Language models as knowledge bases? In *EMNLP*.

Jonathan Pilault, Amine Elhattami, and C. Pal. 2021. Conditionally adaptive multi-task learning: Improving transfer learning in nlp using fewer parameters & less data. In *ICLR*.

J. Pujara, H. Miao, L. Getoor, and William W. Cohen. 2013. Knowledge graph identification. In *International Semantic Web Conference*.

Abhilasha Ravichander, Eduard Hovy, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2020. On the systematicity of probing contextualized word representations: The case of hypernymy in BERT. In *STARSEM*.

Marco Tulio Ribeiro, Carlos Guestrin, and Sameer Singh. 2019. Are red roses red? Evaluating consistency of question-answering models. In *ACL*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *EMNLP*.

Vered Shwartz, Peter West, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2020. Unsupervised commonsense question answering with self-talk. In *EMNLP*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI*.

Sanjay Subramanian, Ben Bogin, Nitish Gupta, Tomer Wolfson, Sameer Singh, Jonathan Berant, and Matt Gardner. 2020. Obtaining faithful interpretations from compositional neural networks. In *ACL*.

Sainbayar Sukhbaatar, Arthur D. Szlam, J. Weston, and R. Fergus. 2015. End-to-end memory networks. In *NeurIPS*.

Oyvind Tafjord and Peter Clark. 2021. General-purpose question-answering with Macaw. *arXiv*, abs/2109.02593.

Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. 2020. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge. In *NeurIPS*.

Xiaojin Zhu. 2015. Machine teaching: An inverse problem to machine learning and an approach toward optimal education. In *AAAI*.

Appendix: BeliefBank: Adding Memory to a Pre-Trained Language Model for a Systematic Notion of Belief

A Selecting Constraint Rules from ConceptNet

As described in Section 5.1, positive implication (constraint) rules were manually gathered from ConceptNet (Speer et al., 2017). First, we identified 121 general concepts of interest, e.g., "mammal", choosing concepts with high occurrence (> 100 times) in ConceptNet, avoiding significantly ambiguous terms (e.g., "bat"), and filtering out plurals and obscure concepts. For these entities, we then collected all ConceptNet facts involving 6 relations: IsA, HasA, MadeOf, PartOf, HasProperty, and CapableOf, and re-expressed them as constraints. For example, the ConceptNet triple (dog, HasA, tail) gives rise to the constraint "X is a dog" $\rightarrow$ "X has a tail." (Triples are converted into English sentences using simple templates). We then manually filter theses constraints for factual correctness. We also add weaker, disjunctive constraints in the backwards direction, e.g., "X has a tail" $\rightarrow$ "X is a dog" *OR* "X is a cat" *OR* .... for all entities with tails. (These backwards rules discourage the trivial solution that everything is false.) Finally, two hyperparameters for weights on forward and backwards rules are set by automatic calibration (Section 5.2).

B Experimental results in table form

Tables 1 and 2 contain the numerical data for the experimental results plotted in Figure 3.

Accuracy (F1) after batch →	1	2	3	4	5	6	7	8	9	10
Raw model	69.2	70.0	70.0	69.3	69.3	69.5	69.2	69.0	69.1	69.3
OnTopic-FB	69.2	74.3	76.0	76.4	77.0	77.3	77.5	77.5	77.8	77.8
Relevant-FB	69.2	73.8	75.1	75.8	76.8	77.2	77.7	78.4	78.9	79.5
Constraints	72.3	75.2	76.8	78.3	80.1	81.6	82.4	83.3	84.9	85.8
Relevant-FB + Constraints	72.4	78.5	80.5	82.1	83.6	84.2	85.1	86.1	86.7	86.6

Table 1: Experimental results for accuracy (F1), as plotted in Figure 3, here shown in tabular form.

Consistency ($1-\tau$) after batch →	1	2	3	4	5	6	7	8	9	10
Raw model	73.2	73.7	74.0	72.8	72.4	72.3	72.3	72.2	72.4	72.5
OnTopic-FB	73.2	72.9	73.6	73.3	73.5	72.6	72.5	72.6	72.5	72.3
Relevant-FB	73.2	77.8	79.3	80.1	80.6	79.7	80.1	80.5	80.8	81.0
Constraints	98.2	97.1	96.5	96.2	96.3	96.4	96.1	96.1	96.0	96.0
Relevant-FB + Constraints	98.8	98.2	97.6	97.8	97.9	97.7	97.3	97.2	97.1	97.0

Table 2: Experimental results for consistency ($1-\tau$), as plotted in Figure 3, here shown in tabular form.

Chapter 6

Language Models with Rationality

Language Models with Rationality

Nora Kassner[1,2] **Oyvind Tafjord**[1] **Ashish Sabharwal**[1] **Kyle Richardson**[1]
Hinrich Schütze[2] **Peter Clark**[1]

[1]Allen Institute for AI, Seattle, WA
[2]Center for Information and Language Processing, LMU Munich, Germany
kassner@cis.lmu.de
{oyvindt,ashishs,kyler,peterc}@allenai.org

Abstract

While large language models (LLMs) are proficient at question-answering (QA), it is not always clear how (or even if) an answer follows from their latent "beliefs". This lack of interpretability is a growing impediment to widespread use of LLMs. To address this, our goals are to make model beliefs and their inferential relationships explicit, and to resolve inconsistencies that may exist, so that answers are supported by interpretable chains of reasoning drawn from a consistent network of beliefs. Our approach, which we call REFLEX, is to add a **rational, self-reflecting layer** on top of the LLM. First, given a question, we construct a **belief graph** using a backward-chaining process to materialize relevant model beliefs (including beliefs about answer candidates) and their inferential relationships. Second, we identify and minimize contradictions in that graph using a formal constraint reasoner. We find that REFLEX significantly improves consistency (by 8%-11% absolute) without harming overall answer accuracy, resulting in answers supported by faithful chains of reasoning drawn from a more consistent belief system. This suggests a new style of system architecture in which an LLM extended with a rational layer can provide an interpretable window into system beliefs, add a systematic reasoning capability, and repair latent inconsistencies present in the LLM.

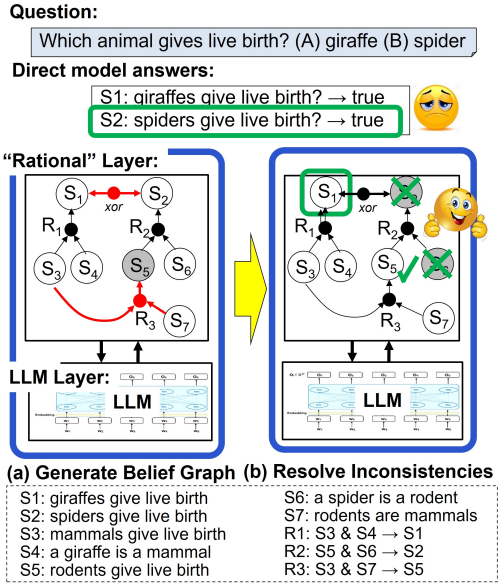


Figure 1: (**Top**) When queried about each answer option independently, the model incorrectly believes both are true, and is more confident in the wrong answer (S_2). (**Bottom**) REFLEX adds a "rational" layer above the LLM layer, in which a **belief graph** is constructed (by iteratively querying the LLM, up/down arrows), containing relevant model-believed facts (white/grey = believed T/F) and their inferential relationships. Inconsistencies are then identified (red) and minimized by a constraint reasoner that flips T/F labels on beliefs (green ✓/X), here resulting in the correct answer (S_1, green box) + explanation (graph) by the overall system (blue).

1 Introduction

While large language models (LLMs) are impressive at question-answering (QA), it is not always clear how (or even if) an answer follows from their latent "beliefs"[1] about the world, or whether the LLM even has a coherent internal belief system. This general opacity is a growing impediment to widespread use of LLMs, e.g., in critical applications such as medicine, law, and hiring decisions, where properties of explainability, interpretability, and trust are paramount. Our goal is to help alleviate such opacity by constructing an explicit representation of system beliefs and their inferential relationships (including to answer candidates), so that answers are supported by interpretable chains of reasoning. These constructed **belief graphs**, e.g., Figures 1 and 2, form a **rational layer** above the LLM explaining how answers follow from beliefs, and provide a window into some of the latent contents of the model, potentially helping users

[1] We adopt a simple definition of belief, namely that a model believes X if it answers "yes" to the question "Is X true?". Other definitions could also be used; see Section 2.

understand and trust model answers.

In addition, when we do this, we find such graphs expose latent inconsistencies in the model's beliefs. We show how such inconsistencies can be resolved using constraint satisfaction techniques. When we do this, the rational layer becomes not just a window onto the model, but an active reasoning component in its own right in a *larger, overall system*, comprising the (frozen) LLM plus rational layer (blue box, Figure 1). We show this results in a more consistent set of beliefs in the overall system, without harming overall answer accuracy (although some individual answers may change). The result is answers supported by faithful, system-believed chains of reasoning drawn from a consistent belief system.

Our approach, called REFLEX, introduces a **rational layer** consisting of two parts. First, to produce a belief graph, we recursively ask the LLM to explain why each candidate answer might be true, expressed as a set of sentences that entail the answer. This builds on earlier work on generating entailment-based and chain-of-thought explanations (Tafjord et al., 2022; Weir and Durme, 2022; Wei et al., 2022). We then add a self-verification step to check that the model itself believes those generations (i.e., that the model believes what it says), allowing us to identify sentences reflecting the model's own internal knowledge. For example, in Figure 1, when asked to explain S1 ("giraffes give live birth"), the model generates S7 ([because] "mammals give live birth") and S4 ([and] "a giraffe is a mammal"). Self-querying then checks if the model actually believes its generations ("Do mammals give live birth?"). The answer ("yes"/"no") assigns a true/false (T/F) value to each generation, indicated in Figure 1 by white/grey nodes. This procedure is then applied recursively to the generated, supporting sentences. The resulting network of model beliefs and their dependencies provides a a window into the model.

Second, we apply a formal constraint reasoner to this graph to resolve inconsistencies, by finding the optimal (minimal cost, Section 3.3) way of flipping T/F values. For example, on the left in Figure 1, S2 and S3 ("spiders do/don't give live birth") are in an XOR relationship (i.e., exactly one must be false), but both are believed as true (white) by the LLM - a latent contradiction within the LLM. Constraint reasoning then seeks to remove such inconsistencies, here flipping the belief value on S2 from T to F (Figure 1, right), repairing the contradiction. This builds on earlier techniques (Kassner et al., 2021; Mitchell et al., 2022; Jung et al., 2022), though in a notably richer setting with over 350 nodes and 80 constraints per question, joint inference across answer candidates, and a variety of constraint types. The overall result is a fully autonomous, self-reflective system that is able to deliberate (and if necessary change) its answers, thereby resolving latent inconsistencies that would otherwise go unnoticed, and provide faithful explanations drawn from a consistent belief system.

We evaluate our implementation of REFLEX on three datasets: EntailmentBank (Dalvi et al., 2021), OBQA (Mihaylov et al., 2018), and QuaRTz (Tafjord et al., 2019). We find that REFLEX is able to construct belief graphs with significantly improved consistency (by 8%-11% absolute) without harming overall answer accuracy. In addition, answers are now supported by a more consistent, system-believed chain of reasoning, providing a window into the previously latent beliefs of the model. Our contributions are thus:

1. A **new style of system architecture** in which an LLM is extended with a **rational layer** in which an explicit representation of system beliefs and relationships is constructed and which can be reasoned over. This layer provides an **interpretable window** into system beliefs, adds a systematic reasoning capablity, and allows latent inconsistencies present in the LLM to be repaired.
2. An implementation of this architecture demonstrating that the **consistency of the overall system's network of beliefs can be significantly improved** without harming answer accuracy. Answers are now supported by explicit, interpretable chains of reasoning drawn from a more consistent network of beliefs.

2 Related Work

Materializing a Model's Internal Knowledge: It is now well recognized that LLMs contain extensive world knowledge (Petroni et al., 2019, 2020; Davison et al., 2019; Peters et al., 2019; Jiang et al., 2020; Roberts et al., 2020) that somehow enables them to perform well. Recent work has attempted to expose that knowledge in various ways, both to justify answers and improve performance, and our work falls into this genre. Standard explanation generation methods (Wiegreffe and Marasović, 2021) can produce compelling explanations, but

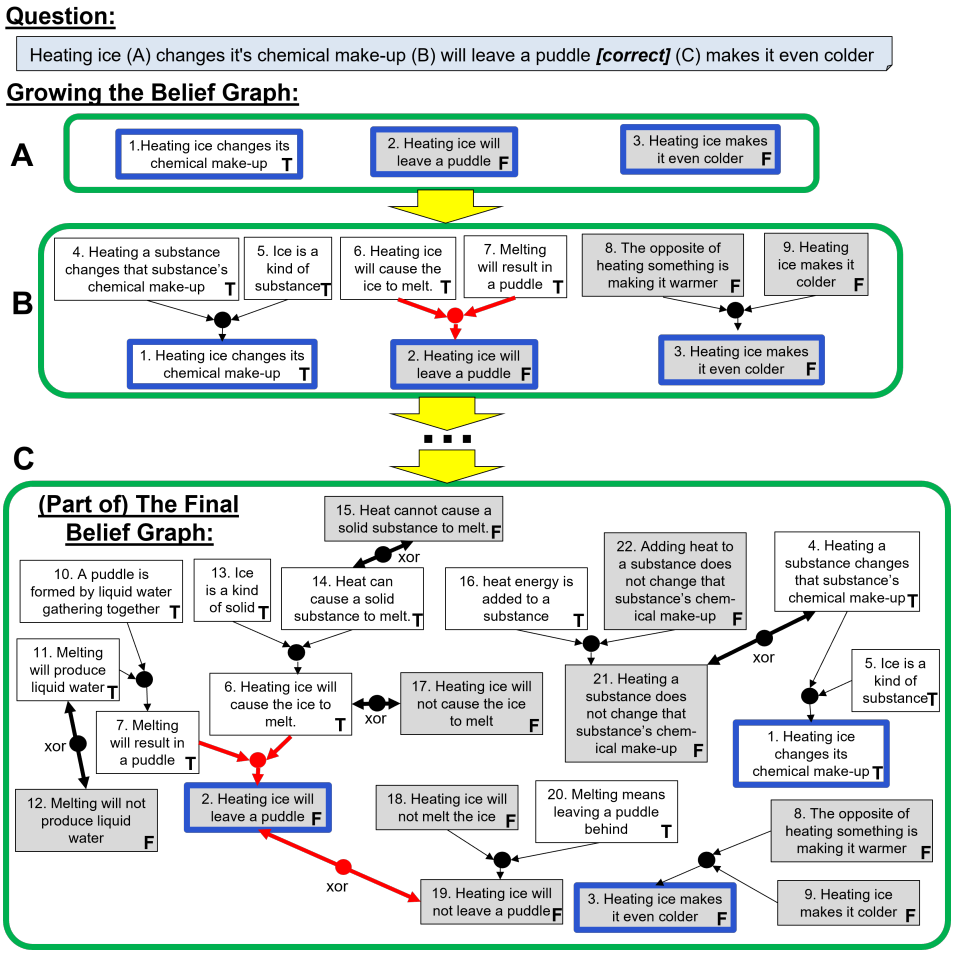


Figure 2: Given a question, each answer choice is first converted to a hypothesis statement (**A**). The belief graph is then constructed in stages, first generating rules that conclude the hypotheses (**B**), then backward-chaining to generate rules concluding the premises of those first rules, etc., and adding in negated versions of graph statements connected with the originals via XOR links (e.g., nodes 11 and 12), until the stopping criterion is met (**C**). Statements are then labeled with the model's belief in them (true/false), found via self-querying (white = believed true, grey = believed false). Finally, logical conflicts are identified (colored red), and constraint satisfaction techniques are used to resolve them. In this case, as there is strong evidence that node 2 is actually true (7 & 6 $\rightarrow$ 2, not(19) $\rightarrow$ 2), the solver finds that the minimum cost repair is to flip node 2's label from FALSE to TRUE. Here, node 2 ends up being selected as the final answer, thus correctly answering the original question.

with no guarantee that the generated sequence of tokens expresses the model's internal knowledge, nor entails the actual answer. Similarly, chain-of-thought (CoT) (Wei et al., 2022) and Least-to-Most (Zhou et al., 2023) prompting generate (in different ways) a step-by-step reasoning chain along with an answer, but again with no claim that the chain reflects the model's internal knowledge nor is valid reasoning (Subramanian et al., 2020).

To add semantics to generations, several systems have used self-querying to verify that generations reflect model-believed facts (by self-querying "Is p true?") (e.g., Kassner et al., 2021; Jung et al., 2022), or model-believed rules (by self-querying "Does p imply q?") (e.g., Tafjord et al., 2022). We build on these to construct a **belief graph**, namely a network of model-believed facts and their inferential relationships, which can then be reflected on.

Beliefs: We refer to the model's factual opinions as "beliefs" rather than "knowledge" because those opinions may be wrong. In general, an agent

can be said to believe p if it acts as if p was true (Schwitzgebel, 2019). Following Kassner et al. (2021) and Richardson et al. (2022), we take a simple, syntactic operationalization of this, namely the agent answers "yes" to the question "p?", but also note that more semantic versions could be used, e.g., the agent also answers "yes" to paraphrases and implications of p.

Reducing Inconsistency: LLMs are known to be inconsistent in their answers (Ettinger, 2020; Kassner and Schütze, 2020; Davison et al., 2019; Ravichander et al., 2020; Elazar et al., 2021; Subramanian et al., 2020; Gu et al., 2023), and several recent works have used constraint reasoners to identify and reduce inconsistency. BeliefBank used a MaxSAT solver to resolve inconsistencies between model beliefs, but required a hand-provided set of constraint rules (Kassner et al., 2021). ConCoRD (Mitchell et al., 2022) similarly used MaxSAT to ensure model answers were consistent with NLI-derived entailment constraints between them, but did not introduce additional model-believed facts and rules. Maieutic Prompting (Jung et al., 2022) also used MaxSAT to resolve inconsistencies between facts in prompt-induced explanation chains. However, those chains were not validated as reflecting model-believed constraint rules[2], and did not support conjunction. REFLEX extends these reasoning chains to provide a full semantic account of how answers are supported by the model's internal knowledge. Additionally, it performs joint reasoning across answer candidates and operates at a much larger scale (e.g., over 350 nodes on average for each question) and with a variety of constraint types.

3 REFLEX: Our Approach

3.1 Belief Graphs

Our belief graphs are defined over a set of natural language true/false *statements* and represent a set of *rules* that constrain the truth values of these statements. We refer to statements that are factually true in the world as *facts*. The truth value assigned by a model M to a statement is referred to as M's *belief* in that statement (cf. Footnote 1). A model's internal beliefs may not always align with facts. Our goal is to extract a model's initial beliefs about statements inferentially related to all top-level hypotheses of interest, and perform reasoning to update these beliefs so as to make them more consistent with respect to the rules, and ideally also factually more accurate.

A belief graph is a type of *factor graph* commonly used in the probabilistic inference literature (Loeliger, 2004). Formally, it is defined as an undirected graph $G = (N, E)$ with nodes N and edges E. Nodes are of two types: A *statement node* (referred to as a "variable node" in a factor graph) is a triple (s, l, c_s) containing a natural language statement s, an associated value $l \in \{T, F\}$ initially denoting M's belief that s is true or false, and a confidence $c_s \in [0, 1]$ denoting a confidence in that label. A *rule node* (referred to as a "factor node" in a factor graph) is a pair (r, c_r) denoting a disjunctive rule or constraint over statements, with confidence c_r. It takes the form $r = (-s_1 \vee \ldots \vee -s_\ell \vee s_{\ell+1} \vee \ldots \vee s_k)$. For ease of interpretation, we view this constraint as $r = p \rightarrow h$ where $p = s_1 \wedge \ldots \wedge s_\ell$ is a conjunctive premise and $h = s_{\ell+1} \vee \ldots \vee s_k$ is a disjunctive hypothesis. The rule says that if p is true, so must be h; and the contrapositive of this.

Edges E connect rule nodes to the statements they constrain, denoting their dependence. For legibility, we draw edges directionally to depict the way the rule reads: the statements in p point to r, which in turn points to h. Mathematically, the influence is bidirectional and the depicted directionality is irrelevant during reasoning (Section 3.3), just as in a standard factor graph.

We adopt the standard probabilistic semantics of factor graphs, thereby associating a belief graph with a well-defined probability distribution over any set of statement beliefs. For a **statement node** (s, l, c_s), the cost $cost_s$ for setting it to l is 0, and that for setting it against l is c_s; the corresponding *weight* of this node is $w_s = \exp(-cost_s)$. Costs and weights for a **rule node** (r, c_r) are defined similarly, based on whether the beliefs satisfy r or not. Finally, the overall weight of a T/F assignment to all statements is $\prod_s w_s \cdot \prod_r w_r$, which, when normalized by the total weight across all possible assignments, yields a probability distribution over such assignments. We will be interested in finding the *most consistent set of beliefs*, i.e., a T/F assignment to statements with the minimum overall weight, which is equivalent to minimizing

[2] REFLEX checks whether both the statements s_i, and the rules $(s_i \rightarrow h)$, are believed by the model via self-querying, e.g., by asking "Does $s_i \rightarrow h$?", and also scores the strength of those beliefs. In maieutic prompting, the generated rules are not checked against the model, resulting in rules that the model itself may not believe, if queried about them.

$\sum_s cost_s + \sum_r cost_r$. This is referred to as the MPE (most probable explanation) problem in the graphical models literature, which we later solve exactly using a MaxSAT constraint solver based on a standard translation of MPE into weighted MaxSAT (Park, 2002; Sang et al., 2007).

3.2 Constructing Belief Graphs

Given an initial node (statement) s, a belief graph G is produced by a backward-chaining process described below, in which G is recursively expanded to add statements that together may entail s.

3.2.1 Basic Operations

Let h denote a hypothesis (language statement s) of interest and p a premise—a set of statements $\{s_1,\ldots,s_n\}$ that together *may* entail h. Given these, there are **three basic operations** required to generate belief graphs:

1. $h \Rightarrow p$: Given h, *generate* a p that may entail h.

2. $s \Rightarrow (l, c_s)$: Given a statement s, output a true/false value l and a confidence in the belief that s has truth value l (as assessed via yes/no question-answering).

3. $(p, h) \Rightarrow c_r$: Given p and h, output a confidence that the candidate rule $r = p \rightarrow h$ holds.

The most important of these is the first operation, in which the model self-generates conjunctive rules concluding h (i.e., reason p for believing h), thus adding new nodes to the graph.

There are several ways of implementing these basic functions, and our algorithm is agnostic to the method used. In our work here, we use Entailer, an off-the-shelf T5-11B trained model with these functionalities (Tafjord et al., 2022). Further, since the raw score produced by the model tends to be skewed towards 0 or 1, when computing c_s and c_r in practice, we re-scale the raw model score using a set of hyperparameters (cf. Appendix B).

One may use alternative ways to implement these operators, such as chain-of-thought prompting a model like GPT3 (Wei et al., 2022) or ChatGPT (OpenAI, 2022). For example, to generate a rule concluding a hypothesis h such as "Plants require CO2 to make their food.", the model could be prompted with h followed by "Explain the last statement with a 2-step reasoning chain.", the numbered generations forming the premise p. Similarly, generated statements and rules can be validated as reflecting the model's beliefs by self-querying ("Is s true?", "Does p imply h?"), and then using the generated yes/no answer token probabilities as the model's confidence (Kadavath et al., 2022).

