Inductive Representation Learning and Natural Language Question Answering on Temporal Knowledge Graphs

Dissertation

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

Zifeng Ding

München, 2024

Inductive Representation Learning and Natural Language Question Answering on Temporal Knowledge Graphs

Dissertation

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

vorgelegt von
Zifeng Ding
aus Shanghai, China

München, den 04.10.2024

Erstgutachter:	Prof. Dr. Volker Tresp
Zweitgutachter:	Prof. Dr. Michael Bronstein
Drittgutachter:	Prof. Dr. Quanming Yao
Tag der Disputation:	18.02.2025

Eidesstattliche Versicherung

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

Hiermit erkläre ich, Zifeng Ding, an Eides statt, dass die vorliegende Dissertation von mir selbstständig und ohne unerlaubte Beihilfe angefertigt worden ist.

München, den 04.10.2024 _____

Zifeng Ding

Contents

Abstract

Real-world applications such as recommender systems, social networks, and protein-protein interactions often involve relational data. In recent years, there has been increasing interest in machine learning on such data, particularly in the context of knowledge graphs (KGs). KGs are structured relational data that store multi-relational information as directed graphs, where each node corresponds to an entity and each labeled edge represents a factual relationship between entities, e.g., (*Oxford*, *located in*, *the United Kingdom*). Traditional KGs assume time-invariant relationships. However, real-world relationships are dynamically evolving over time. For example, the chancellor of Germany in 2020 was Angela Merkel, but in 2022 it became Olaf Scholz. This necessitates the use of temporal knowledge graphs (TKGs), where temporal facts are introduced by coupling stationary facts with additional time identifiers, e.g., (*Angela Merkel*, *is chancellor of*, *Germany*, *2020*). TKGs are more expressive than KGs as they model the temporal evolution of knowledge. Consequently, recent research has paid more attention to machine learning on TKGs. In this thesis, we focus on two machine learning problems: inductive knowledge representation learning and natural language question answering (QA) on TKGs.

Knowledge representation learning is a successful paradigm for TKG modeling, where models learn low-dimensional embedding vectors, i.e., representations, to represent entities and relations based on observed TKG facts. While embedding-based models excel in downstream tasks, they are limited to transductive learning, meaning they can only learn representations for entities and relations present during training. In real-world scenarios, TKGs evolve, and new entities and relations emerge that are unseen during training. Traditional embedding-based TKG models cannot perform inductive learning, which involves handling these new entities and relations. This thesis explores how to design TKG embedding models to achieve inductive learning using modern machine learning techniques. We develop advanced algorithms to significantly enhance the inductive capabilities of knowledge representations.

Abstract

Natural language question answering (QA) with machines has become a critical research area due to the growing power of language models (LMs). LMs are promising for natural language QA as they incorporate extensive background knowledge from vast textual data. QA over TKGs (TKGQA), a subcategory of natural language QA, aims to answer knowledge-intensive temporal questions based on TKGs. Major research in TKGQA typically assumes that answers to questions are accessible from the underlying TKG, allowing LMs to provide perfect answers by retrieving the correct piece of knowledge. However, this is not always the case in real-world applications. For instance, a QA system that helps to predict potential political crises should analyze past relationships between countries to forecast future trends. This introduces the concept of forecasting TKGQA, where LMs cannot rely solely on information retrieval, as the ground truth for future events is not yet available. In this thesis, we comprehensively study the task of forecasting TKGQA. We build a benchmark dataset and a coupled QA model, comparing previous TKGQA methods with our model to highlight the challenges and potential solutions for this new task.

Specifically, this thesis discusses the following contents in details:

First, we introduce the problem of inductive entity representation learning in TKG modeling. We formally define the TKG few-shot out-of-graph (OOG) link prediction task, which tests the inductive power of knowledge representation learning models. In this task, models predict facts involving entities that are unseen during training and then emerge with a few observed edges. To address this, we develop a TKG inductive learning model trained with a meta-learning algorithm. Our approach uses a time-aware graph encoder based on graph neural networks and another message-passing module to extract entity concepts from knowledge bases.

Next, we improve inductive entity representation learning with confidence-augmented reinforcement learning (RL). We train an RL-based model within a meta-learning framework for TKG few-shot OOG link prediction. Our approach employs a Transformer with time-aware positional encoding to capture few-shot information for learning representations of newly-emerged entities. The model follows a learned policy for graph traversal within a TKG, guided by a concept regularizer leveraging entity concepts from knowledge bases. To better address the data scarcity problem in the few-shot setting, we introduce a module that computes the confidence of each candidate action during graph traversal, integrating it into the policy for action selection. Experimental results demonstrate significant improvements over previous methods.