Algorithm 1 The recursive algorithm for constructing a belief graph of max depth $d_{\max}$ for a hypothesis set $\mathcal{H}$. The subroutine EXTEND-GRAPH takes a partial graph G as an input and extends it in place with one statement and its subgraph.

procedure GENERATE-GRAPH(hypotheses $\mathcal{H}$, max depth $d_{\max}$):
  let G = empty graph
  foreach $h \in \mathcal{H}$
    call EXTEND-GRAPH($h, 0, d_{\max}, G$)
  add MC rule node $\left(\bigvee_{h\in\mathcal{H}} h, \infty\right)$ to G
  foreach pair (h_i, h_j) of hypotheses in $\mathcal{H}$
    add MC rule node $(\neg h_i \vee \neg h_j, c_{\text{mc}})$ to G
  return G

procedure EXTEND-GRAPH(statement s, current depth d, max depth $d_{\max}$, partial graph G):
  call operator $s \Rightarrow (l, c_s)$ to score statement s
  add statement node (s, l, c_s) to G
  gen. the negation sentence $negs$ = neg(s)
  add rule node $(\text{XOR}(s, negs), c_{\text{xor}})$ to G
  call EXTEND-GRAPH($negs, d+1, d_{\max}, G$)
  if $d < d_{\max}$ **do**:
    let $h = s$
    call operator $h \Rightarrow p$ to generate p
    call operator $(p, h) \Rightarrow c_r$ to score rule $p \rightarrow h$
    add rule node $(p \rightarrow h, c_r)$ to G
    foreach $s_i \in p$
      call EXTEND-GRAPH($s_i, d+1, d_{\max}, G$)

3.2.2 Initial Hypothesis Generation

Given a question, we first generate a set $\mathcal{H}$ of hypothesis sentences (e.g., "Is the sky (A) blue (B) yellow" $\rightarrow$ { h_1 = "The sky is blue.", h_2 = "The sky is yellow."}).[3] An N-way multiple choice question yields N hypotheses in $\mathcal{H}$. A true/false question yields 2 hypotheses. To handle open-ended questions, candidate answers can be generated, e.g., using nucleus sampling (Holtzman et al., 2019).

3.2.3 Belief Graph Generation

The belief graph generation process is shown in Algorithm 1. An example of (part of) a generated belief graph is shown in Figure 2.

Given a set $\mathcal{H}$ of hypotheses, we generate a single belief graph G by using our basic operations (Section 3.2.1) to recursively generate rules that conclude each $h_i \in \mathcal{H}$ up to a fixed maximum depth $d_{\max}$. (Each original h_i is at depth $d = 0$.)

For each statement s, we also generate nodes $negs$ (and their recursive subgraphs) expressing its negation, e.g., "The sky is not blue." from "The

[3] Conversion of a QA pair to a declarative hypothesis D uses a custom T5-11B model trained on the QA2D dataset (Demszky et al., 2018).

sky is blue.".[4] Each pair s and $negs$ is connected with an XOR rule, indicating a (soft) preference for setting exactly one of them to true; this is represented as two disjunctive constraints $(s \vee negs)$ and $(-s \vee -negs)$ whose weight c_{xor} is a fixed hyperparameter. Lastly, we add a multiple-choice (MC) constraint which has two parts: a hard constraint (with infinite cost) that at least one hypothesis must be chosen, and a soft constraint[5] that no more than one should be chosen. The soft constraint is associated with a fixed hyperparameter weight c_{mc}.

3.3 Reasoning Over Belief Graphs

Belief graphs provide a window into the model's beliefs about some of the relevant statements and their (believed) inferential relationships to candidate answers to a question. As others have shown (Kassner et al., 2021; Mitchell et al., 2022), such beliefs can be inconsistent, and materializing those inconsistencies provides one the opportunity to remove or reduce them.

In a similar vein, and as discussed in Section 3.1, REFLEX performs inference over belief graphs in order to compute an updated set of beliefs that is as consistent as possible with the rules. To this end, it converts belief graphs into an equivalent weighted MaxSAT problem and uses an off-the-shelf MaxSAT solver (RC2, (Ignatiev, 2019)) to compute the optimal flips of initial true/false beliefs that minimize global inconsistency. It then discards all rules that are in conflict with the updated statement beliefs, obtaining a smaller, updated belief graph. This **smaller belief graph produced by REFLEX is self-consistent** and provides inferential support for the top-level hypotheses.

3.4 Generating Faithful Explanations

Notably, the smaller updated belief graph produced by REFLEX provides a **faithful** explanation of the answer it predicts, in the sense that it accurately represents the reasoning process behind *the overall system's* prediction (Lyu et al., 2022). This is true as the MaxSAT reasoning process results precisely in a self-consistent set of beliefs from which REFLEX determines whether to believe a candidate answer or not, and produces its final prediction based on this (rather than on the raw LLM output alone; note that we do not make any claims about how the internal reasoning of the LLM component operates.) Thus, REFLEX provides the user with an interpretable reasoning trace, allowing the user to understand how it derived the answer from more rudimentary facts (Subramanian et al., 2020).

We note that the original belief graph (before reasoning) may reveal that the model's original explanation is, in fact, *not* faithful to its own beliefs. For example, in Figure 2, the model believes statements 6, 7, and that 6 & 7 entail 2, but does not believe 2 (colored grey). Thus, the global reasoning layer of REFLEX plays a critical role in arriving at faithful explanations.

4 Experiments and Results

The goal of our experiments is to evaluate the extent to which our overall system, namely an LLM plus a self-reflecting, rational layer, helps expose and resolve inconsistencies in the LLM's beliefs without harming accuracy. Importantly, REFLEX is evaluated in a *zero-shot* setting, without relying on training instances of the target datasets.

Datasets. We use the test partitions of three existing multiple-choice datasets: EntailmentBank (Dalvi et al., 2021), OBQA (Mihaylov et al., 2018), and QuaRTz (Tafjord et al., 2019). We chose our datasets as they contain inferentially rich questions (typically) requiring reasoning. The partitions contain 339, 500, and 784 examples, respectively.

Models. The **baseline LLM** we use is an LLM that has been trained to perform QA and also supports the basic operations discussed in Sec. 3.2.1, enabling us to assess how much it can be improved by adding a REFLEX layer. To this end, we use a publicly available, frozen, off-the-shelf T5-11B LLM called Entailer (Tafjord et al., 2022). To answer an MC question with this LLM, we score each answer hypothesis (c_s, Section 3.2.1) and select the one with the highest truth confidence. If Entailer assigns false values to all answer choices, we select the hypothesis with the lowest false confidence.

REFLEX then adds a rational layer to this LLM, creating a new system that is also able to self-reflect and modify its beliefs. To ensure the different belief graph scores in REFLEX are appropriately calibrated, we use nine hyperparameters, tuned once on the dev partition of EntailmentBank (Dalvi et al., 2021) and then kept fixed for all experiments. Details are in Appendix B. Note the LLM itself remains frozen, with belief revision occurring in the

[4] We use a simple, custom-built utility for this, namely a T5-base model trained on 9k Turk-generated examples.

[5] soft, to allow for cases with multiple valid answers, e.g., open-ended questions or those asking for the best answer.

rational (belief graph) layer above it.

Metrics. For measuring **self-consistency**, we follow Li et al. (2019) and report the *conditional constraint violation* (τ) metric, defined as follows: the fraction of rules whose *premises* p are believed true, but whose *hypothesis* h is not. In other words, over all rules of the form $p \rightarrow h$, τ is:

$$\tau = \frac{|\{p \rightarrow h \mid p = \mathrm{T}, h = \mathrm{F}\}|}{|\{p \rightarrow h \mid p = \mathrm{T}\}|}$$

where $s = T$ denotes the system believes statement s to be true (similarly for $s = F$). The numerator of τ thus captures the number of constraints the system *violates*. The denominator captures the number of *applicable* constraints. We then report the following metric: **consistency** = 1 - τ.

For **QA performance**, we report standard **multiple-choice accuracy**: 1 point for predicting the correct answer, $1/N$ points for predicting N answers including the correct one, $1/k$ points for no prediction (k = # answer options), 0 otherwise.

4.1 Results

Consistency. Table 1 shows consistency results on the test partitions of our datasets. We observe **significant consistency gains** (by 8%-11% absolute), showing REFLEX's effectiveness at creating a consistent belief network within the overall system.

System	EntailmentBank	OBQA	Quartz
LLM	87.0	88.2	85.7
LLM + rational layer (REFLEX)	**96.1**	**95.9**	**96.6**

Table 1: **Consistency:** By adding a rational layer to the baseline LLM, REFLEX significantly improves consistency among beliefs by resolving uncovered conflicts.

Accuracy. Table 2 shows overall performance on our three datasets (test partitions). As can be seen, we observe stable accuracy, as well as the answers now being faithful to the reasoning chains in the belief graph. This is significant, as it allows users to understand how answers follow from system beliefs (and in cases where an LLM belief was flipped, why that belief is untenable in the broader system).

Ablations. To study the impact of the three different types of rules on consistency improvement, we using the EntilmentBank dataset (dev partition).

System	EntailmentBank	OBQA	Quartz
LLM	79.4	74.0	80.2
LLM + rational layer (REFLEX)	79.9	75.0	80.0

Table 2: **QA accuracy:** REFLEX's belief revision in the rational layer preserves overall QA accuracy.

To do this, given the belief graph for a question, we mask out (separately, rather than cumulatively) each type of rule in turn when providing the graph to the MaxSAT solver. We then run the constraint solver and measure the resulting self-consistency of beliefs on the original graph.

System	EntailmentBank
REFLEX (our system):	96.1
- without $p \rightarrow h$ rules	93.8
- without XOR rules	90.4
- without MC rule	95.8

Table 3: **Consistency:** Ablations on EntailmentBank (Dev) suggest that all three types of rules contribute to improving self-consistency.

The results are shown in Table 3 (the MC rule is the constraint that exactly one multiple-choice option should be chosen, Section 3.2.3). The results indicate that all three types of rules contribute to the system's consistency improvements.

4.2 Success Analysis

We identify three classes of successful reasoning by the constraint reasoner: (a) latent model beliefs correct an initially wrong answer (Figure 3); (b) the system corrects an initially erroneous, latent model belief (Figure 4); and (c) strong model beliefs identify and reject a bad rule (Figure 5). These types of system corrections help to improve accuracy and produce answers supported by valid chains of reasoning, allowing users insight into why an answer follows from the model's knowledge.

4.3 Failure Analysis

Reasoning can also make mistakes. From a manual analysis of 50 random questions from EntailmentBank that REFLEX answered incorrectly, we identified five main causes of failure and their approximate frequency (**Note** that multiple categories can apply, hence total is $> 100\%$):

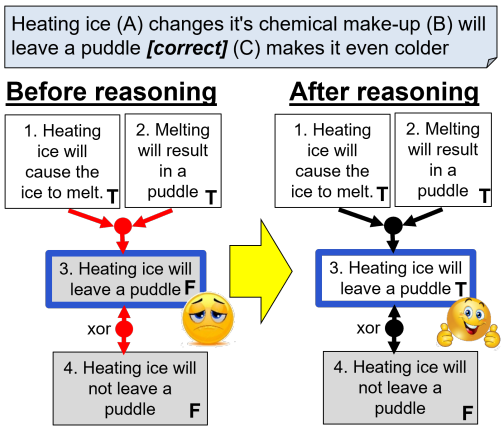


Figure 3: **Example of good reasoning:** The model's beliefs in 1 and 2, and the rule 1 & 2 → 3, as well as the xor constraint, causes it to (desirably) flip its belief in 3 from false (grey, before) to true (white, after).

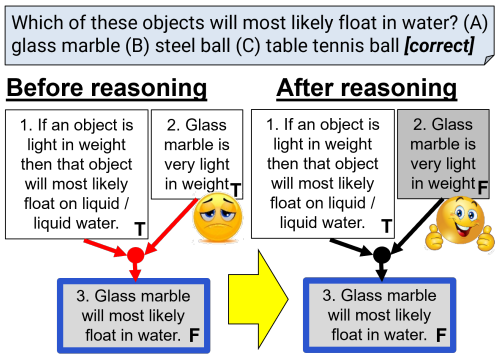


Figure 4: **Example of good reasoning:** Although the model correctly believes option (A) is false (grey, node 3), this answer conflicts with other beliefs (red). Reasoning leads the system to realize that its weakest belief (2) is actually false, correctly flipping its label from true (white) to false (grey, right side) restoring consistency.

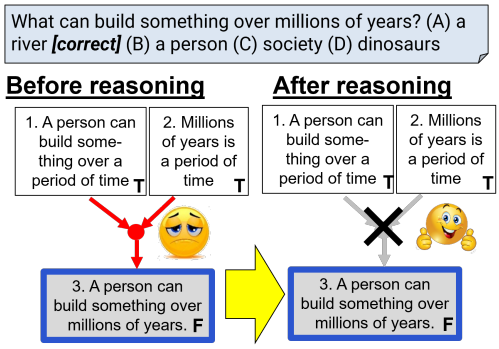


Figure 5: **Example of good reasoning:** Here the reasoner (desirably) chooses to reject the violated (bad) rule rather than flip a belief, as the minimum cost way to restore consistency.

1. Missing Rules (≈30%): In some cases, the system generates irrelevant rules but misses an important one needed to support the correct answer, resulting in incorrect conclusions. While somewhat subjective, this is a notable error category that we observe. For example for the question:

A human cannot survive the loss of (A) The liver ***[correct]*** *(B) A lung (C) A kidney*

the system incorrectly concludes (B) is true, ignoring the commonsense rule that with two lungs, a person can survive without one of them.

2. Incorrect Beliefs (≈30%): Sometimes the reasoner fails to correct incorrect model beliefs, either because the model's confidence is high or evidence against them is weak or missing. In the example shown in Figure 7, the model's strong, incorrect beliefs that "river deltas are reservoirs" and "reservoirs always provide freshwater" (untrue of oceans, say) causes it to incorrectly conclude that "deltas are freshwater reservoirs".

3. Incorrect Rules (≈10%): Rule generation can produce bad rules, e.g., in Figure 5), and in some cases the constraint reasoner fails to reject them if they are strongly believed. In particular, confusion or ambiguity over quantifiers can result in bad rules, e.g., (emphasis added) "*Some* animals catch their prey with trickery." & "A spider is a kind of animal." → "Spiders catch their prey with trickery.". Similarly the model generates the fallacy: "Some people don't mind not moving for an hour" & "breathing is a kind of movement" → "Some people don't mind not breathing for an hour."

4. Ambiguous Statements, Unexpected Reasoning (≈10%): A common cause of error is the surprising ambiguity of belief statements, which can often be read in multiple ways. In several cases, the model adopts a valid but unexpected interpretation, resulting in "errors" compared to the gold answer label. For example, in Figure 6, the model takes the word "always" in a literal sense ("glaciers will not *always* be there"), resulting in an answer that differs from the gold label. Developing ways to attach context to these statements to help disambiguate them would help alleviate such errors.

5. Multiple Valid Answers (≈10%): A final cause of "error" - at least with respect to the gold label - is that multiple answers may be valid, and

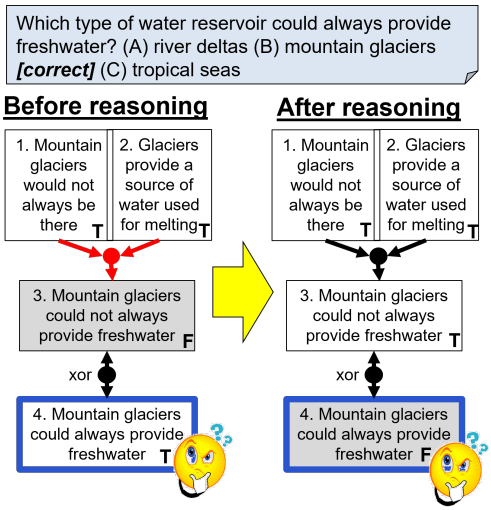


Figure 6: **Unexpected reasoning:** Here the model unexpectedly pays particular attention to the world "always". Because it strongly believes that glaciers will not *always* be there (1, white), the system prefers to flip its beliefs in 3 and 4, rather than flipping 1, thus rejecting answer option B (arguably correctly).

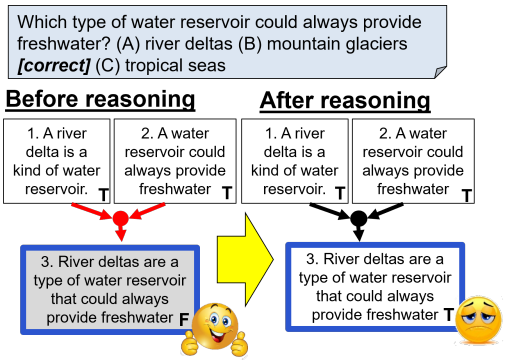


Figure 7: **Failure due to bad beliefs:** The model strongly believes both 1 and 2 (although both are factually incorrect), here causing 3's label to undesirably flip from false (correct) to true (incorrect).

the question is asking for the **best** answer; eg. for "What could fill a beach ball? (A) Oxygen (B) Water ...", A is labeled correct, while B is also a valid answer. REFLEX (desirably) finds valid reasoning chains for both, but the notion of highest-scoring proof does not fully correlate with the notion of "best answer" intended by the question author.

5 Future Work

There are several impactful ways this work could be further extended. First, incorporating the question's *context* in the belief statements in our rational layer could make the semantics of the beliefs more precise, thus avoiding potential ambiguity in their truth value. Second, one could use the belief graph itself to identify the key reasoning pieces that the LLM is most uncertain about. This could then guide a *human-in-the-loop* mechanism to correct or validate uncertain pieces via user interaction. Third, maintaining a *persistent belief graph* over multiple questions could help make the system more consistent across questions. This, in turn, would make a user's conversational experience with the system more coherent in a longer dialog setting. Lastly, after resolving inconsistencies in the rational layer, we could consider *propagating information back to the LLM layer* in order to update it (via fine-tuning, model editing, memory-based architectures, etc.), helping avoid similar inconsistencies in the future.

6 Conclusion

While LLMs perform well, the interdependencies between their answers and their other beliefs is opaque, and may even be in conflict. This lack of interpretability is a significant impediment to widespread use of LLMs. To reduce this opacity, and reduce these conflicts, we have proposed REFLEX, a new system architecture in which an explicit, interpretable representation of beliefs - the **belief graph** - is added as a **rational layer** above the LLM. This layer providing a window into system beliefs, and allows latent inconsistencies in the LLM alone to reasoned about and repaired. Our implementation shows that belief consistency of the overall system is significantly improved, without harming answer accuracy, resulting in answers supported by interpretable chains of reasoning drawn from a more consistent belief system. This new architecture is an important step towards improving confidence in system behavior, and towards trustable deployment of LLMs in practical applications.

Limitations

We have shown how an LLM can be extended with a self-reflective component, allowing latent model knowledge to be made explicit in the form of a **belief graph**, providing a window into the model's system of beliefs. While exciting, there are several limitations with the current work and opportunities for the future.

First, the reasoning component in the rational

layer can make mistakes, resulting in the overall system rejecting true statements or accepting false ones. A detailed analysis and classification of these failure modes was presented in Section 4.3.

Second, for our experiments, we used the T5-11B based Entailer system as the baseline LLM. While there is every reason to expect our proposed architecture to be effective in reducing inconsistency with newer and larger LLMs such as ChatGPT and LLaMA, this is still to be evaluated. Doing so would require implementing the basic operations needed to construct belief graphs (Section 3.2.1) using instruction prompting and in-context learning. Other work has demonstrated such implementations (e.g., Wei et al., 2022; Jiang et al., 2020), making the outlook promising, but indeed their combination still needs to be demonstrated at scale in an architecture like REFLEX.

Lastly, we found consistency-minimized belief graphs to be highly valuable in understanding the system's successes and failures. We expect these graphs to be a valuable starting point for providing explanations and gaining a user's trust in the system. However, we have not conducted a formal user study to measure this.

Ethics Statement

Like any other project using LLMs, despite the best intentions there is a risk of the model producing biased or offensive statements as part of its explanations, and thus must be used with care and appropriate guards and warnings.

Acknowledgements

This research was made possible, in part, by funding from Open Philanthropy, the European Research Council (#740516) and by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. We also thank Google for providing the TPUs for conducting experiments. Finally, we are grateful for the valuable feedback from the anonymous reviewers.

References

Bhavana Dalvi, Peter Alexander Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. 2021. Explaining answers with entailment trees. In *EMNLP*.

Joe Davison, Joshua Feldman, and Alexander Rush. 2019. Commonsense knowledge mining from pretrained models. In *EMNLP*.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets. *ArXiv*, abs/1809.02922.

Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, E. Hovy, Hinrich Schütze, and Yoav Goldberg. 2021. Measuring and improving consistency in pretrained language models. *TACL*, 9.

Allyson Ettinger. 2020. What BERT is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *TACL*, 8:34–48.

Yuling Gu, Bhavana Dalvi, and Peter Clark. 2023. Do language models have coherent mental models of everyday things? In *ACL*.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. In *ICLR*.

Alexey Ignatiev. 2019. RC2: an efficient MaxSAT solver. *J. Satisf. Boolean Model. Comput.*, 11.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *TACL*, 8.

Jaehun Jung, Lianhui Qin, Sean Welleck, Faeze Brahman, Chandra Bhagavatula, Ronan Le Bras, and Yejin Choi. 2022. Maieutic prompting: Logically consistent reasoning with recursive explanations. In *EMNLP*.

Saurav Kadavath, Tom Conerly, Amanda Askell, T. J. Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zachary Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yushi Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, John Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom B. Brown, Jack Clark, Nicholas Joseph, Benjamin Mann, Sam McCandlish, Christopher Olah, and Jared Kaplan. 2022. Language models (mostly) know what they know. *ArXiv*, abs/2207.05221.

Nora Kassner and H. Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. In *ACL*.

Nora Kassner, Oyvind Tafjord, Hinrich Schutze, and Peter Clark. 2021. BeliefBank: Adding memory to a pre-trained language model for a systematic notion of belief. In *EMNLP*.

Tao Li, Vivek Gupta, Maitrey Mehta, and Vivek Srikumar. 2019. A logic-driven framework for consistency of neural models. In *EMNLP*.

Hans-Andrea Loeliger. 2004. An introduction to factor graphs. *IEEE Signal Processing Magazine*, 21:28–41. https://people.binf.ku.dk/∼thamelry/MLSB08/hal.pdf.

Qing Lyu, Marianna Apidianaki, and Chris Callison-Burch. 2022. Towards faithful model explanation in NLP: A survey. *ArXiv*, abs/2209.11326.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *EMNLP*.

Eric Mitchell, Joseph J. Noh, Siyan Li, William S. Armstrong, Ananth Agarwal, Patrick Liu, Chelsea Finn, and Christopher D. Manning. 2022. Enhancing self-consistency and performance of pre-trained language models through natural language inference. In *EMNLP*.

OpenAI. 2022. ChatGPT: Optimizing language models for dialog. Technical report, openai.com. https://openai.com/blog/chatgpt/.

James D Park. 2002. Using weighted MAX-SAT engines to solve MPE. In *AAAI/IAAI*.

Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *EMNLP*.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. In *Automated Knowledge Base Construction*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *EMNLP*.

Abhilasha Ravichander, Eduard Hovy, Kaheer Suleman, Adam Trischler, and Jackie Chi Kit Cheung. 2020. On the systematicity of probing contextualized word representations: The case of hypernymy in BERT. In *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics*.

Kyle Richardson, Ronen Tamari, Oren Sultan, Reut Tsarfaty, Dafna Shahaf, and Ashish Sabharwal. 2022. Breakpoint Transformers for Modeling and Tracking Intermediate Beliefs. In *EMNLP*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *EMNLP*.

Tian Sang, Paul Beame, and Henry A Kautz. 2007. A dynamic approach for MPE and weighted MAX-SAT. In *IJCAI*.

Eric Schwitzgebel. 2019. Belief. *Stanford Encyclopedia of Philosophy*. https://plato.stanford.edu/entries/belief/.

Sanjay Subramanian, Ben Bogin, Nitish Gupta, Tomer Wolfson, Sameer Singh, Jonathan Berant, and Matt Gardner. 2020. Obtaining faithful interpretations from compositional neural networks. In *ACL*.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2022. Entailer: Answering questions with faithful and truthful chains of reasoning. In *EMNLP*.

Oyvind Tafjord, Matt Gardner, Kevin Lin, and Peter Clark. 2019. QuaRTz: An open-domain dataset of qualitative relationship questions. In *EMNLP*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. In *NeurIPS*.

Nathaniel Weir and Benjamin Van Durme. 2022. Dynamic generation of interpretable inference rules in a neuro-symbolic expert system. *ArXiv*, abs/2209.07662.

Sarah Wiegreffe and Ana Marasović. 2021. Teach me to explain: A review of datasets for explainable natural language processing. In *NeurIPS Datasets and Benchmarks*.

Denny Zhou, Nathanael Scharli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Olivier Bousquet, Quoc Le, and Ed Huai hsin Chi. 2023. Least-to-most prompting enables complex reasoning in large language models. *ICLR*.

A Additional Results

We report results on the dev set of the EntailmentBank dataset in Table A1.

System	EntailmentBank (dev)	
	Consistency	Accuracy
LLM	87.5	78.6
LLM + rational layer (REFLEX)	**96.1**	**81.8**

Table A1: Results on EntailmentBank (dev), used to tune the system's hyperparameters.

B Hyperparameters and Runtime

MaxSAT finds the optimal assignment of true/false labels on statement nodes that minimizes the total penalty of constraint violations. If the true/false label on a statement node is flipped, then the penalty is the model confidence c_{s} in the original label. Similarly if a rule (constraint) is violated by the true/false labels on its associated statements, then the penalty is the model confidence c_{r} in that rule.

We set a number of hyperparameters to ensure that the various sources of confidence are appropriately balanced, and tune these on a development set (EntailmentBank (dev) which is separate from our test sets). We use the same set of hyperparameters for all test sets.

1. As raw model confidences c_{s} are highly skewed towards 0 and 1, we re-calibrate these with $e^{k.(c_{\mathrm{s}}-1)}$, where k is a fixed hyperparameter. Note, that for the MC and XOR rule, the raw input score s is 1.0.
2. We calibrate rule confidences in the same way as we calibrate belief confidences but use separate calibration parameters different types of rules namely:
   - Entailer rules $p \rightarrow h$
   - XOR rules
   - MC rules

   i.e., the raw rule score c is re-calibrated to confidence $e^{k_{type}.(c-1)}$ where k_{type} is the respective hyperparameter per rule type.
3. We set three hyperparameters tuning the respective importance of the three different types of rules. Therefore, the final rule score is computed by $c = t_{type} * e^{k_{type}.(c-1)}$ where t_{type} is the respective hyperparameter constant per rule type.
4. For xor rules between statements s_i and $negs_i$, we remove (ignore) those where there is significant uncertainty, namely where $|score(s_i) - score(negs_i)| \leq m_{xor}$, where m_{xor} is a tuned hyperparmeter.
5. Additionally, we tune a damping parameter that downscales rules on the boundary of the graph. Belief nodes involved in these rules are not supported by any premises and should therefore have less influence than rules with strong support.
6. Finally, we tune the maximum depth $d_{\max}$ of the belief graph.

Hyperparameter	Value
k	9
k_{entailer}	36
k_{xor}	30
k_{mc}	9
t_{entailer}	1.02
t_{xor}	1.1
t_{mc}	0.98
m_{xor}	0.3
$d_{\max}$	5

Table B1: Hyperparameters.

The performance on this dev set partition is shown in Table A1 and the hyperparameter values are shown in Table B1.

The runtime for MaxSAT constraint solving is fast (<1 millisecond per question). However, constructing the belief graph is computationally intensive: Each call to expand or score a node takes ∼2 seconds, and our graphs typically contain ∼600 nodes, so if these calls were maximally parallelized, with each step growing the graph one level deeper, the runtime would be the maximum graph depth (5) x 2 seconds = ∼10 seconds total (or several minutes if a naive sequential implementation were used).

Chapter 7

Are Pretrained Language Models Symbolic Reasoners Over Knowledge?

Are Pretrained Language Models Symbolic Reasoners Over Knowledge?

Nora Kassner*, **Benno Krojer***, **Hinrich Schütze**
Center for Information and Language Processing (CIS)
LMU Munich, Germany
kassner@cis.lmu.de

Abstract

How can pretrained language models (PLMs) learn factual knowledge from the training set? We investigate the two most important mechanisms: reasoning and memorization. Prior work has attempted to quantify the number of facts PLMs learn, but we present, using synthetic data, the first study that investigates the causal relation between facts present in training and facts learned by the PLM. For reasoning, we show that PLMs seem to learn to apply some symbolic reasoning rules correctly but struggle with others, including two-hop reasoning. Further analysis suggests that even the application of learned reasoning rules is flawed. For memorization, we identify schema conformity (facts systematically supported by other facts) and frequency as key factors for its success.

1 Introduction

Pretrained language models (PLMs) like BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019) and RoBERTa (Liu et al., 2019) have emerged as universal tools that capture a diverse range of linguistic and – as more and more evidence suggests – factual knowledge (Petroni et al., 2019; Radford et al., 2019).

Recent work on knowledge captured by PLMs is focused on probing, a methodology that identifies the set of facts a PLM has command of. But little is understood about how this knowledge is acquired during pretraining and why. We analyze the ability of PLMs to acquire factual knowledge focusing on two mechanisms: reasoning and memorization. We pose the following two questions: **a) Symbolic reasoning:** Are PLMs able to infer knowledge not seen explicitly during pretraining? **b) Memorization:** Which factors result in successful memorization of a fact by PLMs?

*equal contribution

We conduct our study by pretraining BERT from scratch on synthetic corpora. The corpora are composed of short knowledge-graph like facts: subject-relation-object triples. To test whether BERT has learned a fact, we mask the object, thereby generating a cloze-style query, and then evaluate predictions.

Symbolic reasoning. We create synthetic corpora to investigate six symbolic rules (equivalence, symmetry, inversion, composition, implication, negation); see Table 1. For each rule, we create a corpus that contains facts from which the rule can be learned. We test BERT's ability to use the rule to infer unseen facts by holding out some facts in a test set. For example, for composition, BERT should infer, after having seen that leopards are faster than sheep and sheep are faster than snails, that leopards are faster than snails.

Our setup is similar to link prediction in the knowledge base domain and therefore can be seen as a natural extension of the question: "Language models as knowledge bases?" (Petroni et al., 2019). In the knowledge base domain, prior work (Sun et al., 2019; Zhang et al., 2020) has shown that models that are able to learn symbolic rules are superior to ones that are not.

Talmor et al. (2019) also investigate symbolic reasoning in BERT using cloze-style queries. However, in their setup, there are two possible reasons for BERT having answered a cloze-style query correctly: (i) the underlying fact was correctly inferred or (ii) it was seen during training. In contrast, since we pretrain BERT from scratch, we have full control over the training setup and can distinguish cases (i) and (ii).

A unique feature of our approach compared to prior work (Sinha et al., 2019; Richardson et al., 2020; Weston et al., 2016; Clark et al., 2020) is that we do not gather all relevant facts and present them to the model at inference time. This is a crucial

Rule		Definition	Example
EQUI	Equivalence	$(e, r, a) \iff (e, s, a)$	(bird, can, fly) $\iff$ (bird, is able to, fly)
SYM	Symmetry	$(e, r, f) \iff (f, r, e)$	(barack, married, michelle) $\iff$ (michelle, married, barack)
INV	Inversion	$(e, r, f) \iff (f, s, e)$	(john, loves, soccer) $\iff$ (soccer, thrills, john)
NEG	Negation	$(e, r, a) \iff (e, \text{not } r, b)$	(jupiter, is, big) $\iff$ (jupiter, is not, small)
IMP	Implication	$(e, r, a) \Rightarrow (e, s, b), (e, s, c), ...$	(dog, is, mammal) $\Rightarrow$ (dog, has, hair), (dog, has, neocortex), ...
COMP	Composition	$(e, r, f) \wedge (f, s, g) \Rightarrow (e, t, g)$	(tiger, faster than, sheep) $\wedge$ (sheep, faster than, snail) $\Rightarrow$ (leopard, faster than, snail) with $r = s = t$

Table 1: The six symbolic rules we investigate (cf. (Nayyeri et al., 2019)) with an example in natural language for entities $e, f, g \in E$, relations $r, s, t \in R$ and attributes $a, b, c \in A$.

difference – note that human inference similarly does not require that all relevant facts are explicitly repeated at inference time.

We find that i) BERT is capable of learning some one-hop rules (equivalence and implication). ii) For others, even though high test precision suggests successful learning, the rules were not in fact learned correctly (symmetry, inversion and negation). iii) BERT struggles with two-hop rules (composition). However, by providing richer semantic context, even two-hop rules can be learned.

Given that BERT can in principle learn some reasoning rules, the question arises whether it does so for standard training corpora. We find that BERT-large has only partially learned the types of rules we investigate here. For example, BERT has some notion of "X shares borders with Y" being symmetric, but it fails to understand rules like symmetry in other cases.

Memorization. During the course of pretraining, BERT sees more data than any human could read in a lifetime, an amount of knowledge that surpasses its storage capacity. We simulate this with a scaled-down version of BERT and a training set that ensures that BERT cannot memorize all facts in training. We identify two important factors that lead to successful memorization. (i) Frequency: Other things being equal, low-frequency facts are not learned whereas frequent facts are. (ii) Schema conformity: Facts that conform with the overall schema of their entities (e.g., "sparrows can fly" in a corpus with many similar facts about birds) are easier to memorize than exceptions (e.g., "penguins can dive").

We publish our code for training and data generation. [1]

[1] https://github.com/BennoKrojer/reasoning-over-facts

2 Data

To test PLMs' reasoning capabilities, natural corpora like Wikipedia are limited since it is difficult to control what the model sees during training. Synthetic corpora provide an effective way of investigating reasoning by giving full control over what knowledge is seen and which rules are employed in generating the data.

In our investigation of PLMs as knowledge bases, it is natural to use (subject, relation, object) triples as basic units of knowledge; we refer to them as *facts*. The underlying vocabulary consists of a set of entities $e, f, g, ... \in E$, relations $r, s, t, ... \in R$ and attributes $a, b, c, ... \in A$, all represented by artificial strings such as e_{14}, r_3 or a_{35}. Two types of facts are generated. (i) **Attribute facts**: relations linking entities to attributes, e.g., (e, r, a) = (leopard, is, fast). (ii) **Entity facts**: relations linking entities, e.g., (e, r, f) = (Paris, is the capital of, France).

In the test set, we mask the objects and generate cloze-style queries of the form "e r [MASK]". The model's task is then to predict the correct object.

2.1 Symbolic Reasoning

Table 1 gives definitions and examples for the six rules (EQUI, SYM, INV, COMP, IMP, NEG) we investigate. The definitions are the basis for our corpus generation algorithms, shown in Figure 1. SYM, INV, COMP generate entity facts and EQUI, IMP, NEG attribute facts. We create a separate corpus for each symbolic rule. Facts are generated by sampling from the underlying vocabulary. For §2.1, this vocabulary consists of 5000 entities, 500 relations and 1000 attributes. Half of the relations follow the rule, the other half is used to generate random facts of entity or attribute type.

We can most easily think of the corpus generation as template filling. For example, looking at SYM in Table 1, the template is $(e, r, f) \iff (f, r, e)$. We first sample a relation r from R and

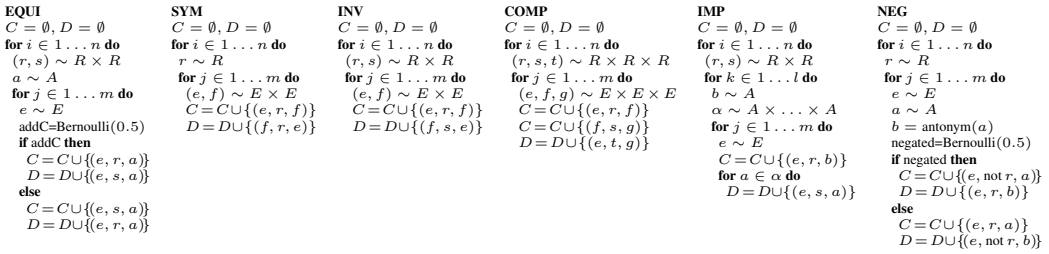


Figure 1: Pseudocode for symbolic reasoning corpus generation. "$a \sim A$" stands for: a is randomly sampled from A. ("$\alpha \sim A \times \ldots \times A$": a tuple of 4 attributes is sampled.) The vocabulary consists of entities $e, f, g \in E$, relations $r, s, t \in R$ and attributes $a, b, c \in A$. Train/test corpora are formed from C and D. $n = 20$, $m = 800$, $l = 2$. See §2.1 for details.

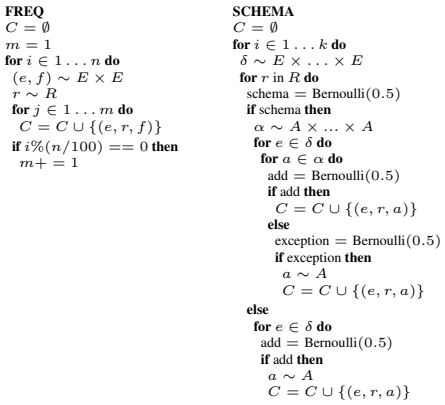


Figure 2: Pseudocode for memorization corpus generation. "$a \sim A$" stands for: a is randomly sampled from A. ("$\delta \sim E \times \ldots \times E$": a tuple of 250 entities is sampled. "$\alpha \sim A \times ... \times A$": a tuple of 10 attributes is sampled.) The vocabulary consists of entities $e \in E$, relations $r \in R$ and attributes $a \in A$. C is both training set and test set. n = 800,000, $k = 250$. See §2.2 for details.

then two entities e and f from E. We then add (e, r, f) and (f, r, e) to the corpus – this is one *instance* of applying the SYM rule from which symmetry can be learned. Similarly, the other rules also generate instances.

For each of the other rules, the template filling is modified to conform with its definition in Table 1. INV corresponds directly to SYM. COMP is a two-hop rule whereas the other five are one-hop rules. EQUI generates instances from which one can learn that the relations r and s are equivalent. IMP generates implication instances, e.g., (e, r, b) (= (dog, is, mammal)) implies (e, s, a_1) (= (dog, has, hair)), (e, s, a_2) (= (dog, has, neocortex)) etc. Per premise we create four implied facts.

For NEG, we generate pairs of facts (e, r, a) (= (jupiter, is, big)) and $(e, \text{not } r, b)$ (= (jupiter, is not, small)). We define the antonym function in Figure 1 (NEG) as returning for each attribute its antonym, i.e., attributes are paired, each pair consisting of a positive and a negative attribute.

Each of the six generation algorithms has the outer loop "for $i \in 1 \ldots n$" (where $n = 20$) that samples one, two or three relations (and potentially attributes) and generates a subcorpus for these relations; and the inner loop "for $j \in 1 \ldots m$" (where $m = 800$) that generates the subcorpus of instances for the sampled relations.

Train/test split. The data generation algorithms generate two subsets of facts C and D, see Figure 1. For each rule, we merge all of C with 90% of D (randomly sampled) to create the training set. The rest of D (i.e., the other 10%) serves as the test set.

For some of the cloze queries "e r [MASK]", there are multiple correct objects that can be substituted for MASK. Thus, we rank predictions and compute precision at m, i.e., precision in the top m where m is the number of correct objects. We average precision at m for all cloze queries.

This experimental setup allows us to test to what extent BERT learns the six rules, i.e., to what extent the facts in the test set are correctly inferred from their premises in the training set.

2.2 Memorization

For memorization, the vocabulary consists of 125,000 entities, 20 relations and 2250 attributes.

Effect of frequency on memorization. Our first experiment tests how the frequency of a fact influences its successful memorization by the model. Figure 2 (left, FREQ) gives the corpus generation algorithm. The outer loop generates 800,000 random facts. These are divided up in groups of 8000. A fact in the first group of 8000 is added once to

the corpus, a fact from the second group is added twice and so on. A fact from the last group is added 100 times to the corpus. The resulting corpus C is both the training set and the test set.

Effect of schema conformity. In this experiment, we investigate the hypothesis that a fact can be memorized more easily if it is schema conformant.

Figure 2 (right, SCHEMA) gives the corpus generation algorithm. We first sample an entity group: $\delta \sim E \times \ldots \times E$. For each group, relations are either related to the schema ("if schema") or are not (else clause). For example, for the schema "primate" the relations "eat" (eats fruit) and "climb" (climbs trees) are related to the schema, the relation "build" is not since some primates build nests and treehouses, but others do not.

For non-schema relations, facts with random attributes are added to the corpus. In Figure 4, we refer to these facts as (facts with) **unique attributes**. For relations related to the schema, we sample the attributes that are part of the schema: $\alpha \sim A \times ... \times A$ (e.g., ("paranut",...,"banana") for "eat"). Facts are then generated involving these attributes and added to the corpus. In Figure 4, we refer to these facts as (facts with) **group attributes**. We also generate exceptions (e.g., "eats tubers") since schemas generally have **exceptions**.

Similarly, the two lines "add $=$ Bernoulli(0.5)" are intended to make the data more realistic: for a group of entities, its relations and its attributes, the complete cross product of all facts is not available to the human learner. For example, a corpus may contain sentences stating that chimpanzees and baboons eat fruit, but none that states that gorillas eat fruit.

For this second memorization experiment, training set and test set are again identical (i.e., $= C$).

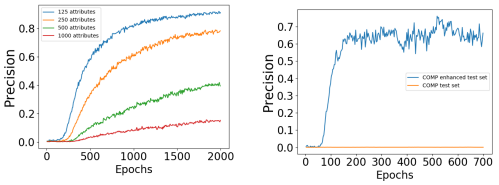
In a final experiment, we modify SCHEMA as follows: exceptions are added 10 times to the corpus (instead of once). This tests the interaction between schema conformity and frequency.

3 BERT Model

BERT uses a deep bidirectional Transformer (Vaswani et al., 2017) encoder to perform masked language modeling. During pretraining, BERT randomly masks positions and learns to predict fillers. We use source code provided by Wolf et al. (2019). Following (Liu et al., 2019), we perform dynamic masking and no next sequence prediction.

rule	train	test
EQUI	99.95	98.28
SYM	99.97	98.40
INV	99.99	87.21
IMP	100.00	80.53
NEG	99.98	20.54
COMP	99.98	0.01
ANTI	100.00	14.85

Table 2: Precision in % of completing facts for symbolic rules. Training corpora generated as specified in Figure 1. See §4.1 for detailed discussion.



(A) NEG (B) COMP

Figure 3: Learning curves for symbolic reasoning. (A) shows precision for NEG with a varying number of attributes. A reduction to 125 attributes enables BERT to successfully apply antonym negation to the test set. (B) shows test set precision for COMP following the standard setup (orange) and an enhanced version (blue). Only in enhanced, i.e., with the introduction of additional facts adding more semantic information, is COMP generalized.

For symbolic rules, we start with BERT-base and tune hyperparameters. We vary the number of layers to avoid that rule learning fails due to over-parametrization, see appendix for details. We report precision based on optimal configuration.

In the memorization experiment, our goal is to investigate the effect of frequency on memorization. Due to a limited compute infrastructure, we scale down BERT to a single hidden layer with 3 attention heads, a hidden size of 192 and an intermediate size of 768.

4 Results, Analysis and Discussion

4.1 Symbolic Reasoning

Table 2 gives results for the symbolic reasoning experiments. BERT has high test set precision for EQUI, SYM, INV and IMP. As we see in Table 1, these rules share that they are "one-hop": The inference can be straightforwardly made from a single premise to a conclusion, e.g., "(barack married michelle)" implies "(michelle married barack)". The crucial difference to prior work is that the premise is not available at inference time. "(michelle married barack)" is correctly inferred by

the model based on its memory of having seen the fact "(barack married michelle)" in the training set and based on the successful acquisition of the symmetry rule. Table 2 seems to suggest that BERT is able to learn one-hop rules and it can successively apply these rules in a natural setting in which the premise is not directly available.

In the rest of this section, we investigate these results further for SYM, INV, NEG and COMP.

4.1.1 Analysis of SYM and INV

Table 2 seems to indicate that BERT can learn that a relation r is symmetric (SYM) and that s and t are inverses (INV) – the evidence is that it generates facts based on the successfully acquired symmetry and inversion properties of the relations r, s and t. We now show that while BERT acquires SYM and INV partially, it also severely overgenerates. Our analysis points to the complexity of evaluating rule learning in PLMs and opens interesting avenues for future work.

Our first observation is that in the SYM experiment, BERT understands *all relations* to be symmetric. Recall that of the total of 500 relations, 250 are symmetric and 250 are used to generate random facts. If we take a fact with a random relation r, say (e, r, f), and prompt BERT with "$(f, r, [MASK])$", then e is predicted in close to 100% of cases. So BERT has simply learned that any relation is symmetric as opposed to distinguishing between symmetric and non-symmetric relations.

This analysis brings to light that our setup is unfair to BERT: it never sees evidence for non-symmetry. To address this, we define a new experiment, which we call ANTI because it includes an additional set of "anti" relations that are sampled from R^* with $R^* \cap R = \emptyset$ and $|R| = |R^*|$. ANTI facts take the following form: (e, r, f), (f, r, g) with $e \neq g$. Using this ANTI template we follow the standard data generation procedure. The corpus is now composed of symmetric, anti-symmetric and random facts. ANTI training data indicate to BERT that $r \in R^*$ is not symmetric since many instances of r facts are seen, with specific entities (f in the example) occurring in both slots, but there is never a symmetric example.

Table 2 (ANTI) shows that BERT memorizes ANTI facts seen during training but on test, BERT only recognizes 14.85% of ANTI facts as non-symmetric. So it still generalizes from the 250 symmetric relations to most other relations (85.15%), even those without any "symmetric" evidence in training. So it is easy for BERT to learn the concept of symmetry, but it is hard to teach it to distinguish between symmetric and non-symmetric relations.

Similar considerations apply to INV. BERT successfully predicts correct facts once it has learned that s and t are inverses – but it overgeneralizes by also predicting many incorrect facts; e.g., for (e, s, f) in train, it may predict (f, t, e) (correct), but also (e, t, f) and (f, s, e) (incorrect).

In another INV experiment, we add, for each pair of (f, r, e) and (e, s, f) two facts that give evidence of non-symmetry: (f, r, g) and (e, s, h) with $e \neq g$ and $h \neq f$. We find that test set precision for INV (i.e., inferring (e, s, f) in test from (f, r, e) in train) drops to 17% in this scenario. As for SYM, this indicates how complex the evaluation of rule learning is.

In summary, we have found that SYM and INV are learned in the sense that BERT generates correct facts for symmetric and inverse relations. But it severely overgenerates. Our analysis points to a problem of neural language models that has not received sufficient attention: they can easily learn that the order of arguments is not important (as is the case for SYM relations), but it is hard for them to learn that this is the case *only for a subset of relations*. Future work will have to delineate the exact scope of this finding – e.g., it may not hold for much larger training sets with millions of occurrences of each relation. Note, however, that human learning is likely to have a bias against symmetry in relations since the vast majority of verbs[2] in English (and presumably relations in the world) is asymmetric. So unless we have explicit evidence for symmetry, we are likely to assume a relation is non-symmetric. Our results suggest that neural language models do not have this bias – which would be problematic when using them for learning from natural language text.

4.1.2 Analysis of NEG

NEG was the only rule for which parameter tuning improved performance. A reduction to four layers obtained optimal results.

In Table 2 we report a test set precision of 20.54%. Why is negation more challenging than implication? Implication allows the model to generalize over several entities all following the same rule (e.g., every animal that is a mammal has a

[2]For example, almost all of the verb classes in (Levin, 1993) are asymmetric.

neocortex). This does not hold for negation (e.g., a leopard is fast but a snail is not fast). BERT must learn antonym negation from a large number of possible combinations. By reducing the number of possible combinations (decreasing the number of attributes from 1000 to 500, 250 and 125) BERT's test set precision increases, see Figure 3 (A). With 125 attributes a precision of 91% is reached. A reduction of attributes makes antonym negation very similar to implication.

We investigate BERT's behavior concerning negation further by adding an additional attribute set A^*, with $A^* \cap A = \emptyset$ and $|A| = |A^*|$ to the vocabulary. A^* does not follow an antonym schema. We sample $a \in A^*$, $e \in E$, $r \in R$ to add additional random facts of the type (e, r, a) or (e, not r, a) to NEG's training set. After training we test on the additional random facts seen during training by inserting or removing the negation marker. We see that BERT is prone to predict both (e, r, b) and (e, not r, b) for b $\in A^*$ (for 38%). Antonym negation was still learned.

We conclude that antonym negation can be learned via co-occurrences but a general concept of negation is not understood.

This is in agreement with prior work (Ettinger, 2020; Kassner and Schütze, 2020) showing that BERT trained on natural language corpora is as likely to generate a true statement like "birds can fly" as a factually false negated statement like "birds cannot fly".

4.1.3 Analysis of COMP

Why does BERT not learn COMP? COMP differs from the other rules in that it involves two-hop reasoning. Recall that a novelty of our experimental setup is that premises are not presented at inference time – two-hop reasoning requires that two different facts have to be "remembered" to make the inference, which intuitively is harder than a one-hop inference. Figure 3 (B) shows that the problem is not undertraining (orange line).

Similar to the memorization experiment, we investigate whether stronger semantic structure in form of a schema can make COMP learnable. We refer to this new experiment as **COMP enhanced**. Data generation is defined as follows: Entities are divided into groups of 10. Relations are now defined between groups in the sense that the members of a group are "equivalent". More formally, we sample entity groups (groups of 10) E_1, E_2, E_3 and relations r, s, t. For all $e_1 \in E_1, e_2 \in E_2, e_3 \in E_3$, we add (e_1, r, e_2) and (e_2, s, e_3) to C and (e_1, t, e_3) to D. In addition, we introduce a relation "samegroup" and add, for all $e_m, e_n \in E_i$, $(e_m, \text{samegroup}, e_n)$ to C – this makes it easy to learn group membership. As before, the training set is the merger of C and 90% of D and the test set is the rest of D.

Similar semantic structures occur in real data. The simplest case is a transitive example: (r) planes (group 1) are faster than cars (group 2), (s) cars (group 2) are faster than bikes (group 3), (t) planes (group 1) are faster than bikes (group 3).

Figure 3 (B) shows that BERT can learn COMP moderately well from this schema-enhanced corpus (blue curve): precision is clearly above 50% and peaks at 76%.

The takeaway from this experiment is that two-hop rules pose a challenge to BERT, but that they are learnable if entities and relations are embedded in a rich semantic structure. Prior work (Brown et al., 2020) has identified the absence of "domain models" (e.g., a domain model for common sense physics) as one shortcoming of PLMs. To the extent that PLMs lack such domain knowledge (which we simulate here with a schema), they may not be able to learn COMP.

4.2 Natural Language Corpora

In this section, we investigate to what extent the PLMs BERT and RoBERTa have learned SYM and INV from natural language corpora. See Table 3. For "smaller/larger" (INV), we follow Talmor et al. (2019) and test which of the two words is selected as the more likely filler in a pattern like "Jupiter is [MASK] than Mercury". For the other three relations ("shares borders with" (SYM), "is the opposite of" (SYM), "is the capital of" / "'s capital is" (INV)), we test whether the correct object is predicted in the pattern "e r [MASK]" (as in the rest of the paper). We give the number of (i) consistent ("cons."), (ii) correct and consistent ("correct") and (iii) inconsistent ("inc.") predictions. (A prediction is consistent and incorrect if it is consistent with the rule, but factually incorrect.)

In more detail, we take a set of entities (countries like "Indonesia", cities like "Jakarta") or adjectives like "low" that are appropriate for the relation and test which of the entities / adjectives is predicted. For each of the five relations, we run both BERT-large-cased and RoBERTa-large and report the more consistent result.

relation	rule	completions cons.	correct	inc.	examples
shares borders with	SYM	152	152	2	(ecuador,peru) (togo,ghana), (ghana,nigeria)
is the opposite of	SYM	179	170	71	(demand,supply) (injustice,justice), (justice,truth)
is the capital of (C-of) *'s capital is (s-C-is)*	INV	59	59	1	(indonesia,s-C-is,jakarta) (canada,s-C-is,ottawa), (ottawa,C-of,ontario)
is smaller/larger than (countries)	INV	54	23	99	(russia,larger,canada), (canada,smaller,russia) (brazil,smaller,russia), (russia,smaller,brazil)
is smaller/larger than (planets)	INV	9	9	36	(jupiter,larger,mercury), (mercury,smaller,jupiter) (sun,bigger,earth), (earth,bigger,sun)

Table 3: Can PLMs (BERT and RoBERTa) learn SYM and INV from natural language corpora? For "smaller/larger", we follow Talmor et al. (2019) and test which of the two words is selected as a filler in a pattern like "Jupiter is [MASK] than Mercury". For the other three relations, we test whether the correct object is predicted (as in the rest of the paper). We give the number of (i) consistent ("cons."), (ii) correct and consistent ("correct") and (iii) inconsistent ("inc.") predictions. Blue: consistent examples. Red: inconsistent examples. (We make the simplifying assumption that "justice" can only have one opposite.)

Consistency and accuracy are high for "shares borders with" and "capital". However, this is most likely due to the fact that many of these facts occur verbatim in the training corpora of the two models. For example, Google shows 54,800 hits for "jakarta is the capital of indonesia" and 1,290 hits for "indonesia's capital is jakarta" (both as a phrase). It is not possible to determine which factor is decisive here: successful rule-based inference or memorization. The ultimate futility of this analysis is precisely the reason that we chose to work with synthetic data.

Consistency for "is the opposite of" is much lower than for the first two relations, but still decent. To investigate this relation further, we also tested the relation "is the same as". It turns out that many of the "opposite" objects are also predicted for "is the same as", e.g., "high is the same as *low*" and "low is the same as *high*" where the predicted word is in italics. This indicates that the models have not really learned that "is the opposite of" is symmetric, but rather know that antonyms are closely associated and often occur together in phrases like "X is the opposite of Y", "X and Y", "X noun, Y noun" (e.g., "good cop, bad cop") etc. Apparently, this is then incorrectly generalized to "is the same as".

Consistency and accuracy are worse for "smaller/larger". "smaller/larger" sentences of the sort considered here are probably rarer in genres like Wikipedia than "shares borders with" and "is the capital of". A Wikipedia article about a country will always say what its capital is and which countries it borders, but it will not enumerate the countries that are smaller or larger.

In summary, although we have shown that pretrained language models have some ability to learn symbolic rules, there remains considerable doubt that they can do so based on natural corpora.

4.3 Memorization

Experimental results for the memorization experiments are shown in Figure 4.

(A) shows that frequent facts are memorized well (0.8 for frequency 100) and that rare facts are not (≈ 0.0 for frequencies < 15).

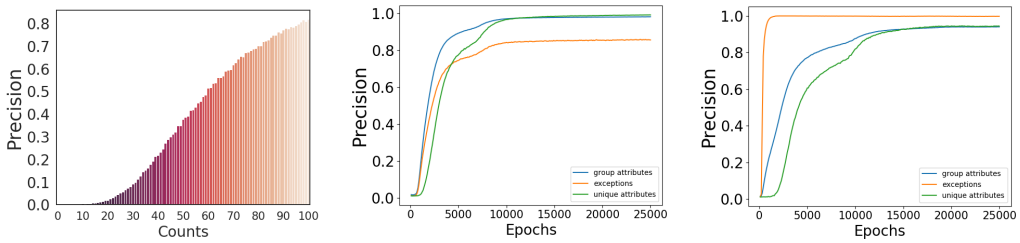
(B) shows that BERT memorizes schema conformant facts perfectly ("group attributes"). Accuracy for exceptions is clearly lower than those of schema conformant facts: about 80%. The frequency of each fact in the training corpus in this experiment is 1. Overall, the total amount of exceptions is much lower than the total amount of schema conformant facts.

(C) shows that exceptions are perfectly learned if 10 copies of each exception are added to the corpus – instead of 1 in (B). In this case, limited capacity affects memorization of schema-conformant facts: accuracy drops to ≈ 0.9.

In summary, we find that both frequency and schema conformity facilitate memorization. Schema conformant facts and exceptions compete for memory if memory capacity is limited – depending on frequency one or the other is preferentially learned by BERT.

5 Limitations

Our experimental design makes many simplifying assumptions: i) Variation in generated data is more



(A) frequency

(B) schema conformity, exceptions rare

(C) schema conformity, exceptions frequent

Figure 4: Memorization experiments. We investigate the effect of frequency and schema conformity on memorization. (A) Frequent facts are memorized well (0.8 for frequency 100), rare facts are not (≈ 0.0 for frequencies < 15). (B) BERT memorizes schema conformant facts perfectly ("group attributes"). Accuracy for rare exceptions is clearly lower (80%). (C) Exceptions are perfectly learned if 10 copies of each exception are added to the corpus – instead of 1 in (B). In this case, limited capacity affects memorization of schema-conformant facts ("group attributes" drops to ≈ 0.9).

limited than in naturally occurring data. ii) Semantics are deliberately restricted to one rule only per generated corpus. iii) We do not investigate effects of model and corpus size.

i) In natural corpora relations can have more than two arguments, entities can have several tokens, natural data are noisier than synthetic data etc. Also, we study each rule in isolation.

ii) While our simplified corpora make learning easier in some respects, they may make it harder in others. Each corpus is focused on providing training material for one symbolic rule, but it does not contain any other "semantic" signal that may be helpful in learning symbolic reasoning: distributional signals, entity groupings, hierarchies, rich context etc. The experimental results of "COMP enhanced" indicate that indeed such signals are beneficial to symbolic rule learning. The interplay of such additional sources of information for learning with symbolic rules is an interesting question for follow up work.

iii) Results are based on BERT-base and scaled-down versions of BERT-base only, just as training corpora are orders of magnitude smaller than natural training corpora. We varied model and corpus sizes within the limits of our compute infrastructure, but did not systematically study their effect on our findings.

Our work is an initial exploration of the question whether symbolic rules can be learned in principle, but we view it mainly as a starting point for future work.

6 Related Work

Radford et al. (2019) and Petroni et al. (2019) show in a zero-shot question answering setting that PLMs have factual knowledge. Our main question is: under what conditions do PLMs learn factual knowledge and do they do so through memorization or rule-based inference?

Sun et al. (2019) and Zhang et al. (2020) show in the knowledge graph domain that models that have the ability to capture symbolic rules like SYM, INV and COMP outperform ones that do not. We investigate this question for PLMs that are trained on language corpora.

Talmor et al. (2019) test PLMs' symbolic reasoning capabilities probing pretrained and finetuned models with cloze-style queries. Their setup makes it impossible to distinguish whether a fact was inferred or memorized during pretraining. Our synthetic corpora allow us to make this distinction.

Clark et al. (2020) test finetuned BERT's reasoning capabilities, but they always make premise and conclusion locally available to the model, during training and inference. This is arguably not the way much of human inference works; e.g., the fact F that X borders Y allows us to infer that Y borders X even if we were exposed to F a long time ago.

Richardson et al. (2020) introduce synthetic corpora testing logic and monotonicity reasoning. They show that BERT performs poorly on these new datasets, but can be quickly finetuned to good performance. The difference to our work again is that they make the premise available to the model at inference time.

For complex reasoning QA benchmarks (Yang

et al., 2018; Sinha et al., 2019), PLMs are finetuned to the downstream tasks. Their performance is difficult to analyze: it is not clear whether any reasoning capability is learned by the PLM or by the task specific component.

Another line of work (Gururangan et al., 2018; Kaushik and Lipton, 2018; Dua et al., 2019; McCoy et al., 2019) shows that much of PLMs' performance on reasoning tasks is due to statistical artifacts in datasets and does not exhibit true reasoning and generalization capabilities. With the help of synthetic corpora, we can cleanly investigate PLMs' reasoning capabilities.

Hupkes et al. (2020) study the ability of neural models to capture compositionality. They do not investigate our six rules, nor do they consider the effects of fact frequency and schema conformity. Our work confirms their finding that transformers have the ability to capture both rules and exceptions.

A large body of research in psychology and cognitive science has investigated how some of our rules are processed in humans, e.g., Sloman (1996) for implication. There is also a lively debate in cognitive science as to how important rule-based reasoning is for human cognition (Politzer, 2007).

Yanaka et al. (2020); Goodwin et al. (2020) are concurrent studies of systematicity in PLMs. The first shows that monotonicity inference is feasible for syntactic structures close to the ones observed during training. The latter shows that PLMs can exhibit high over-all performance on natural language inference despite being non-systematic.

Roberts et al. (2020) show that the amount of knowledge captured by PLMs increases with model size. Our memorization experiments investigate the factors that determine successful acquisition of knowledge.

Guu et al. (2020) modify the PLM objective to incentivize knowledge acquisition. They do not consider symbolic rule learning nor do they analyze what factors influence successful memorization.

Based on perceptrons and convolutional neural networks, Arpit et al. (2017); Zhang et al. (2017) study the relation of generalizing from real structured data vs. memorizing random noise in the image domain, similar to our study of schema-conformant facts and outliers. They do not study transformer based models trained on natural language.

7 Conclusion

We studied BERT's ability to capture knowledge from its training corpus by investigating its reasoning and memorization capabilities. We identified factors influencing what makes successful memorization possible and what is learnable beyond knowledge explicitly seen during training. We saw that, to some extent, BERT is able to infer facts not explicitly seen during training via symbolic rules.

Overall, effective knowledge acquisition must combine both parts of this paper: memorization and symbolic reasoning. A PLM is not able to store an unlimited amount of knowledge. Through acquiring reasoning capabilities, knowledge gaps can be filled based on memorized facts. A schema-conformant fact ("pigeons can fly") need not be memorized if there are a few facts that indicate that birds fly and then the ability of flight can be filled in for the other birds. The schema conformity experiments suggest that this is happening. It is easier to capture knowledge that conforms with a schema instead of memorizing facts one by one.

There are several directions for future work. First, we made many simplifying assumptions that should be relaxed in future work. Second, how can we improve PLMs' ability to learn symbolic rules? We see two avenues here, either additional inductive biases could be imposed on PLMs' architectures or training corpora could be modified to promote learning of symbolic rules.

Acknowledgements

We thank Peter Clark for helpful discussions and our reviewers for constructive feedback.

This work was funded by the German Federal Ministry of Education and Research (BMBF, Grant No. 01IS18036A) and by Deutsche Forschungsgemeinschaft (DFG, Grant ReMLAV: Relational Machine Learning for Argument Validation). The authors of this work take full responsibility for its content.

References

Devansh Arpit, Stanisław Jastrzundefinedbski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. 2017. A closer look at memorization in deep networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 233242. JMLR.org.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners.

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. *IJCAI*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *NAACL-HLT*.

Allyson Ettinger. 2020. What bert is not: Lessons from a new suite of psycholinguistic diagnostics for language models. *Transactions of the Association for Computational Linguistics*, 8:34–48.

Emily Goodwin, Koustuv Sinha, and Timothy J. O'Donnell. 2020. Probing linguistic systematicity. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1958–1969, Online. Association for Computational Linguistics.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 107–112, New Orleans, Louisiana. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. *ArXiv*, abs/2002.08909.

Dieuwke Hupkes, Verna Dankers, Mathijs Mul, and Elia Bruni. 2020. Compositionality decomposed: How do neural networks generalise? *J. Artif. Intell. Res.*, 67:757–795.

Nora Kassner and Hinrich Schütze. 2020. Negated and misprimed probes for pretrained language models: Birds can talk, but cannot fly. *ACL*.

Divyansh Kaushik and Zachary C. Lipton. 2018. How much reading does reading comprehension require? a critical investigation of popular benchmarks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5010–5015, Brussels, Belgium. Association for Computational Linguistics.

Beth Levin. 1993. *English Verb Classes and Alternations*. The University of Chicago Press, Chicago.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Tom McCoy, Ellie Pavlick, and Tal Linzen. 2019. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy. Association for Computational Linguistics.

Mojtaba Nayyeri, Chengjin Xu, Jens Lehmann, and Hamed Shariat Yazdi. 2019. Logicenn: A neural based knowledge graphs embedding model with logical rules. *ArXiv*, abs/1908.07141.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Guy Politzer. 2007. Reasoning with conditionals. *Topoi*, 26(1):79–95.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Kyle Richardson, Hai Hu, Lawrence S. Moss, and Ashish Sabharwal. 2020. Probing natural language inference models through semantic fragments. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? *ArXiv*, abs/2002.08910.

Koustuv Sinha, Shagun Sodhani, Jin Dong, Joelle Pineau, and William L. Hamilton. 2019. CLUTRR: A diagnostic benchmark for inductive reasoning from text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*

and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 4506–4515, Hong Kong, China. Association for Computational Linguistics.

Steven A. Sloman. 1996. The empirical case for two systems of reasoning. *Psychological Bulletin*, 119(1):3–22.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations*.

Alon Talmor, Yanai Elazar, Yoav Goldberg, and Jonathan Berant. 2019. oLMpics - on what language model pre-training captures. *ArXiv*, abs/1912.13283.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2016. Towards ai-complete question answering: A set of prerequisite toy tasks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771.

Hitomi Yanaka, Koji Mineshima, Daisuke Bekki, and Kentaro Inui. 2020. Do neural models learn systematicity of monotonicity inference in natural language? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6105–6117, Online. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2017. Understanding deep learning requires rethinking generalization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24 - 26, 2017, Conference Track Proceedings*.

Zhanqiu Zhang, Jianyu Cai, Yongdong Zhang, and Jie Wang. 2020. Learning hierarchy-aware knowledge graph embeddings for link prediction. In *Thirty-Fourth AAAI Conference on Artificial Intelligence*.

A Hyperparameters

A.1 Model hyperparameters

For all reported results we trained with a batch-size of 1024 and a learning rate of 6e-5.

Our experiments for symbolic rules started with the BERT-base model with 12 layers, 12 attention heads, hidden size of 768 and intermediate size of 3072. For rules with a low test precision (NEG and COMP) we then conducted a restricted grid search (restricted due to limited compute infrastructure): We tried all possible numbers of layers from 1 to 12 and then only considered the best result. For NEG the best performance came from 4 layers, whereas COMP did not show improvements for any number of layers. For NEG with 3 layers (which had a very similar performance to 4 layers) we exemplarily tested whether changing the attention heads, hidden size or intermediate size improves precision. For this we trained with the following 4 settings:

- attention heads = 6, hidden size = 768, intermediate size = 3072
- attention heads = 12, hidden size = 384, intermediate size = 1536
- attention heads = 12, hidden size = 192, intermediate size = 768
- attention heads = 12, hidden size = 96, intermediate size = 192

However this did not further improve precision.

A.2 Data hyperparameters

In previous iterations of our experiments, we had used different settings for generating our data. For instance, we had varied the number of rules in our corpora: 50 or 100 instead of the presented 20 rules. Even the sampling process itself can be tweaked to allow for less overlaps between rules and between instances of one rule. However, we observed the same trends and similar numbers across these different settings.

B Symbolic rules

In the following sections, we present illustrating corpora for **INV**, **IMP** and **COMP enhanced**. Each line is one datapoint. We also include the control group at the end of each corpus that does not follow any rule. In the case of composition enhanced, "{...}" indicates the sampled group which is not part of the actual dataset.

We illustrate our training corpora using real world entities and relations. Note that the actual corpora used for training are composed of an entirely synthetic vocabulary. For simplicity we show grouped composition with enhancement with groups of 4, instead of 10 as it is in the real data.

B.1 INV

Paris CapitalOf France
France HasCapital Paris
...
Egypt HasCapital Cairo (counterpart in test-set)
...
Apple Developed iOS
iOS DevelopedBy Apple
...
Germany RandomRelation China
Cairo RandomRelation Norway

B.2 IMP

{(Flu), (Cough, RunningNose, Headache, Fever)}
Kevin HasDisease Flu
Kevin HasSymptom Cough
Kevin HasSymptom RunningNose
Kevin HasSymptom Headache
Kevin HasSymptom Fever
...
Mariam HasDisease Flu
...
Peter RandomRelation Pain
Sarah RandomRelation Tooth

B.3 Comp enhanced

{e8, e2, e4, e5}
e8 ConnectedTo e2
e8 ConnectedTo e4
e8 ConnectedTo e5
e2 ConnectedTo e8
...
{e15, e13, e12, e19}
e15 ConnectedTo e13
e15 ConnectedTo e12
...
{e25, e24, e29, e20}
e25 ConnectedTo e24
e25 ConnectedTo e29
...
e8 r1 e15
e8 r1 e13
e8 r1 e12
e8 r1 e19
e2 r1 e15
e2 r1 e13
...
e5 r1 e19
...
e15 r2 e25
e15 r2 e24
e15 r2 e29
e15 r2 e20
...
e19 r2 e20
...
e8 r3 e25
e8 r3 e24
e8 r3 e29
e8 r3 e20
...
...
e133 r61 e23
e56 r61 e29
...

Chapter 8

BERT-kNN: Adding a kNN search component to pretrained language models for better QA

BERT-kNN: Adding a kNN Search Component to Pretrained Language Models for Better QA

Nora Kassner, Hinrich Schütze
Center for Information and Language Processing (CIS)
LMU Munich, Germany
kassner@cis.lmu.de

Abstract

Khandelwal et al. (2020) use a k-nearest-neighbor (kNN) component to improve language model performance. We show that this idea is beneficial for open-domain question answering (QA). To improve the recall of facts encountered during training, we combine BERT (Devlin et al., 2019) with a traditional information retrieval step (IR) and a kNN search over a large datastore of an embedded text collection. Our contributions are as follows: i) BERT-kNN outperforms BERT on cloze-style QA by large margins without any further training. ii) We show that BERT often identifies the correct response category (e.g., US city), but only kNN recovers the factually correct answer (e.g., "Miami"). iii) Compared to BERT, BERT-kNN excels for rare facts. iv) BERT-kNN can easily handle facts not covered by BERT's training set, e.g., recent events.

1 Introduction

Pretrained language models (PLMs) like BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019) and RoBERTa (Liu et al., 2019) have emerged as universal tools that not only capture a diverse range of linguistic, but also (as recent evidence seems to suggest) factual knowledge.

Petroni et al. (2019) introduced LAMA (LAnguage Model Analysis) to test BERT's performance on open-domain QA and therefore investigate PLMs' capacity to recall factual knowledge without the use of finetuning. Since the PLM training objective is to predict masked tokens, question answering tasks can be reformulated as cloze questions; e.g., "Who wrote 'Ulysses'?" is reformulated as "[MASK] wrote 'Ulysses'." In this setup, Petroni et al. (2019) show that, on QA, PLMs outperform baselines trained on automatically extracted knowledge bases (KBs).

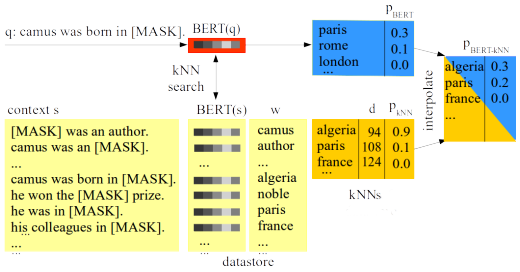


Figure 1: BERT-kNN interpolates BERT's prediction for question q with a kNN-search. The kNN search runs in BERT's embedding space, comparing the embedding of q with the embeddings of a retrieved subset of a large text collection: Pairs of a word w in the text collection and the BERT embedding of w's context ($BERT(s)$) are stored in a key-value datastore. An IR step is used to define a relevant subset of the full datastore (yellow). $BERT(q)$ (red) is BERT's embedding of the question. The kNN search runs between $BERT(q)$ and $BERT(s)$ and the corresponding distance d and word w is returned (orange). Finally, BERT's predictions (blue) are interpolated with this kNN search result.

Still, given that PLMs have seen more text than humans read in a lifetime, their performance on open-domain QA seems poor. Also, many LAMA facts that PLMs do get right are not "recalled" from training, but are guesses instead (Poerner et al., 2019). To address PLMs' poor performance on facts and choosing BERT as our PLM, we introduce BERT-kNN.

BERT-kNN combines BERT's predictions with a kNN search. The kNN search runs in BERT's embedding space, comparing the embedding of the question with the embeddings of a retrieved subset of a large text collection. The text collection can be BERT's training set or any other suitable text corpus. Due to its kNN component and its resulting ability to directly access facts stated in the searched text, BERT-kNN outperforms BERT on cloze-style

Dataset	BERT-base	BERT-large	ERNIE	Know-BERT	E-BERT	BERT-kNN
LAMA	27.7	30.6	30.4	31.7	36.2	**39.4**
LAMA-UHN	20.6	23.0	24.7	24.6	31.1	**34.8**

Table 1: Mean P@1 on LAMA and LAMA-UHN on the TREx and GoogleRE subsets for BERT-base, BERT-large, ERNIE (Zhang et al., 2019), KnowBert (Peters et al., 2019), E-BERT (Poerner et al., 2019) and BERT-kNN. BERT-kNN performs best.

QA by large margins.

A schematic depiction of the model is shown in Figure 1. Specifically, we use BERT to embed each token's masked context s in the text collection ($BERT(s)$). Each pair of context embedding and token is stored as a key-value pair in a datastore. Testing for a cloze question q, the embedding of q ($BERT(q)$) serves as query to find the k context-target pairs in the subset of the datastore that are closest. The final prediction is an interpolation of the kNN search and the PLM predictions.

We find that the kNN search over the full datstore alone does not obtain good results. Therefore, we first query a separate information retrieval (IR) index with the original question q and only search over the most relevant subset of the full datastore when finding the k-nearest-neighbors of $BERT(q)$ in embedding space.

We find that the PLM often correctly predicts the answer category and therefore the correct answer is often among the top k-nearest-neighbors. A typical example is "Albert Einstein was born in [MASK]": the PLM knows that a city is likely to follow and maybe even that it is a German city, but it fails to pick the correct city. On the other hand, the top-ranked answer in the kNN search is "Ulm" and so the correct filler for the mask can be identified.

BERT-kNN sets a new state-of-the-art on the LAMA cloze-style QA dataset without any further training. Even though BERT-kNN is based on BERT-base, it also outperforms BERT-large. The performance gap between BERT and BERT-kNN is most pronounced on hard-to-guess facts. Our method can also make recent events available to BERT without any need of retraining: we can simply add embedded text collections covering recent events to BERT-kNN's datastore.

The source code of our experiments is available under: https://github.com/norakassner/BERT-kNN.

2 Data

The LAMA dataset is a cloze-style QA dataset that allows to query PLMs for facts in a way analogous to KB queries. A cloze question is generated using a subject-relation-object triple from a KB and a templatic statement for the relation that contains variables X and Y for subject and object; e.g, "X was born in Y". The subject is substituted for X and [MASK] for Y. In all LAMA triples, Y is a single-token answer.

LAMA covers different sources: The GoogleRE[1] set covers the relations "place of birth", "date of birth" and "place of death". TREx (ElSahar et al., 2018) consists of a subset of Wikidata triples covering 41 relations. ConceptNet (Li et al., 2016) combines 16 commonsense relations among words and phrases. The underlying Open Mind Common Sense corpus provides matching statements to query the language model. SQuAD (Rajpurkar et al., 2016) is a standard question answering dataset. LAMA contains a subset of 305 context-insensitive questions. Unlike KB queries, SQuAD uses manually reformulated cloze-style questions which are not based on a template.

We use SQuAD and an additional 305 ConceptNet queries for hyperparamter search.

Poerner et al. (2019) introduce LAMA-UHN, a subset of LAMA's TREx and GoogleRE questions from which easy-to-guess facts have been removed.

To test BERT-kNN's performance on unseen facts, we collect Wikidata triples containing TREx relations from Wikipedia pages created January–May 2020 and add them to the datastore.

3 Method

BERT-kNN combines BERT with a kNN search component. Our method is generally applicable to PLMs. Here, we use BERT-base-uncased (Devlin et al., 2019). BERT is pretrained on the BookCorpus (Zhu et al., 2015) and the English Wikipedia.

Datastore. Our text collection C is the 2016-12-21 English Wikipedia.[2] For each single-token word occurrence w in a sentence in C, we com-

[1] https://code.google.com/archive/p/relation-extraction-corpus/

[2] dumps.wikimedia.org/enwiki

Dataset	Statistics		Model		
	Facts	Rel	BERT	kNN	BERT -kNN
GoogleRE	5527	3	9.8	**51.1**	48.6
TREx	34039	42	29.1	34.4	**38.7**
ConceptNet	11153	16	**15.6**	4.7	11.6
SQuAD	305	-	14.1	**25.5**	24.9
unseen	34637	32	18.8	21.5	**27.1**

Table 2: Mean P@1 for BERT-base, kNN and their interpolation (BERT-kNN) for LAMA subsets and unseen facts. BERT results differ from Petroni et al. (2019) where a smaller vocabulary is used.

Configuration	P@1
hidden layer 12	36.8
hidden layer 11	**39.4**
hidden layer 10	34.7
hidden layer 11 (without IR)	26.9

Table 3: Mean P@1 on LAMA (TREx, GoogleRE subsets) for different context embedding strategies. Top: The context embedding is represented by the embedding of the masked token in different hidden layers. Best performance is obtained using BERT's hidden layer 11. Bottom: We show that BERT-kNN's performance without the additional IR step drops significantly. We therefore conclude that the IR step is an essential part of BERT-kNN.

pute the pair (c, w) where c is a context embedding computed by BERT. To be specific, we mask the occurrence of w in the sentence and use the embedding of the masked token. We store all pairs (c, w) in a key-value datastore D where c serves as key and w as value.

Information Retrieval. We find that just using the datastore D does not give good results (see result section). We therefore use Chen et al. (2017)'s IR system to first select a small subset of D using a keyword search. The IR index contains all Wikipedia articles. An article is represented as a bag of words and word bigrams. We find the top 3 relevant Wikipedia articles using TF-IDF search. For KB queries, we use the subject to query the IR index. If the subject has its dedicated Wikipedia page, we simply use this. For non-knowledge base queries, we use the cloze-style question q ([MASK] is removed).

Inference. During testing, we first run the IR search to identify the subset D' of D that corresponds to the relevant Wikipedia articles. For the kNN search, q is embedded in the same way as the context representations c in D: we set $BERT(q)$ to the embedding computed by BERT for [MASK]. We then retrieve the $k = 128$ nearest-neighbors of

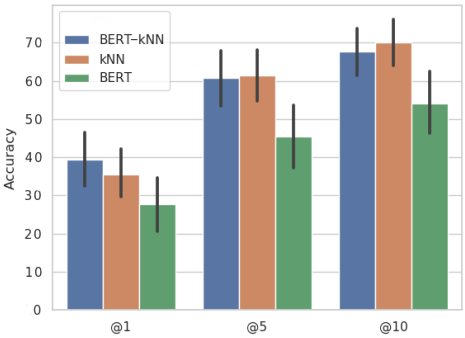


Figure 2: Mean P@1, P@5, P@10 on LAMA for original BERT and BERT-kNN.

$BERT(q)$ in D'. We convert the distances (Euclidean) between $BERT(q)$ and the kNNs to a probability distribution using softmax. Since a word w can occur several times in kNN, we compute its final output probability as the sum over all occurrences.

In the final step, we interpolate kNN's (weight 0.3) and BERT's original predictions (weight 0.7). We optimize hyperparameters on dev. See supplementary for details.

Evaluation. Following Petroni et al. (2019) we report mean precision at rank r (P@r). P@r is 1 if the top r predictions contain the correct answer, otherwise it returns 0. To compute mean precision, we first average within each relation and then across relations.

4 Results and Discussion

Table 1 shows that BERT-kNN outperforms BERT on LAMA. It has about 10 precision point gain over BERT, base and large. Recall that BERT-kNN uses BERT-base. The performance gap between original BERT and BERT-kNN becomes even larger when evaluating on LAMA-UHN, a subset of LAMA with hard-to-guess facts.

It also outperforms entity-enhanced versions of BERT (see related work) – ERNIE (Zhang et al., 2019), KnowBert (Peters et al., 2019) and E-BERT (Poerner et al., 2019) – on LAMA.

Table 2 shows that BERT-kNN outperforms BERT on 3 out of 4 LAMA subsets. BERT prevails on ConceptNet; see discussion below. Huge gains are obtained on the GoogleRE dataset. Figure 2 shows precision at 1, 5 and 10. BERT-kNN performs better than BERT in all three categories.

Table 3 compares different context embedding strategies. BERT's masked token embedding of

	Query and True Answer	Generation
Google RE	hans gefors was born in [MASK]. True: stockholm	BERT-kNN: stockholm (0.36), oslo (0.15), copenhagen (0.13) BERT: oslo (0.22), copenhagen (0.18), bergen (0.09) kNN: stockholm (1.0), lund (0.00), hans (0.00)
TREx	regiomontanus works in the field of [MASK]. True: mathematics	BERT-kNN: astronomy (0.20), mathematics (0.13), medicine (0.06) BERT: medicine (0.09), law (0.05), physics (0.03) kNN: astronomy (0.63), mathematics (0.36), astronomical (0.00)
Concept Net	ears can [MASK] sound. True: hear	BERT-kNN: hear (0.27), detect (0.23), produce (0.06) BERT: hear (0.28), detect (0.06), produce (0.04) kNN: detect (0.77), hear (0.14), produce (0.10)
Squad	tesla was in favour of the [MASK] current type. True: ac	BERT-kNN: alternating (0.39), electric (0.18), direct (0.11) BERT: electric (0.28), alternating (0.18), direct (0.11) kNN: alternating (0.87), direct (0.12), ac (0.00)

Table 4: Examples of generation for BERT-base, kNN, BERT-kNN. The last column reports the top three tokens generated together with the associated probability (in parentheses).

hidden layer 11 performs best. We also show the necessity of the IR step by running a kNN search over all Wikipedia contexts, which results in precision lower than original BERT. To run an efficient kNN search over all contexts instead of the relevant subset identified by the IR step, we use the FAISS libary (Johnson et al., 2017).

Table 2 also shows that neither BERT nor kNN alone are sufficient for top performance, while the interpolation of the two yields optimal results. In many cases, BERT and kNN are complementary. kNN is worse than BERT on ConceptNet, presumably because commonsense knowledge like "birds can fly" is less well-represented in Wikipedia than entity triples and also because relevant articles are harder to find by IR search. We keep the interpolation parameter constant over all datasets. Table 4 shows that kNN often has high confidence for correct answers – in such cases it is likely to dominate less confident predictions by BERT. The converse is also true (not shown). Further optimization could be obtained by tuning interpolation per dataset.

BERT-kNN answers facts unseen during pretraining better than BERT, see Table 2. BERT was not trained on 2020 events, so it must resort to guessing. Generally, we see that BERT's knowledge is mainly based on guessing as it has seen Wikipedia during training but is not able to recall the knowledge recovered by kNN.

Table 4 gives examples for BERT and BERT-kNN predictions. We see that BERT predicts the answer category correctly, but it often needs help from kNN to recover the correct entity within that category.

5 Related work

PLMs are top performers for many tasks, including QA (Kwiatkowski et al., 2019; Alberti et al., 2019; Bosselut et al., 2019). Petroni et al. (2019) introduced the LAMA QA task to probe PLMs' knowledge of facts typically modeled by KBs.

The basic idea of BERT-kNN is similar to Khandelwal et al. (2020)'s interpolation of a PLM and kNN for language modeling. In contrast, we address QA. We introduce an IR step into the model that is essential for good performance. Also, our context representations differ as we use embeddings of the masked token.

Grave et al. (2016) and Merity et al. (2017), inter alia, also make use of memory to store hidden states. They focus on recent history, making it easier to copy rare vocabulary items.

DRQA (Chen et al., 2017) is an open-domain QA model that combines an IR step with a neural reading comprehension model. We use the same IR module, but our model differs significantly. DRQA does not predict masked tokens, but extracts answers from text. It does not use PLMs nor a kNN module. Most importantly, BERT-kNN is fully unsupervised and does not require any extra training.

Some work on knowledge in PLMs focuses on injecting knowledge into BERT's encoder. ERNIE (Zhang et al., 2019) and KnowBert (Peters et al., 2019) are entity-enhanced versions of BERT. They introduce additional encoder layers that are integrated into BERT's original encoder by expensive additional pretraining. Poerner et al. (2019) injects factual entity knowledge into BERT's embeddings without pretraining by aligning Wikipedia2Vec entity vectors (Yamada et al., 2016) with BERT's wordpiece vocabulary. This approach is also limited to labeled entities. Our approach on the other hand is not limited to labeled entities nor does it require any pretraining. Our approach is conceptually different from entity-enhanced versions of BERT and could potentially be combined with them for

even better performance. Also, these models address language modeling, not QA.

The combination of PLMs with an IR step/kNN search has attracted a lot of recent research interest. The following paragraph lists concurrent work:

Petroni et al. (2020) also combine BERT with an IR step to improve cloze-style QA. They do not use a kNN search nor an interpolation step but feed the retrieved contexts into BERT's encoder. Guu et al. (2020) augment PLMs with a latent knowledge retriever. In contrast to our work they continue the pretraining stage. They jointly optimize the masked language modeling objective and backpropagate through the retrieval step. Lewis et al. (2020); Izacard and Grave (2020) leverage retrieved contexts for better QA using finetuned generative models. They differ in that the latter fuse evidence of multiple contexts in the decoder. Joshi et al. (2020) integrate retrieved contexts into PMLs for better reading comprehension.

6 Conclusion

This work introduced BERT-kNN, an interpolation of BERT predictions with a kNN search for unsupervised cloze-style QA. BERT-kNN sets a state-of-the-art on LAMA without any further training. BERT-kNN can be easily enhanced with knowledge about new events that are not covered in the training text used for pretraining BERT.

In future work, we want to exploit the utility of the kNN component for explainability: kNN predictions are based on retrieved contexts, which can be shown to users to justify an answer.

Acknowledgements

This work has been funded by the German Federal Ministry of Education and Research (BMBF) under Grant No. 01IS18036A. The authors of this work take full responsibility for its content.

References

Chris Alberti, Kenton Lee, and Michael Collins. 2019. A BERT baseline for the natural questions. *ArXiv*, abs/1901.08634.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Hady ElSahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon S. Hare, Frédérique Laforest, and Elena Simperl. 2018. T-rex: A large scale alignment of natural language with knowledge base triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*.

Edouard Grave, Armand Joulin, and Nicolas Usunier. 2016. Improving neural language models with a continuous cache. *ICLR*, abs/1612.04426.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Mandar Joshi, Kenton Lee, Yi Luan, and Kristina Toutanova. 2020. Contextualized representations using textual encyclopedic knowledge. *arXiv preprint arXiv:2004.12006*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through Memorization: Nearest Neighbor Language Models. In *International Conference on Learning Representations (ICLR)*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Patrick Lewis, Ethan Perez, Aleksandara Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Kttler, Mike Lewis, Wen tau Yih, Tim Rocktschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. 2016. Commonsense knowledge base completion. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1445–1455, Berlin, Germany. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. In *International Conference on Learning Representations (ICLR)*.

Matthew E. Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A. Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 43–54, Hong Kong, China. Association for Computational Linguistics.

Fabio Petroni, Patrick Lewis, Aleksandra Piktus, Tim Rocktäschel, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2020. How context affects language models' factual predictions. In *Automated Knowledge Base Construction*.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. 2019. Bert is not a knowledge base (yet): Factual knowledge vs. name-based reasoning in unsupervised qa. *ArXiv*, abs/1911.03681.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Ikuya Yamada, Hiroyuki Shindo, Hideaki Takeda, and Yoshiyasu Takefuji. 2016. Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 250–259, Berlin, Germany. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *The IEEE International Conference on Computer Vision (ICCV)*.

A Data

LAMA and LAMA-UHN can be downloaded from:
https://dl.fbaipublicfiles.com/LAMA/

For TREx unseen, we downloaded the latest Wikidata and Wikipedia dump from:
https://dumps.wikimedia.org/wikidatawiki/entities/wikipedia_en/latest-all.json.bz2
and
https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2.

We filter for TREx relations and only consider facts which have a Wikipedia page created after January 1st 2020. We only consider relations with 5 questions or more. We add the additional embedded Wikipedia articles to the datastore.

B Inference

The probability of the kNN search for word w is given by:
$p_{kNN}(w \mid q) \sim \sum_{(c_w,w)\in kNN} e^{-d(BERT(q),c_w)/l}$.

The final probability of BERT-kNN is the interpolation of the predictions of BERT and the kNN search:
$p_{BERT-kNN}(q) = \lambda p_{kNN}(q) + (1-\lambda) p_{BERT}(q)$,

with
q question,
$BERT(q)$ embedding q,
w target word,
s_w context of w,
$c_w = BERT(s)$ embedded context,
d distance,
l distance scaling,
λ interpolation parameter.

C Hyperparameters

Hyperparameter optimization is done with the 305 SQuAD questions and additional randomly sampled 305 ConceptNet questions. We remove the 305 ConceptNet questions from the test set. We run the hyperparameter search once.
We run a grid search for the following hyperparameters:
Number of documents N = [1, 2, 3, 4, 5],
Interpolation λ = [0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8],
Number of NN k = [64, 128, 512],
Distance scaling l = [5, 6, 7, 8, 9, 10, 11, 12].

The optimal P@1 was found for:
Number of documents N = 3,
Interpolation parameter λ = 0.3,
Number of NN k= 128,
Distance scaling l = 6.

D kNN without IR

To enable a kNN search over the full datastore we use FAISS index (Johnson et al., 2017). We train the index using 1M randomly sampled keys and 40960 number of clusters. Embeddings are quantized to 64 bytes. During inference the index looks up 64 clusters.

E Computational Infrastructure

The creation of the datastore is computationally expensive but only a single forward pass is needed. The datastore creation is run on a server with 128 GB memory, Intel(R) Xeon(R) CPU E5-2630 v4, CPU rate 2.2GHz, number of cores 40(20), 8x GeForce GTX 1080Ti. One GPU embedds 300 contexts/s. The datastore includes 900M contexts.

Evaluation is run on a server with 128 GB memory, Intel(R) Xeon(R) CPU E5-2630 v4, CPU rate 2.2GHz, number of cores 40(20). Evaluation time for one query is 2 s but code can be optimized for better performance.

Chapter 9

EDIN: An End-to-end Benchmark and Pipeline for Unknown Entity Discovery and Indexing

EDIN: An End-to-end Benchmark and Pipeline for Unknown Entity Discovery and Indexing

Nora Kassner, Fabio Petroni, Mikhail Plekhanov, Sebastian Riedel, Nicola Cancedda
Meta AI
kassner@meta.com

Abstract

Existing work on Entity Linking mostly assumes that the reference knowledge base is complete, and therefore all mentions can be linked. In practice this is hardly ever the case, as knowledge bases are incomplete and because novel concepts arise constantly. We introduce the temporally segmented *Unknown Entity Discovery and Indexing* (EDIN) *-benchmark* where unknown entities, that is entities not part of the knowledge base and without descriptions and labeled mentions, have to be integrated into an existing entity linking system. By contrasting EDIN with zero-shot entity linking, we provide insight on the additional challenges it poses. Building on dense-retrieval based entity linking, we introduce the end-to-end EDIN-*pipeline* that detects, clusters, and indexes mentions of unknown entities in context. Experiments show that indexing a single embedding per entity unifying the information of multiple mentions works better than indexing mentions independently.

1 Introduction

Most existing works on Entity linking (EL) – the fundamental task of detecting mentions of entities in context and disambiguating them against a reference knowledge base (KB) – assume that such KB is complete, and therefore all mentions can be linked. In practice this is hardly ever the case, as KBs are incomplete when they are created and because novel concepts arise constantly. For example, English Wikipedia, often used as the reference KB for large scale linking, is growing by more than 17k entities every month.[1]

Consequently, at the time of deployment EL systems are quickly outdated and static evaluation overestimates performance. But as these systems play significant role in many real world industry applications, e.g., moderating discussions around recent events, a dynamic look on EL is crucial.

Nonetheless, related work on this problem is sparse. Available datasets (Ji et al., 2015; Derczynski et al., 2017; Nakashole et al., 2013) and models (Hoffart et al., 2014) are outdated and/or small scale and use features which are not readily available (Nakashole et al., 2013; Wu et al., 2016). Most importantly, they approach the problem only in parts. We revisit this problem in context of dense-retrieval and large-scale EL, e.g., EL relying on bi-encoder architecture that runs a nearest neighbor search between mention encoding and a large-scale index of entity encodings. To this end, we introduce EDIN-*benchmark* and EDIN-*pipeline* where *unknown entities*, that is entities with no available canonical names, descriptions and labeled mentions, have to be integrated into an existing EL model in an end-to-end fashion. To the best of our knowledge, EDIN-*pipeline* is the first end-to-end pipeline tackling this problem.

Note that this setting is strictly more demanding than zero-shot (zs) entity linking (Logeswaran et al., 2019), where a textual description of the zs entities is available at the time of training.

The EDIN-*benchmark* is temporally segmented into two parts, one preceding time t_1 and one preceding time t_2. With current approaches, an EL system created at t_1 is unable to create a dense-index entry – and therefore successfully link – unknown entities introduced after t_1. The task that we propose consists in adapting a model trained at t_1 using only an *adaptation dataset* – a set of new documents also mentioning unknown entities – and unsupervised techniques. There are therefore two parts to this task: i) Discovery, which consists in detecting mentions of unknown entities in the adaptation dataset and classifying them as unknown and ii) Indexing, consisting in mapping co-referring mentions of unknown entities to a single representation compatible with the entity index.

[1] https://en.wikipedia.org/wiki/Wikipedia:Size_of_Wikipedia (09.05.2022)

By introducing a clear-cut temporal segmentation EDIN-*benchmark* targets unknown entities which are truly novel/unseen to all parts of an EL system, specifically the pre-trained language model (PLM). Therefore, the EL system cannot rely on implicit knowledge captured by the PLM. This is, to the best of our knowledge, a setting that has not been explored before in the context of dense-retrieval based EL.

Temporal segmentation also lets us study effects of entity encoder and PLM degradation. We observe that precision drops for known entities in novel contexts which points to a large problem of PLM staleness also discussed by (Agarwal and Nenkova, 2021; Dhingra et al., 2022; Lazaridou et al., 2021).

We show that distinguishing known from unknown entities, arguably a key feature of an intelligent system, poses a major challenge to dense-retrieval based EL systems, as a model has to strike a delicate balance between relying on mention vs. context: context is crucial to distinguish unknown entities carrying the same name as known entities and to co-refer different mentions of the same unknown entities, while mentions are essential to distinguish unknown entities with different name but semantic similarity to existing ones.

On the side of indexing, inserting unknown entities into a space of known entities poses problems of interference with known entities in their close proximity. For instance, when first encountering mentions of BioNTech we want to create an index entry in proximity of other biotech companies but in a way that linking can still differentiate between them. We find that adapting the EL model to the updated index, is essential.

We experiment with different indexing methods. In particular, we contrast single mention-level indexing (FitzGerald et al., 2021) with indexing clusters of mentions. We find that unifying the information of multiple mentions into a single embedding is beneficial.

We summarize our contributions as follows: i) We introduce the EDIN-*benchmark*, a large scale end-to-end EL dataset where unknown entities need to be discovered and integrated into an existing entity index in an unsupervised fashion. ii) We propose the EDIN-*pipeline* in the form of an extension of existing dense-retrieval architectures. iii) We contrast this task with zs EL, and provide insight on the challenges it poses. iv) We show that indexing a single embedding per entity, unifying the information of multiple mentions, works better than indexing mentions independently.

Data and evaluation code is located here: `https://github.com/facebookresearch/EDIN`

2 Task definition

We formally define end-to-end EL as follows: Given a paragraph p and a set of known entities $E_K = \{e_i\}$, each with canonical name, the title, $t(e_i)$ and textual description $d(e_i)$, our goal is to output a list of tuples, $(e, [i, j])$, where $e \in E_K$ is the entity corresponding to the mention $m_{i,j}$ spanning from the i^{th} to j^{th} token in p. We call a system that solves this task based on $d(e_i)$ a **Description-based** entity linking system L.

For EDIN-*benchmark*, after training a model L_{t1} at time step t_1, a set of unknown entities $E_U = \{e_i\}$ with $E_U \bigcap E_K = \emptyset$ and no available canonical names, descriptions and labeled mentions is introduced between t_1 and $t_2 > t_1$. The task is to adapt L_{t1} in an unsupervised fashion such that it can successfully link mentions of $E_U \bigcup E_K$.

We use three dataset splits: the training set D_{train} to train L_{t1}, the adaptation dataset D_{adapt} used to adapt L_{t1} and the test set D_{test} to evaluate. Both D_{adapt} and D_{test} include mentions between t_1 and t_2. The model relies on D_{adapt} to discover E_U and extract representations to integrate E_U into the entity index. We ensure that D_{adapt} and D_{test} are disjoint to prevent leakage of test mentions into entity representations extracted from D_{adapt}.

3 EDIN-*pipeline*

Our EDIN-*pipeline* is built on top an end-to-end extension of the dense-retrieval based model BLINK (Ledell Wu, 2020) and is similar to (Li et al., 2020). It is composed of a Mention Detection (MD), Entity Disambiguation (ED) and Rejection (R) components. MD detects entity mention spans $[i, j]$ in context relying on BERT (Devlin et al., 2019). ED links these mentions to $e \in E_K$. It relies on bi-encoder architecture running a k-nearest-neighbor (kNN) search between *mention encoding* and candidate *entity encodings* (the entity index). Mention encodings are pooled from BERT-encoded paragraph tokens $p_{1..n}$:

$$\mathbf{m}_{i,j} = FFL(BERT([CLS]p_1 \dots p_n[SEP])_{i \dots j})$$

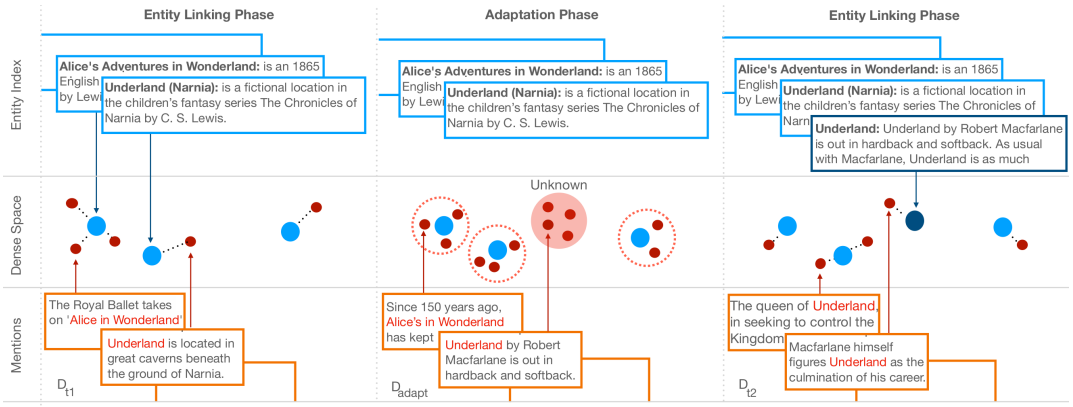


Figure 1: **EDIN-*pipeline*:** In the adaptation phase, detected mentions in D_{adapt} are mapped into a joint dense space with E_K representations. A clustering algorithm groups mentions and entities based on kNN-similarity. Clusters of mentions without entity encoding are collected in E'_U. To integrate these into the index of E_K, mentions in single-sentence contexts are concatenated and mapped to a single embedding using the entity encoder. After adaptation, the updated entity index is used for standard EL in an inductive setting.

Entities are represented using BLINK's *frozen* entity encoder:

$$\mathbf{e} = BERT_{[CLS]}([CLS]t(e)[SEP]d(e)[SEP])$$

Mention-entity candidates are passed to R that controls precision-recall trade-off by thresholding a learned candidate score.

More information about architecture and training are detailed in appendix A.

4 Unknown Entity Discovery and Indexing

We introduce an end-to-end pipeline to encode E_U into L_{t1}'s entity index. The process is depicted in Figure 1. This pipeline is fully unsupervised and only relies on D_{adapt}. It follows a two-step process: i) in Discovery the EL system detects mentions of unknown entities and recognises them as being unknown; ii) during Indexing, co-referring mentions of unknown entities are mapped to a single embedding compatible with the entity index. After adaptation the updated model is tested on D_{test}.

4.1 Unknown Entity Discovery

First, L_{t1} detects and encodes mentions part of D_{adapt}. The MD head is trained to detect mentions leveraging the context around them, and can therefore detect mentions of both E_K and E_U. Encoded mentions $\mathbf{M} = \{\mathbf{m}_1, ..., \mathbf{m}_{|M|}\}$ are then input to a clustering algorithm that partitions M into disjoint clusters $C = \{c_1, ..., c_{|C|}\}$. We adopt the same greedy NN clustering algorithm as Logan IV et al. (2021) where $\mathbf{m}_i$ is assigned to cluster c_k if $\mathbf{m}_j \in c_k$ is NN mention to $\mathbf{m}_i$ and $sim(\mathbf{m}_i, \mathbf{m}_j) > \delta$.

Next, entity encodings of $e \in E_K$ are assigned to these clusters if $\sum_{j=0..J}(sim(\mathbf{e}_i, \mathbf{m}_j)))/J > \tau$ holds for $\mathbf{m}_j \in c_i$ with $\mathbf{e}_i$ being the nearest entity of $\mathbf{m}_j \in c_i$. δ and τ are tuned on D_{adapt}-dev to optimize for recall. For more details see appendix C. Following Agarwal et al. (2021), all clusters not containing any entity representation are deemed to refer to entities in E_U. We refer to this subset of automatically identified unknown entities as E'_U.

4.2 Unknown Entity Indexing

Next, clusters identified as E'_U are integrated into the EL index of L_{t1}. We explore two different methods of indexing:

Cluster-based: We concatenate all mentions part of a cluster, each with the sentence they occur in, and use the entity encoder to map to a single entity encodings. We pool over all $m_i \in c_i$ and select the most occurring mention as canonical name $t(e)$.

Mention-based: Mentions in single sentence contexts are indexed individually using the entity encoder. Individual mentions are used as $t(e)$.

5 Evaluation

As mentions of type E_U are significantly less frequent than mentions of type E_K, we report results on these two types separately.

For *Discovery*, we report precision and recall of E_U classification and clustering metrics.

	Wikipedia	OSCAR
Train	100k (908k)	100k (1.7M)
Adapt	17k (183k)	17k (380k)
Dev Train	8k (78k)	8k (142k)
Dev Adapt	-	9k (183k)
Test	198k (1.8M)	569k (11M)

Table 1: **Dataset Statistics:** Number of samples (number of mentions) for training, adaptation and testing.

Bin	Support	E_K R	Support	E_U R
[0)	68,241	21.1	7,095	17.5
[1)	59,227	29.1	3,923	25.9
[1, 10)	313,232	45.6	9,939	40.7
[10, 100)	901,857	65.7	7,765	57.3
[100, 1k)	2,860,880	76.9	7,399	64.4
[1k, +)	5,981,028	84.4	6,717	86.7

Table 2: **Frequency effects:** End-to-end EL performance of upper baseline model L_{t2} per frequency bins.

To evaluate end-to-end EL, we compute precision (P) and recall (R) following Li et al. (2020) but using a hard matching criteria.

To do so for cluster-based discovery, canonical names of indexed clusters need to be consistent with the set of test labels. Our method of assigning canonical names to clusters based on pooling over mentions is not. To resolve this mismatch we pool over the gold labels associated with these mentions instead of the mentions themselves. This is only done for evaluation.

Unsupervised clustering of mentions in D_{adapt} may suffer from two kinds of errors: i) Clusters can be incomplete, e.g., mentions of a single entity can be split into multiple clusters which can lead to indexing the same entity multiple times and ii) Clusters can be impure, e.g., mentions of different entities end in the same cluster, which leads to conflation of multiple entities into one representation.

In our evaluation we use the gold labels for computing standard EL metrics by associating possibly more than one cluster to each E_U, and consider a prediction correct if a mention is linked to any of the clusters associated with the correct entity. EL metrics could fail to capture shortcomings in establishing co-references between mentions though, therefore we report clustering metrics alongside EL metrics. We follow Agarwal et al. (2021) and report normalized mutual information (NMI).

6 EDIN-*benchmark*

To construct the entity index, we download Wikipedia dumps from $t1$ and $t2$ and extract entity titles and descriptions. Setting t_1 to September 2019 (the date when BLINK was trained) the KB consists of 5.9M entities, setting t_2 to March 2022 an additional set of 0.7M entities is introduced.

Wikipedia and Oscar data is created as follows.

Wikipedia: Since usually only the first mention of an entity inside a Wikipedia article is hyperlinked, we annotate a subset of Wikipedia. We use a version of L that was trained at $t2$ on a labelled non-public dataset. While noisy, these predictions are significantly better than what our best discovery and indexing methods can achieve, therefore we adopt them as pseudo-labels for the purpose of comparing approaches. As discovery and indexing methods improve, manual labelling of the evaluation data will afford more accurate measures. Wikipedia provides time stamps which enables us to separate two time splits.

OSCAR news: This dataset is based on the common-crawl dataset OSCAR (Abadji et al., 2021). We select a subset of English language news pages which we label automatically as described above. The dataset consists of 797k samples, which we split based on their publication date. We publish this dataset using stand-off annotations and code to download the relevant raw data. To enable evaluation of future versions of PLMs and EL systems, we also publish our data processing scripts.

For both types of datasets we publish two time splits: D_1, containing samples preceding $t1$, which is used to train model L_{t1} and D_2, with samples preceding $t2$, which is used to train an upper bound model L_{t2}. To adapt L_{t1}, we hold out a subset of data from between $t1$ and $t2$ to construct D_{adapt} ($D_{adapt} \cap D_2 = \emptyset$). Remaining samples are randomly split into train, dev, test. Figure 2 illustrates the different data splits. Overall dataset statistics are listed in Table 1.

To construct D_{adapt}, we follow Agarwal et al. (2021), and set the ratio of mentions of type E_U to E_K to 0.1. [2] As D_{t2}-test covers both known and unknown entities, we use this dataset for EDIN-*pipeline* evaluation. In Oscar D_{t2}-test, the average number of mentions per E_U is 5.6 and it is ten

[2]Naturally this ratio would lie at 0.02. We made this artificial adjustment to reduce the strong class imbalance and obtain more interpretable and statistically stable results. Such adjustment could be lifted once considerably more precise unknown entity discovery components become available.

times lower than for E_K. COVID-19 is the most occurring unknown entity with 12k mentions. 638k E_U are not mentioned at all and only 733 are mentioned more than ten times.

7 Results and Discussion

In the following sections, we discuss results for OSCAR data. Results on Wikipedia data are consistent but lower and shown in appendix F. Our main findings are shown in Table 3 where we report end-to-end performance on OSCAR D_{t2}-test.

Overall, our results show:

- EDIN-*benchmark* is challenging. Particularly attributed to imperfect discovery, end-to-end performance in terms of recall lacks significantly behind the upper bound, see 7.5.
- When contrasting with zs EL, we find that i) adapting the model to the updated index by re-training the model after indexing is crucial, see 7.2 and ii) entity encodings relying on clusters of mentions in context instead of human crafted descriptions have high potential but discovering these clusters is challenging, see 7.4.1.
- Our best performing system relies on Cluster-based indexing, with the advantage of attending to and unifying the information of multiple mentions, see 7.4. We call this version the EDIN-*pipeline*.

In whats to come, we first discuss upper and lower performance bounds. Then, we follow our two-step pipeline where we first present results on discovery and indexing separately and then assemble the full end-to-end pipeline.

Recall our terminology:

- **Cluster-based:** E_U encodings rely on mentions in context which are concatenated and embedded into a single encoding.
- **Mention-based:** E_U encodings rely on individually indexed mentions in context.
- **Description-based:** E_U encodings rely on human crafted descriptions. This type of indexing is used in the zs setting.

7.1 Lower and upper bounds

Our starting point, and an obvious lower performance bound, is given by model L_{t1} trained at D_{t1}. This model lacks representations of E_U and its training data does not contain any corresponding mentions. Therefore, performance on the subset of E_U is 0 for all metrics.

For an upper performance bound we take model L_{t2} trained at D_{t2}. The entities in E_U were introduced to Wikipedia past t_1 but before t_2, meaning that to L_{t2} these entities are actually *known*: labeled mentions of E_U are part of the training data and entity representations are part of the index.

L_{t2} reaches similar performance as L_{t1} for E_K. We suspect performance differences can be attributed to the difference in training data.

Performance of L_{t2} on mentions of E_U is lower than on mentions of E_K. The performance discrepancy between E_U and E_K is largely due to frequency differences, see Table 2. We suspect that the remaining difference can be attributed to the degradation of PLM and entity encoder. Note that while labelled mentions of E_U were seen during the training phase of L_{t2}, BLINK's entity encoder was not re-trained. To investigate this hypothesis further, we test L_{t1} on mentions of E_K that meet two conditions: i) time stamps of these samples are posterior to $t1$ and ii) two or more mentions of E_U occur in their context. Thus, we target mentions of E_K in novel contexts to which neither BLINK nor the PLM have been exposed. The total number of entities that meet these conditions are 40,055. We find that recall drops only slightly from 80.1 to 79.9 but precision drops from 82.0 to 75.9. This result indicates that E_U are also a source of noise when trying to link mentions of E_K.

7.2 Additional upper bound: Zero-shot EL

Zs EL relies on Description-based indexing. It may be a valid option in some practical settings, where we may e.g. be able to frequently download fresh Wikipedia snapshots and rerun all or part of the training, but it does not meet the conditions for being a valid entry to EDIN-*benchmark*, because it relies on supervision for deciding what novel entities to add to the index, and because it requires manually written descriptions for such entities. For these reasons, we present it here as an additional upper bound comparison point.

We note that in the zs problem, all entities are part of the index at training time. In the setting

	Known Entities			Unknown Entities			Unknown Entities filtered		
Model	**R**	**P**	**NMI**	**R**	**P**	**NMI**	**R**	**P**	**NMI**
L_{t1} (lower bound)	80.1	82.0	93.5	0.0	0.0	0.0	0.0	0.0	0.0
L_{t2} (upper bound)	78.7	79.7	93.1	49.2	31.8	93.8	63.1	26.0	93.4
L_{t1}-Descp (zs setting)	80.2	82.6	93.5	46.5	32.4	93.8	58.3	26.2	90.5
L_{t1}-Mention-Oracle	80.6	81.5	93.3	24.0	46.6	87.0	40.7	46.6	87.0
L_{t1}-Mention	80.3	81.9	93.4	20.5	43.7	87.6	34.5	43.5	88.7
L_{t1}-Cluster-Oracle	80.3	82.0	94.2	30.5	51.8	85.9	51.8	51.8	85.9
EDIN-*pipeline* (L_{t1}-Cluster)	80.3	81.9	93.4	20.8	43.1	85.9	35.4	43.1	85.3

Table 3: **EL performance** on OSCAR D_{t2}-test for unknown entities E_U and known entities E_K. **Left** shows end-to-end performance; **Right** shows filtered performance where mentions of E_U not part of D_{adapt} are dropped from test. **Upper/Lower bounds:** L_{t1}, trained at $t1$, uses Description-based entity encodings and constitutes the lower bound. It lacks encodings of E_U. L_{t2}, trained at $t2$, uses Description-based entity encodings and constitutes the upper bound. E_U are part of the index and their labeled mentions are part of training. L_{t1}-Descp adapts L_{t1} by adding Description-based entity encodings of E_U to the index. As it relies on human discovery and descriptions it constitutes an additional upper bound. **Adaptation:** For L_{t1}-Mention Mention-based encodings of i) oracle E_U and ii) discovered E'_U part of D_{adapt} are added to L_{t1}'s entity index. For L_{t1}-Cluster Cluster-based encodings of i) oracle E_U and discovered E'_U part of D_{adapt} are added to L_{t1}'s entity index.

of EDIN-*benchmark*, indexing happens after training. We run the following experiments to study the effect this difference has:

- **Not Re-trained:** Description-based entity representations are added to the index *without* re-training L_{t1} after indexing.
- **Re-trained:** Description-based entity representations are added to the index *with* re-training L_{t1} after indexing.

Recall and precision of E_U with *Re-trained* is 46.5% and 32.4% respectively, see Table 4. Recall and precision with *Not Re-trained* is 26% and 17% points lower respectively.

We note that unknown entities can potentially be placed in close proximity to known ones in embedding space. When these entity encodings are present during training, they can be picked up as hard negatives and the mention encoder can learn to circumvent them. This hypothesis is supported by experiments showing that the mean similarity between mentions and correct known entity embeddings increases significantly when the mention encoder is re-trained after adding the new entities. For details see the appendix D.

The take-away for the EDIN-*pipeline* is that, after adding new entity representations to the index, another round of training is needed to adapt the mention encoder to the updated index. We adopt this approach for the following experiments. Besides adapting the mention encoder, re-training BLINK could have a similar effect: in such case learning from hard negative can affect the spacing of entity encodings. As re-training BLINK is expensive, we did not explore this option.

	Unknown Entities			Known Entities		
	R	**P**	**NMI**	**R**	**P**	**NMI**
Not re-trained	20.6	15.5	95.2	80.1	82.3	93.5
Re-trained	46.5	32.4	93.8	80.2	82.6	93.5

Table 4: **Adapting the model to the updated index:** End-to-end EL performance on OSCAR D_{t2}-test when adding Description-based representation of unknown entities E_U to the entity index with (Re-trained) and without (Not re-trained) re-training of L_{t1}.

7.3 EDIN Discovery

First condition for effective discovery is the ability to reliably detect mentions of both E_K and E_U. Recall of L_{t1} on D_{adapt} for MD task is 90% for E_K and 86% for E_U. As expected, recall of mentions of E_K is higher as no mentions of E_U were seen during training. As a reference, running L_{t2} on D_{adapt} we find that for both E_K and E_U 91% of mentions are recalled. Note again, that for L_{t2}, E_U are *known*. This indicates that MD is not affected by frequency differences and PLM degradation.

Once mentions are detected, we adopt a clustering approach to classify between mentions of E_U

and E_K. We measure clustering quality of 91.2% NMI on D_{adapt}. We evaluate discovery based on these clusters by evaluating whether a discovered cluster is indeed referring to an E_U. Note, that here duplicated discovery of the same entity is not penalized. We set the minimum number of mentions per cluster to 3 and report low discovery precision (10%) but relatively high recall (86%). Overall, this results in detecting 71% of all unknown entities part of D_{adapt}.

We find that the constraint requiring that most mentions in a cluster are within a region controlled by hyper-parameter τ, as described in 4.1, is crucial. In an ablation study we drop this condition and greedily assign E_K to clusters if $sim(\mathbf{e}_i, \mathbf{m}_i) > \tau$ holds for *any* $m_i \in c_i$. This setting is similar to Agarwal et al. (2021) where a single entity-mention link is sufficient for cluster assignment. Discovery dropped to 49% recall and 8% precision.

A qualitative error analysis reveals that false negatives are mostly caused by the problem that mention embeddings of E_U (e.g. BioNTech) can have high similarity with entity embeddings of E_K (e.g. of other biotechnology companies). We suspect that this problem is particularly pronounced in our setting because EDIN-*benchmark* is large scale (up to 6 times more entities in the reference KB and up to 36 times more mentions in the clustering set compared to Agarwal et al. (2021)) with many tail entities.

Conversely, false positives are mostly due to known entities being misclassified unknown when occurring in novel contexts, e.g., "blood tests" or "vaccine" in context of COVID form distinct clusters. But, low precision in discovery is less problematic than low recall as re-training after indexing gives the ability to learn to ignore clusters of E_K.

7.4 EDIN indexing

After discovery, we need mention clusters of E_U to be integrated into the entity index.

We compare Mention-based and Cluster-based indexing. To isolate discovery and indexing performance, we first evaluate indexing using oracle clusters, where we replace the discovery method run on D_{adapt} with an oracle where mentions of E_U are discovered and clustered perfectly. Mention-based indexing performs worse than Cluster-based indexing with a gap of around 5% points, see Table 3 (left), L_{t1}-Mention-Oracle vs. L_{t1}-Cluster-Oracle. When reducing the test set to mentions of entities that were actually discoverable, the difference in recall becomes even more pronounced: 41% for Mention-based vs. 52% for Cluster-based indexing, see Table 3 (right).

Interestingly, this means that the ability to attend over multiple mentions in context and unify their information into a single embedding leads to superior representations. Note that here the entity encoder was neither trained to deal with the style of individual mentions in context nor with clusters of mentions in context. For future work, it would be interesting to see if Cluster-based indexing can be generally beneficial to EL, outside of the context of EDIN-*pipeline*.

7.4.1 Cluster-based vs. Description-based

As an upper baseline, we compare Cluster-based indexing with the zs setting which uses Description-based indexing. Zs EL does not rely on D_{adapt} but on a human's decision to add an entry to the index and therefore discovery is perfect. To isolate indexing from discovery, we again filter the test set to actually discoverable entities and assume perfect oracle clusters.

In this setting, see Table 3 (right), we find that Cluster-based-Oracle indexing performs 7% points lower than Description-based indexing in recall but 26% better in terms of precision.

The take-away is that when discovery is perfect, Cluster-based indexing relying on concatenated mentions in context instead of manually crafted descriptions has high potential. In the end-to-end setting, we see that assembling these perfect clusters is challenging.

We also want to emphasise that results in Table 3 show that EL performance on E_K is not affected by this adaptation process. Recall and precision remain, with 80.3 and 81.9, stable. We also test if this finding also holds on standard EL datasets. We compare performance on AIDA test before and after adaptation and report no difference in performance on E_K.

7.5 End-to-end pipeline

We assemble the full end-to-end pipeline. We replace oracle clusters of E_U by discovered clusters of E'_U. Errors in discovery that affect indexing are: i) misclassification of clusters as either known or unknown and ii) incomplete and impure clusters. We find that performance of Mention-based and Cluster-based indexing in terms of recall and precision converges and is significantly lower than their

oracle counterparts.

When reducing the test set to mentions of entities that were discoverable, thus part of D_{adapt}, Cluster-based indexing is 1% point better in terms of recall and 0.4 % worse in precision, Table 3 (right). When reducing the test set further to mentions of entities that were in fact discovered, recall of Cluster-based indexing is, with 58.4%, better than that of Mention-based indexing (55.5%).

We also report performance of ED with oracle mention detection in Table 6 in the appendix E. Here, we find that Cluster-based indexing is performing better than Mention-based indexing across all metrics.

We conclude that Cluster-based indexing performs better than Mention-based indexing. We call this version the EDIN-*pipeline*.

Besides yielding an index that scales in memory with the number of entities rather than the number of mentions – a significant advantage when the number of entities is already large and in view of a streaming extension – Cluster-based indexing generates fixed-size entity embeddings as a by-product that can have applications of their own and can be used to enhance PLMs (e.g., Peters et al. (2019)).

Overall, EDIN-*pipeline* performance shows that EDIN-*benchmark* is challenging. In terms of recall, end-to-end performance lacks 26% points behind the upper bound L_{t2}. In this setting, errors in discovery propagate. Most notably, we see this manifest when i) comparing Table 3 unfiltered and filtered where the recall problem of E_U becomes apparent and ii) comparing performance of oracle and automatic clusters where precision drops by 10% points.

In future work, we want to explore a setting where E_U are discovered in a streaming fashion, thus scaling up D_{adapt} and dropping the artificially imposed ratio of E_K vs. E_U. This would pose challenges in terms of scale and precision in discovery. Here, a human in the loop approach, as proposed by Hoffart et al. (2016) in the context of keeping KBs fresh, to introduce a component of supervision, might be needed.

8 Related work

EL is an extensively studied task. Prior to the introduction of PLMs, EL systems used frequency and typing information, alias tables, TF-IDF-based methods and neural networks to model context, mention and entity (Cucerzan, 2007; Bunescu and Paşca, 2006; Milne and Witten, 2008; He et al., 2013; Sun et al., 2015a; Lazic et al., 2015; Raiman and Raiman, 2018; Kolitsas et al., 2018; Gupta et al., 2017; Ganea and Hofmann, 2017; Khalife and Vazirgiannis, 2018; Onoe and Durrett, 2019).

Gillick et al. (2019) present a PLM-based dual encoder architecture that encodes mentions and entities in the same dense vector space and performs EL via kNN search. Logeswaran et al. (2019) proposed the zs EL task and show that domain adaptive training can address the domain shift problem. Subsequently, Wu et al. (2020) showed that pre-trained zs architectures are both highly accurate and computationally efficient at scale. None of these works tackle the problem of unknown entities.

Recently, FitzGerald et al. (2021) model EL entirely as mappings between mentions, where inference involves a NN search against all known mentions of all entities in the training set. In this setting mentions need to be labeled. They do not explore their approach in the setting of unknown entities.

Prior to dense retrieval-based EL, unknown entity discovery work includes: Ratinov et al. (2011) train a classifier to determine whether the top ranked EL candidate is unknown relying on local context, global Wikipedia coherence, and additional manually crafted features. Nakashole et al. (2013) introduce a model for unknown entity discovery and typing leveraging incompatibilities and correlations among entity types. Hoffart et al. (2014); Wu et al. (2016) study a variety of features for unknown entity discovery: Hoffart et al. (2014) use perturbation-based confidence measures and key-phrase representations and Wu et al. (2016) explore different feature spaces, e.g., topical and search engine features. These features are not readily available and incorporating them into PLM-based approaches is not straightforward; Ji et al. (2015); Derczynski et al. (2017) introduce shared tasks for discovery. These tasks are defined on comparatively small datasets and target only named entities; Akasaki et al. (2019) introduces a time sensitive method of discovering emerging entities relying on Twitter data.

None of these works consider unknown entities in an end-to-end setting including mention detection, unknown entity discovery and indexing. Also, we cannot use their datasets to evaluate as these entities were part of training the PLM.

In the context of named entity tagging, Mota and

Grishman (2009) showed that entity taggers can be effectively updated by incorporating contemporary unlabeled data using semi-supervised learning.

Closely related to EL is the task of cross document entity co-reference (CDC), where no reference KB is present (Bagga and Baldwin, 1998; Gooi and Allan, 2004; Singh et al., 2011; Dutta and Weikum, 2015; Barhom et al., 2019; Cattan et al., 2021a; Caciularu et al., 2021; Cattan et al., 2021b). Most recently, Logan IV et al. (2021) benchmark methods for streaming CDC, where mentions are disambiguated in a scalable manner via incremental clustering. Our work can be seen as bridging between the world of CDC and EL.

Most recently, Angell et al. (2021) introduce a new EL method using document-level supervised graph-based clustering. Agarwal et al. (2021) extend this work to cross-document EL and entity discovery. In this work, we adopt a more standard bi-encoder architecture (i.e. BLINK), with better EL scalability potential (memory linear in the number of entities and not in the number of mentions) and an existing end-to-end extension. We use a modified version of their discovery method.

9 Conclusion

This work introduced EDIN-*benchmark* and EDIN-*pipeline*. EDIN-*benchmark* is a large-scale, end-to-end EL benchmark with a clear cut temporal segmentation for *Unknown Entity Discovery and Indexing*. EDIN-*pipeline* detects and clusters mentions of unknown entities in context. These clusters of unknown mentions are then collapsed into single embeddings and integrated into the entity index of the original EL system.

Limitations

The main limitations of EDIN-*benchmark* are: i) The dataset is not human-annotated. Instead we used an upper-bound model to label data automatically. ii) We limit D_{adapt} in size and artificially adjust class imbalance between mentions of type E_U to E_K. The limited size of D_{adapt} in turn limits the discoverability of unknown entities, specifically low-frequency ones. Once progress is made in the accuracy and scalability of entity discovery, EDIN-*benchmark* can be modified to a truly dynamic setting where unknown entities are continuously discovered in a stream of incoming documents and integrated into the EL system.

EDIN-*pipeline* is tailored to dense-retrieval based EL and adapting it to different EL approaches, e.g., to generative EL systems De Cao et al. (2021), is not straightforward.

We study EDIN-*benchmark* and -*pipeline* in a monolingual setting using English language only. EDIN-*benchmark*'s extension to a multilingual setting is straight forward. OSCAR and Wikipedia data are available in 166 different languages but coverage will be a problem. EDIN-*pipeline* can be extended to more languages by following (Botha et al., 2020) but EDIN performance is expected to vary across languages as it does for standard EL.

EDIN-*benchmark* covers news and Wikipedia domain entities only, and we have not evaluated the EDIN-*pipeline* on other domains.

The overall performance of EDIN-*pipeline* has ample margins for improvement, with the precision of clustering-based discovery as the main bottleneck at present. The significant number of false positives (mentions of known entities classified as unknown) is still a barrier to deployment in most real-world settings.

Ethical Considerations

EL is a standard NLP task. Outside of academia EL can be deployed in both non-problematic (e.g., content understanding for hate speech detection) and problematic (e.g., surveillance) settings. Independent of the use-case, potential bias that these models could exhibit needs to be evaluated. EL relies on human curated knowledge bases (here Wikipedia) which could carry bias e.g. in terms of language, genders and races, see for example Sun and Peng (2021). Another source of bias in the context of dense-retrieval based EL, is the bias of the underlying language model (here BERT). Both potential sources of bias could be propagated to the down-stream task. To mitigate biases, we refer to Goldfarb-Tarrant et al. (2021); Steed et al. (2022) that show bias mitigation needs to be done on the side of the downstream task rather than the language model. Rudinger et al. (2018); Zhao et al. (2018) introduce methods of downstream bias mitigation, here in the context of co-reference resolution.

We publish our dataset/scripts that generate the datasets. Our dataset is based on English Wikipedia and a subset of English online news pages extracted from OSCAR. All Wikipedia based data is made fully available. OSCAR is common-crawl based data and only available to researchers upon

request. We release code and stand-off annotations which enables researchers to reproduce the dataset. Our EL annotations rely on an upper bound model which is due to the performance gap sufficient for EDIN but should not be considered gold data for general EL tasks. We will indicate this prominently on the website we use to host the data.

Acknowledgements

We thank our colleagues Louis Martin and Frederic Dreyer for their valuable feedback.

References

Julien Abadji, Pedro Javier Ortiz Suárez, Laurent Romary, and Benoît Sagot. 2021. Ungoliant: An optimized pipeline for the generation of a very large-scale multilingual web corpus. Proceedings of the Workshop on Challenges in the Management of Large Corpora (CMLC-9) 2021. Limerick, 12 July 2021 (Online-Event), pages 1 – 9, Mannheim. Leibniz-Institut für Deutsche Sprache.

Dhruv Agarwal, Rico Angell, Nicholas Monath, and Andrew McCallum. 2021. Entity linking and discovery via arborescence-based supervised clustering.

Oshin Agarwal and Ani Nenkova. 2021. Temporal effects on pre-trained models for language processing tasks.

Satoshi Akasaki, Naoki Yoshinaga, and Masashi Toyoda. 2019. Early discovery of emerging entities in microblogs. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI 2019, Macao, China, August 10-16, 2019*, pages 4882–4889. ijcai.org.

Rico Angell, Nicholas Monath, Sunil Mohan, Nishant Yadav, and Andrew McCallum. 2021. Clustering-based inference for biomedical entity linking. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2598–2608, Online. Association for Computational Linguistics.

Amit Bagga and Breck Baldwin. 1998. Entity-based cross-document coreferencing using the vector space model. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1*, ACL '98/COLING '98, page 79–85, USA. Association for Computational Linguistics.

Shany Barhom, Vered Shwartz, Alon Eirew, Michael Bugert, Nils Reimers, and Ido Dagan. 2019. Revisiting joint modeling of cross-document entity and event coreference resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4179–4189, Florence, Italy. Association for Computational Linguistics.

Jan A. Botha, Zifei Shan, and Daniel Gillick. 2020. Entity Linking in 100 Languages. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7833–7845, Online. Association for Computational Linguistics.

Razvan Bunescu and Marius Paşca. 2006. Using encyclopedic knowledge for named entity disambiguation. In *11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 9–16, Trento, Italy. Association for Computational Linguistics.

Avi Caciularu, Arman Cohan, Iz Beltagy, Matthew Peters, Arie Cattan, and Ido Dagan. 2021. CDLM: Cross-document language modeling. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2648–2662, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yixin Cao, Lei Hou, Juan-Zi Li, and Zhiyuan Liu. 2018. Neural collective entity linking. In *COLING*.

Arie Cattan, Alon Eirew, Gabriel Stanovsky, Mandar Joshi, and Ido Dagan. 2021a. Realistic evaluation principles for cross-document coreference resolution. In *Proceedings of *SEM 2021: The Tenth Joint Conference on Lexical and Computational Semantics*, pages 143–151, Online. Association for Computational Linguistics.

Arie Cattan, Sophie Johnson, Daniel S. Weld, Ido Dagan, Iz Beltagy, Doug Downey, and Tom Hope. 2021b. Scico: Hierarchical cross-document coreference for scientific concepts. In *3rd Conference on Automated Knowledge Base Construction*.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 708–716, Prague, Czech Republic. Association for Computational Linguistics.

Nicola De Cao, Gautier Izacard, Sebastian Riedel, and Fabio Petroni. 2021. Autoregressive entity retrieval. In *International Conference on Learning Representations*.

Leon Derczynski, Eric Nichols, Marieke van Erp, and Nut Limsopatham. 2017. Results of the WNUT2017 shared task on novel and emerging entity recognition. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 140–147, Copenhagen, Denmark. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Jeremy Cole, Julian Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William Cohen. 2022. Time-aware language models as temporal knowledge bases. *Transactions of the Association for Computational Linguistics*, 10(0):257–273.

Sourav Dutta and Gerhard Weikum. 2015. Cross-document co-reference resolution using sample-based clustering with knowledge enrichment. *Transactions of the Association for Computational Linguistics*, 3:15–28.

Nicholas FitzGerald, Dan Bikel, Jan Botha, Daniel Gillick, Tom Kwiatkowski, and Andrew McCallum. 2021. MOLEMAN: Mention-only linking of entities with a mention annotation network. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 278–285, Online. Association for Computational Linguistics.

Octavian-Eugen Ganea and Thomas Hofmann. 2017. Deep joint entity disambiguation with local neural attention. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2619–2629, Copenhagen, Denmark. Association for Computational Linguistics.

Daniel Gillick, Sayali Kulkarni, Larry Lansing, Alessandro Presta, Jason Baldridge, Eugene Ie, and Diego Garcia-Olano. 2019. Learning dense representations for entity retrieval. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 528–537, Hong Kong, China. Association for Computational Linguistics.

Seraphina Goldfarb-Tarrant, Rebecca Marchant, Ricardo Muñoz Sánchez, Mugdha Pandya, and Adam Lopez. 2021. Intrinsic bias metrics do not correlate with application bias. In *ACL*.

Chung Heong Gooi and James Allan. 2004. Cross-document coreference on a large scale corpus. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics: HLT-NAACL 2004*, pages 9–16, Boston, Massachusetts, USA. Association for Computational Linguistics.

Nitish Gupta, Sameer Singh, and Dan Roth. 2017. Entity linking via joint encoding of types, descriptions, and context. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2681–2690, Copenhagen, Denmark. Association for Computational Linguistics.

Zhengyan He, Shujie Liu, Mu Li, Ming Zhou, Longkai Zhang, and Houfeng Wang. 2013. Learning entity representation for entity disambiguation. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 30–34, Sofia, Bulgaria. Association for Computational Linguistics.

Johannes Hoffart, Yasemin Altun, and Gerhard Weikum. 2014. Discovering emerging entities with ambiguous names. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, page 385–396, New York, NY, USA. Association for Computing Machinery.

Johannes Hoffart, Dragan Milchevski, Gerhard Weikum, Avishek Anand, and Jaspreet Singh. 2016. The knowledge awakens: Keeping knowledge bases fresh with emerging entities. *Proceedings of the 25th International Conference Companion on World Wide Web*.

Heng Ji, Joel Nothman, Ben Hachey, and Radu Florian. 2015. Overview of TAC-KBP2015 tri-lingual entity discovery and linking. In *Proceedings of the 2015 Text Analysis Conference, TAC 2015, Gaithersburg, Maryland, USA, November 16-17, 2015, 2015*. NIST.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*.

Sammy Khalife and Michalis Vazirgiannis. 2018. Scalable graph-based individual named entity identification. *CoRR*, abs/1811.10547.

Nikolaos Kolitsas, Octavian-Eugen Ganea, and Thomas Hofmann. 2018. End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 519–529, Brussels, Belgium. Association for Computational Linguistics.

Angeliki Lazaridou, Adhi Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomas Kocisky, Sebastian Ruder, Dani Yogatama, Kris Cao, Susannah Young, and Phil Blunsom. 2021. Mind the gap: Assessing temporal generalization in neural language models. In *Advances in Neural Information Processing Systems*, volume 34, pages 29348–29363. Curran Associates, Inc.

Nevena Lazic, Amarnag Subramanya, Michael Ringgaard, and Fernando Pereira. 2015. Plato: A selective context model for entity resolution. *Transactions of the Association for Computational Linguistics*, 3:503–515.

Martin Josifoski Sebastian Riedel Luke Zettlemoyer Ledell Wu, Fabio Petroni. 2020. Zero-shot entity linking with dense entity retrieval. In *EMNLP*.

Belinda Z. Li, Sewon Min, Srinivasan Iyer, Yashar Mehdad, and Wen-tau Yih. 2020. Efficient one-pass end-to-end entity linking for questions. In *EMNLP*.

Robert L Logan IV, Andrew McCallum, Sameer Singh, and Dan Bikel. 2021. Benchmarking scalable methods for streaming cross document entity coreference. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4717–4731, Online. Association for Computational Linguistics.

Lajanugen Logeswaran, Ming-Wei Chang, Kenton Lee, Kristina Toutanova, Jacob Devlin, and Honglak Lee. 2019. Zero-shot entity linking by reading entity descriptions. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3449–3460, Florence, Italy. Association for Computational Linguistics.

David N. Milne and Ian H. Witten. 2008. Learning to link with wikipedia. In *CIKM '08*.

Cristina Mota and Ralph Grishman. 2009. Updating a name tagger using contemporary unlabeled data. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 353–356, Suntec, Singapore. Association for Computational Linguistics.

Ndapandula Nakashole, Tomasz Tylenda, and Gerhard Weikum. 2013. Fine-grained semantic typing of emerging entities. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1488–1497, Sofia, Bulgaria. Association for Computational Linguistics.

Yasumasa Onoe and Greg Durrett. 2019. Fine-grained entity typing for domain independent entity linking. *CoRR*, abs/1909.05780.

Yasumasa Onoe and Greg Durrett. 2020. Fine-grained entity typing for domain independent entity linking. In *AAAI*.

Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*.

Jonathan Raiman and Olivier Raiman. 2018. Deeptype: Multilingual entity linking by neural type system evolution. In *AAAI*.

Lev Ratinov, Dan Roth, Doug Downey, and Mike Anderson. 2011. Local and global algorithms for disambiguation to Wikipedia. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1375–1384, Portland, Oregon, USA. Association for Computational Linguistics.

Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. 2018. Gender bias in coreference resolution. In *NAACL*.

Sameer Singh, Amarnag Subramanya, Fernando Pereira, and Andrew McCallum. 2011. Large-scale cross-document coreference using distributed inference and hierarchical models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 793–803, Portland, Oregon, USA. Association for Computational Linguistics.

Ryan Steed, Swetasudha Panda, Ari Kobren, and Michael L. Wick. 2022. Upstream mitigation is not all you need: Testing the bias transfer hypothesis in pre-trained language models. In *ACL*.

Jiao Sun and Nanyun Peng. 2021. Men are elected, women are married: Events gender bias on wikipedia. *ArXiv*, abs/2106.01601.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015a. Modeling mention, context and entity with neural networks for entity disambiguation. In *IJCAI*.

Yaming Sun, Lei Lin, Duyu Tang, Nan Yang, Zhenzhou Ji, and Xiaolong Wang. 2015b. Modeling mention, context and entity with neural networks for entity disambiguation. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, page 1333–1339. AAAI Press.

Ledell Yu Wu, Fabio Petroni, Martin Josifoski, Sebastian Riedel, and Luke Zettlemoyer. 2020. Scalable zero-shot entity linking with dense entity retrieval. In *EMNLP*.

Zhaohui Wu, Yang Song, and C. Lee Giles. 2016. Exploring multiple feature spaces for novel entity discovery. In *AAAI*.

Jieyu Zhao, Tianlu Wang, Mark Yatskar, Vicente Ordonez, and Kai-Wei Chang. 2018. Gender bias in coreference resolution: Evaluation and debiasing methods. In *NAACL*.

A Model

In the following sections, we explain our model's architecture in detail. It relies on Blink's bi-encoder architecture (680M parameters). The model can be downloaded from:

https://github.com/facebookresearch/BLINK

The code for clustering is located here:

https://github.com/rloganiv/streaming-cdc

A.1 Mention Detection

For every span [i,j], the MD head calculates the probability of [i,j] being the mention of an entity by scoring whether i is the start of the mention, j is the end of the mention, and the tokens between i and j are the insides:

$$s_{start(i)} = \mathbf{w}_{start}^T \mathbf{p}_i$$
$$s_{end(j)} = \mathbf{w}_{end}^T \mathbf{p}_j$$
$$s_{mention(t)} = \mathbf{w}_{mention}^T \mathbf{p}_t$$

where $\mathbf{w}_{start}, \mathbf{w}_{end}, \mathbf{w}_{mention}$ are learnable vectors and $\mathbf{p}_i$ paragraph token representations based on BERT:

$$[\mathbf{p}_1 \dots \mathbf{p}_n] = BERT([CLS]p_1 \dots p_n[SEP])$$

Overall mention probabilities are computed as:

$$p([i,j]) = \sigma(s_{start(i)} + s_{end(j)} + \sum_{t=i}^{j}(s_{mention(t)}))$$

Top candidates are selected as mention candidates and propagate to the next step.

A.2 Entity Disambiguation

The ED head receives mention spans in the text and finds the best matching entity in the KB.

Following Wu et al. (2020), ED is based on dense retrieval. Description-based entity representations are computed as follows:

$$\mathbf{e} = BERT_{[CLS]}([CLS]t(e)[SEP]d(e)[SEP])$$

Following Li et al. (2020), mention representations are constructed with one pass of the encoder and without mention boundary tokens by pooling mention tokens through a single feed-forward layer (*FFL*) from the encoder output:

$$\mathbf{m}_{i,j} = FFL(\mathbf{p}_i \dots \mathbf{p}_j)$$

Similarity score s between the mention candidate and an entity candidate $e \in E$ are computed:

$$s(e,[i,j]) = \mathbf{e} * \mathbf{m}_{i,j}$$

A likelihood distribution over all entities, conditioned on the mention $[i,j]$ is computed:

$$p(e|[i,j]) = \frac{exp(s(e,[i,j]))}{\sum_{e' \in E} exp(s(e',[i,j]))}$$

$< [i,j], e^* >$, such that

$$e^* = argmax_e(p([i,j],e)),$$

are passed as a candidate < mention span, entity > tuple to the rejection head.

A.3 Rejection head

MD and ED steps over-generate. R looks at an $(e^*,[i,j])$ pair holistically decides whether to accept it. Input features to R are the MD score $p([i,j])$, the ED score $p(e^*|[i,j])$, the mention representation $\mathbf{y}_{i,j}$, top-ranked candidate representation $\mathbf{x}_{e^*}$ as well as their difference and Hadamard product. The concatenation of these features is fed through a feed-forward network to output the final entity linking score $p([i,j],e^*)$. All $p([i,j],e^*) > \gamma$ are accepted where γ is a threshold set to 0.4.

A.4 Training

Following prior work (Sun et al., 2015b; Cao et al., 2018; Gillick et al., 2019; Onoe and Durrett, 2020), training is split into two stages. First, ED only is trained on a Wikipedia dataset. This dataset is constructed by extracting Wikipedia hyperlinks to labeled mention-entity pairs and consists of 17M training samples. Then, ED, MD and R are trained jointly on the downstream dataset (either Oscar or Wikipedia). Outputs from one component are fed as input to the next and losses are summed together. To train the ED head, frozen entity representations are used. As entity embeddings do not change during training, entity embeddings can be indexed using quantization algorithms for a fast kNN search (using FAISS (Johnson et al., 2017) framework with HNSW index). A likelihood distribution over positive and mined hard negative entities for each

Parameter	Value
d_m	0.8171
s_m	0.5
d_e	110

Table 5: Hyper-parameters adaptation phase

mention is computed. Negative Log-Likelihood loss across all gold mentions in the text is used.

To train MD, binary cross-entropy loss between all possible valid spans and gold mentions in the training set is computed. Valid spans are spans with begin < end, less than a maximum length, and we also filter out spans that start or end in the middle of the word.

To train R, binary cross-entropy loss between retrieved mention-entity pairs and gold mention-entity pairs is used.

Outputs from one component are fed as input to the next and losses are summed together.

B OSCAR-based dataset

OSCAR data can be downloaded here:

https://oscar-corpus.com/

We select the following six online news pages:

BBC: https://www.bbc.com/
CNN: https://www.cnn.com/
Deutsche Welle: https://www.dw.com/en/
Reuters: https://www.reuters.com/article/
Guardian: https://www.theguardian.com/
Associated Press: https://apnews.com/article/

C Hyper-parameters adaptation phase

Using OSCAR D_{adapt}-dev, we optimize mention score threshold s_m, greedy NN distance threshold d_m and mention entity similarity threshold d_e.

We optimize s_m in range 0.0 to 1.0 in steps of 0.1 for E_U discovery recall. We optimize d_m in range 0.5 to 1.0, in steps of 0.0001 for NMI. We optimize d_e for E_U discovery recall in range 50 to 250 in steps of 10. For results, see Table 5.

We report recall of 81% and precision of 6% for clusters referring to unknown entities. Recall of clusters referring to known entities is 88% with precision 96%. Clustering NMI is 0.92.

D Adapting to the updated index

We show that by re-training L after indexing, L learns to circumvent E_U: We identify known entities part of the training set that are in close proximity of unknown entities (*confusable known entities*). We compare the average similarity between mentions and their respective linked entity when adding unknown entities before training vs. after training. Mean similarity when adding unknown entities before training is 93.28 for confusable known entities and 92.57 for other known entities. A t-test shows that this difference is significant (p-value of 0.0001 with < 0.05). As a reference, mean similarity when adding unknown entities post training is 92.65 irrespective of whether they are confusable or not.

E Disambiguation Results

Besides end-to-end performance, we also report entity disambiguation performance with oracle mention detection in Table 6.

F Wikipedia Results

We report performance on Wikipedia D_{t2}-test in Table 7. Due to a smaller D_{adapt}, end-to-end performance is lower. When filtering Wikipedia D_{t2}-test for mentions of discovered entities, L_{t1}-Cluster-Oracle precision is 40.5 and L_{t1}-Cluster recall is 15.3.

G Infrastructure, Training and Inference Details

We ran all training distributed across 8 NVIDIA TESLA V100 GPUs, each with 32 GB of memory. The first training stage took 48h, the second one 12h.

Adaptation phase is currently limited by expensive greedy NN clustering with quadratic time complexity but the type of clustering is interchangeable for more efficient ones. We chose this type of clustering as Logan IV et al. (2021) showed it performs decently for BLINK based mention encodings.

	OSCAR					
	Unknown Entities			Known Entities		
Model	**R**	**P**	**NMI**	**R**	**P**	**NMI**
L_{Dt1}	0.0	0.0	0.0	92.2	92.2	96.0
L_{Dt2}	63.5	45.3	96.8	90.0	90.2	96.0
L_{t1}-Descp	58.0	33.9	96.3	92.1	92.3	96.1
L_{t1}-Mention	26.2	30.8	92.7	92.2	92.2	96.0
EDIN (L_{t1}-Cluster)	27.9	34.1	93.4	92.2	92.2	96.2

Table 6: **Entity Disambiguation performance** on OSCAR D_{t2}-test.

	Unknown Entities			Known Entities		
Model	**R**	**P**	**NMI**	**R**	**P**	**NMI**
L_{t1}	0.0	0.0	0.0	70.5	75.8	95.4
L_{t2}	33.6	25.0	98.3	70.6	75.4	95.3
L_{t1}-Descp	33.9	20.0	98.0	71.2	74.4	95.3
L_{t1}-Cluster-Oracle	7.8	55.6	90.6	70.1	75.9	95.6
EDIN (L_{t1}-Cluster)	1.8	15.4	93.4	71.1	74.1	95.3

Table 7: **End-to-end EL performance** on Wikipedia D_{t2}-test.

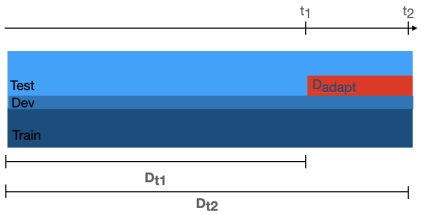


Figure 2: **Dataset splits:** A schema illustrating the composition of D_{t1} and D_{t2}. Note, that contrary to what this plot suggests, the number of samples per data split is equal for D_{t1} and D_{t2}.

Bibliography

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. *Computing Research Repository*, arXiv:1607.06450.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *The 3rd International Conference on Learning Representations*, San Diego, USA. OpenReview.net.

Lukas Berglund, Meg Tong, Maximilian Kaufmann, Mikita Balesni, Asa Cooper Stickland, Tomasz Korbak, and Owain Evans. 2024. The reversal curse: LLMs trained on "a is b" fail to learn "b is a". In *The 12th International Conference on Learning Representations*, Vienna, Austria. OpenReview.net.

Su Lin Blodgett, Solon Barocas, Hal Daumé III, and Hanna Wallach. 2020. Language (technology) is power: A critical survey of "bias" in NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5454–5476, Online. Association for Computational Linguistics.

Bernd Bohnet, Vinh Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Massimiliano Ciaramita, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, Tom Kwiatkowski, Ji Ma, Jianmo Ni, Tal Schuster, Lierni Sestorain Saralegui, William Weston Cohen, Michael Collins, Dipanjan Das, Don Metzler, Slav Petrov, and Kellie Webster. 2023. Attributed question answering: Evaluation and modeling for attributed large language models. *Computing Research Repository*, arXiv:2212.08037.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego De Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack Rae, Erich Elsen, and Laurent Sifre. 2022. Improving language models by retrieving from trillions of tokens. In

Proceedings of the 39th International Conference on Machine Learning, pages 2206–2240, Baltimore, USA. Proceedings of Machine Learning Research.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. COMET: Commonsense transformers for automatic knowledge graph construction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4762–4779, Florence, Italy. Association for Computational Linguistics.

Ronald Brachman and Hector Levesque. 2004. *Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers Inc., San Francisco, USA.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Nicholas Carlini, Daphne Ippolito, Matthew Jagielski, Katherine Lee, Florian Tramer, and Chiyuan Zhang. 2023. Quantifying memorization across neural language models. In *The 11th International Conference on Learning Representations*, Kigali, Rwanda. OpenReview.net.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. 2014. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, 8th Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar. Association for Computational Linguistics.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Benton C. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark

Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Díaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24:240:1–240:113.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Silviu Cucerzan. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 708–716, Prague, Czech Republic. Association for Computational Linguistics.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Bhavana Dalvi, Peter Alexander Jansen, Oyvind Tafjord, Zhengnan Xie, Hannah Smith, Leighanna Pipatanangkura, and Peter Clark. 2021. Explaining answers with entailment trees. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7358–7370, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Joe Davison, Joshua Feldman, and Alexander M Rush. 2019. Commonsense knowledge mining from pretrained models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 1173–1178, Hong Kong, China. Association for Computational Linguistics.

Nicola De Cao, Wilker Aziz, and Ivan Titov. 2021. Editing factual knowledge in language models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6491–6506, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. 2022. Time-aware language models as temporal knowledge bases. In *Transactions of the Association for Computational Linguistics*, volume 10, pages 257–273, Cambridge, MA. MIT Press.

Yanai Elazar, Shauli Ravfogel, Alon Jacovi, and Yoav Goldberg. 2021. Amnesic probing: Behavioral explanation with amnesic counterfactuals. In *Transactions of the Association for Computational Linguistics*, volume 9, pages 160–175, Cambridge, MA. MIT Press.

J. R. Firth. 1957. A synopsis of linguistic theory 1930-55. In *Studies in Linguistic Analysis (special volume of the Philological Society)*, volume 1952-59, pages 1–32, Oxford. The Philological Society.

Google Gemini Team. 2023. Gemini: A family of highly capable multimodal models. *Computing Research Repository*, arXiv:2312.11805.

Google Gemini Team. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *Computing Research Repository*, arXiv:2403.05530.

Mor Geva, Jasmijn Bastings, Katja Filippova, and Amir Globerson. 2023. Dissecting recall of factual associations in auto-regressive language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 12216–12235, Singapore. Association for Computational Linguistics.

Yoav Goldberg. 2019. Assessing BERT's syntactic abilities. *Computing Research Repository*, arXiv:1901.05287.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: Retrieval-augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3929–3938, Online. Proceedings of Machine Learning Research.

John Hewitt and Christopher D. Manning. 2019. A structural probe for finding syntax in word representations. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, USA. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard De Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, et al. 2021. Knowledge graphs. *ACM Computing Surveys*, 54(4):1–37.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2022. Unsupervised dense information retrieval with contrastive learning. *Transactions on Machine Learning Research*.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics*, pages 874–880, Online. Association for Computational Linguistics.

Myeongjun Jang and Thomas Lukasiewicz. 2023. Consistency analysis of ChatGPT. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 15970–15985, Singapore. Association for Computational Linguistics.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? In *Transactions of the Association for Computational Linguistics*, volume 8, pages 423–438, Cambridge, MA. MIT Press.

Daniel Jurafsky and James H. Martin. 2009. *Speech and language processing*, 2nd edition. Prentice Hall Series in Artificial Intelligence. Prentice Hall, Pearson Education International, London.

Nora Kassner, Oyvind Tafjord, Ashish Sabharwal, Kyle Richardson, Hinrich Schuetze, and Peter Clark. 2023. Language models with rationality. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14190–14201, Singapore. Association for Computational Linguistics.

Nora Kassner, Oyvind Tafjord, Hinrich Schütze, and Peter Clark. 2021. BeliefBank: Adding memory to a pre-trained language model for a systematic notion of belief. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8849–8861, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *The 8th International Conference on Learning Representations*, Online. OpenReview.net.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. *Computing Research Repository*, arXiv/1909.11942.

Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, Jörg Frohberg, Mario Šaško, Quentin Lhoest, Angelina McMillan-Major, Gérard Dupont, Stella Biderman, Anna Rogers, Loubna Ben allal, Francesco De Toni, Giada Pistilli, Olivier Nguyen, Somaieh Nikpoor, Maraim Masoud, Pierre Colombo, Javier de la Rosa, Paulo Villegas, Tristan Thrush, Shayne Longpre, Sebastian Nagel, Leon Weber, Manuel Romero Muñoz, Jian Zhu, Daniel Van Strien, Zaid Alyafeai, Khalid Almubarak, Vu Minh Chien, Itziar Gonzalez-Dios, Aitor Soroa, Kyle Lo, Manan Dey, Pedro Ortiz Suarez, Aaron Gokaslan, Shamik Bose, David Ifeoluwa Adelani, Long Phan, Hieu Tran, Ian Yu, Suhas Pai, Jenny Chim, Violette Lepercq, Suzana Ilic, Margaret Mitchell, Sasha Luccioni, and Yacine Jernite. 2022. The BigScience ROOTS corpus: A 1.6TB composite multilingual dataset. In *Advances in Neural Information Processing Systems*, volume 36. Curran Associates, Inc.

Angeliki Lazaridou, Elena Gribovskaya, Wojciech Stokowiec, and Nikolai Grigorev. 2022. Internet-augmented language models through few-shot prompting for open-domain question answering. *Computing Research Repository*, arXiv:2203.05115.

Angeliki Lazaridou, Adhiguna Kuncoro, Elena Gribovskaya, Devang Agrawal, Adam Liska, Tayfun Terzi, Mai Gimenez, Cyprien de Masson d'Autume, Tomás Kociský, Sebastian Ruder, Dani Yogatama, Kris Cao, Susannah Young, and Phil Blunsom. 2021. Mind the gap: Assessing temporal generalization in neural language models. In *Advances in Neural Information Processing Systems*, volume 35. Curran Associates, Inc.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems*, volume 34, Red Hook, USA. Curran Associates Inc.

Tao Li, Vivek Gupta, Maitrey Mehta, and Vivek Srikumar. 2019. A logic-driven framework for consistency of neural models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3924–3935, Hong Kong, China. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. *Computing Research Repository*, arXiv:1907.11692.

Adam Livska, Tom'avs Kovcisk'y, Elena Gribovskaya, Tayfun Terzi, Eren Sezener, Devang Agrawal, Cyprien de Masson d'Autume, Tim Scholtes, Manzil Zaheer, Susannah Young, Ellen Gilsenan-McMahon, Sophia Austin, Phil Blunsom, and Angeliki Lazaridou. 2022. Streamingqa: A benchmark for adaptation to new knowledge over time in question answering models. In *Proceedings of the 39th International Conference on Machine Learning*, Baltimore, USA. Proceedings of Machine Learning Research.

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, volume 30, pages 6294–6305, Long Beach, USA. Curran Associates Inc.

Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. 2022. Locating and editing factual associations in gpt. In *Advances in Neural Information Processing Systems*, volume 35, pages 17359–17372, New Orleans, USA. Curran Associates Inc.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? A new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, Brussels, Belgium. Association for Computational Linguistics.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. In *The 1st International Conference on Learning Representations*, Scottsdale, USA. OpenReview.net.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 5783–5797, Online. Association for Computational Linguistics.

Eric Mitchell, Charles Lin, Antoine Bosselut, Christopher D Manning, and Chelsea Finn. 2022. Memory-based model editing at scale. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 15817–15831, Baltimore, USA. Proceedings of Machine Learning Research.

Reiichiro Nakano, Jacob Hilton, Suchir Balaji, Jeff Wu, Long Ouyang, Christina Kim, Christopher Hesse, Shantanu Jain, Vineet Kosaraju, William Saunders, Xu Jiang, Karl Cobbe, Tyna Eloundou, Gretchen Krueger, Kevin Button, Matthew Knight, Benjamin Chess, and John Schulman. 2019. Webgpt: Browser-assisted question-answering with human feedback. *Computing Research Repository*, arXiv:2112.09332.

OpenAI. 2024. GPT-4 technical report. *Computing Research Repository*, arXiv:2303.08774.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul F Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, volume 35, pages 27730–27744. Curran Associates, Inc.

Matthew Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, pages 1756–1765, Vancouver, Canada. 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*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Matthew E Peters, Mark Neumann, Robert Logan, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 43–54, Online. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*, pages 2463–2473, Online. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *Open AI Blog*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Simon Razniewski, Andrew Yates, Nora Kassner, and Gerhard Weikum. 2021. Language models as or for knowledge bases. *Deep Learning for Knowledge Graphs workshop at 20th International Semantic Web Conference*.

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, page 5418–5426, Online. Association for Computational Linguistics.

Stephen E Robertson. 1997. Overview of the okapi projects. *Journal of documentation*, 53(1):3–7.

Timo Schick. 2023. Few-shot learning mit Sprachmodellen. In *Ausgezeichnete Informatikdissertationen 2022 (Band D23)*, pages 251–260. Gesellschaft für Informatik e.V., Bonn, Germany.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and Korean voice search. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 5149–5152, Kyoto, Japan. Institute of Electrical and Electronics Engineers.

Eric Schwitzgebel. 2024. Belief. In Edward N. Zalta and Uri Nodelman, editors, *The Stanford Encyclopedia of Philosophy*, Spring 2024 edition. Metaphysics Research Lab, Stanford University.

Kurt Shuster, Mojtaba Komeili, Leonard Adolphs, Stephen Roller, Arthur Szlam, and Jason Weston. 2022. Language models that seek for knowledge: Modular search & generation for dialogue and prompt completion. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, page 373–393, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Anton Sinitsin, Vsevolod Plokhotnyuk, Dmitry Pyrkin, Sergei Popov, and Artem Babenko. 2020. Editable neural networks. In *The 8th International Conference on Learning Representations*, Addis Ababa, Ethiopia. OpenReview.net.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. *Proceedings of the AAAI Conference on Artificial Intelligence, 31*, pages 4444–4451.

Martin Sundermeyer, Ralf Schlüter, and Hermann Ney. 2012. LSTM neural networks for language modeling. In *13th Annual Conference of the International Speech Communication Association*, pages 194–197, Portland, USA. International Speech Communication Association.

Oyvind Tafjord, Bhavana Dalvi, and Peter Clark. 2022. Entailer: Answering questions with faithful and truthful chains of reasoning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, page 2078–2093, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Oyvind Tafjord, Matt Gardner, Kevin Lin, and Peter Clark. 2019. QuaRTz: An open-domain dataset of qualitative relationship questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5941–5946, Hong Kong, China. Association for Computational Linguistics.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT rediscovers the classical NLP pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601, Online. Association for Computational Linguistics.

Romal Thoppilan, Daniel De Freitas, Jamie Hall, Noam Shazeer, Apoorv Kulshreshtha, Heng-Tze Cheng, Alicia Jin, Taylor Bos, Leslie Baker, Yu Du, YaGuang Li, Hongrae Lee, Huaixiu Steven Zheng, Amin Ghafouri, Marcelo Menegali, Yanping Huang, Maxim Krikun, Dmitry Lepikhin, James Qin, Dehao Chen, Yuanzhong Xu, Zhifeng Chen, Adam Roberts, Maarten Bosma, Yanqi Zhou, Chung-Ching Chang, Igor Krivokon, Will Rusch, Marc Pickett, Kathleen S. Meier-Hellstern, Meredith Ringel Morris, Tulsee Doshi, Renelito Delos Santos, Toju Duke, Johnny Soraker, Ben Zevenbergen, Vinodkumar Prabhakaran, Mark Diaz, Ben Hutchinson, Kristen Olson, Alejandra Molina, Erin Hoffman-John, Josh Lee, Lora Aroyo, Ravi Rajakumar, Alena Butryna, Matthew Lamm, Viktoriya Kuzmina, Joe Fenton, Aaron Cohen, Rachel Bernstein, Ray Kurzweil, Blaise Agüera y Arcas, Claire Cui, Marian Croak, Ed H. Chi, and Quoc Le. 2022. LaMDA: Language models for dialog applications. *Computing Research Repository*, arXiv:2201.08239.

Thinh Hung Truong, Timothy Baldwin, Karin Verspoor, and Trevor Cohn. 2023. Language models are not naysayers: an analysis of language models on negation benchmarks. In *Proceedings of the 12th Joint Conference on Lexical and Computational Semantics*, pages 101–114, Toronto, Canada. Association for Computational Linguistics.

Neeraj Varshney, Swaroop Mishra, and Chitta Baral. 2022. Towards improving selective prediction ability of NLP systems. In *Proceedings of the 7th Workshop on Representation Learning for NLP*, pages 221–226, Dublin, Ireland. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008. Curran Associates, Inc.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32, pages 3266–3280. Curran Associates, Inc.

Kevin Ro Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. 2023a. Interpretability in the wild: A circuit for indirect object identification in GPT-2 small. In *The 11th International Conference on Learning Representations*, Kigali, Rwanda. OpenReview.net.

Thomas Wang, Adam Roberts, Daniel Hesslow, Teven Le Scao, Hyung Won Chung, Iz Beltagy, Julien Launay, and Colin Raffel. 2022a. What language model architecture and pretraining objective works best for zero-shot generalization? In *Proceedings of the 39th International Conference on Machine Learning*, volume 162, pages 22964–22984, Baltimore, USA. Proceedings of Machine Learning Research.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc V Le, Ed H. Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023b. Self-consistency improves chain of thought reasoning in language models. In *The 11th International Conference on Learning Representations*, Kigali, Rwanda. OpenReview.net.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Atharva Naik, Arjun Ashok, Arut Selvan Dhanasekaran, Anjana Arunkumar, David Stap, Eshaan Pathak, Giannis Karamanolakis, Haizhi Lai, Ishan Purohit, Ishani Mondal, Jacob Anderson, Kirby Kuznia, Krima Doshi, Kuntal Kumar Pal, Maitreya Patel, Mehrad Moradshahi, Mihir Parmar, Mirali Purohit, Neeraj Varshney, Phani Rohitha Kaza, Pulkit Verma, Ravsehaj Singh Puri, Rushang Karia, Savan Doshi, Shailaja Keyur Sampat, Siddhartha Mishra, Sujan Reddy A, Sumanta Patro, Tanay Dixit, and Xudong Shen. 2022b. Super-NaturalInstructions: Generalization via declarative instructions on 1600+ NLP tasks. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 5085–5109, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yogatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol Vinyals, Percy Liang, Jeff Dean, and William Fedus. 2022a. Emergent abilities of large language models. *Transactions on Machine Learning Research*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022b. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837. Curran Associates, Inc.

Lilian Weng. 2023. Llm-powered autonomous agents. *lilianweng.github.io*.

Sohee Yang, Elena Gribovskaya, Nora Kassner, Mor Geva, and Sebastian Riedel. 2024. Do large language models latently perform multi-hop reasoning? In *Proceedings of the 62nd Annual Meeting of the Association for Computational*

Linguistics, pages 10210–10229, Bangkok, Thailand. Association for Computational Linguistics.

Wen-tau Yih, Kristina Toutanova, John C. Platt, and Christopher Meek. 2011. Learning discriminative projections for text similarity measures. In *Proceedings of the 15th Conference on Computational Natural Language Learning*, pages 247–256, Portland, Oregon, USA. Association for Computational Linguistics.