We then explore inductive representation learning for TKG relations, defining the zero-shot TKG forecasting task where models predict links involving previously unseen relations. We use large language models (LLMs) to enrich textual relation descriptions provided by temporal knowledge bases and then generate LLM-empowered relation representations. These text-based relation representations are aligned to the graph representation space to enhance the inductive capabilities of knowledge representation learning methods. By integrating our approach with various TKG embedding-based models, we observe a significant enhancement in their inductive power.

Finally, we study natural language question answering over TKGs in the forecasting setting. We define the task of forecasting TKGQA, constructing a TKG from the Integrated Crisis Early Warning System and generating a large-scale benchmark dataset, i.e., ForecastTKGQuestions, with 727k time-related questions requiring forecasting power to answer. The benchmark includes diverse question types: entity prediction, yes-unknown, and fact reasoning questions. To perform forecasting TKGQA, we develop a dedicated model, i.e., ForecastTKGQA, by combining a pre-trained LM with a TKG representation learning model trained for forecasting. Our model demonstrates strong performance, highlighting the importance of building knowledge-aware QA models with forecasting capabilities.

Abstract

Zusammenfassung

Echtweltanwendungen wie Empfehlungssysteme, soziale Netzwerke und Protein-Protein-Interaktionen beinhalten häufig relationale Daten. In den letzten Jahren hat Maschinelles Lernen auf solchen Daten, insbesondere im Kontext von Wissensgraphen, zunehmend an Interesse gewonnen. Wissensgraphen sind strukturierte relationale Daten, die mehrrelationale Informationen als gerichtete Graphen speichern, wobei jeder Knoten einer Entität entspricht und jede beschriftete Kante eine faktische Beziehung zwischen Entitäten darstellt, z. B. (*Oxford*, *befindet sich in*, *das Vereinigte Königreich*). Traditionelle Wissensgraphen gehen von zeitinvarianten Beziehungen aus. Allerdings entwickeln sich reale Beziehungen dynamisch im Laufe der Zeit. Zum Beispiel war Angela Merkel im Jahr 2020 Bundeskanzlerin von Deutschland, aber 2022 wurde Olaf Scholz Kanzler. Dies erfordert die Verwendung von temporalen Wissensgraphen, bei denen zeitliche Fakten eingeführt werden, indem stationäre Fakten mit zusätzlichen Zeitangaben verknüpft werden, z. B. (*Angela Merkel*, *ist Kanzler(-in) von*, *Deutschland*, *2020*). Temporalen Wissensgraphen sind ausdrucksstärker als Wissensgraphen, da sie die zeitliche Entwicklung von Wissen modellieren. Folglich hat die Forschung in letzter Zeit vermehrt maschinelles Lernen auf temporalen Wissensgraphen untersucht. In dieser Arbeit konzentrieren wir uns auf zwei maschinelle Lernprobleme: induktives Wissensrepräsentationslernen und natürlichsprachliches Frage-Antworten auf temporalen Wissensgraphen.

Das Wissensrepräsentationslernen ist ein erfolgreiches Paradigma für die Modellierung von temporalen Wissensgraphen, bei dem Modelle nieder-dimensionale Einbettungsvektoren, d. h. Repräsentationen, lernen, um Entitäten und Relationen basierend auf die beobachteten Fakten von temporalen Wissensgraphen darzustellen. Während einbettungsbasierte Modelle in nachgelagerten Aufgaben hervorragend abschneiden, sind sie auf transduktives Lernen beschränkt, was bedeutet, dass sie nur Repräsentationen für Entitäten und Relationen lernen können, die während des Trainings vorhanden sind. In realen Szenarien entwickeln sich temporale Wissensgraphen weiter, und neue Entitäten und Re-

lationen tauchen auf, die im Training nicht gesehen wurden. Traditionelle einbettungsbasierte Modelle von temporalen Wissensgraphen können kein induktives Lernen durchführen, dass es ermöglicht, mit diesen neuen Entitäten und Relationen umzugehen. Diese Arbeit untersucht, wie man Einbettungsmodelle von temporalen Wissensgraphen so entwerfen kann, dass sie mithilfe moderner maschineller Lerntechniken induktives Lernen ermöglichen. Wir entwickeln fortschrittliche Algorithmen, um die induktiven Fähigkeiten von Wissensrepräsentationen erheblich zu verbessern.

Natürlichsprachliches Frage-Antworten mit Maschinen ist aufgrund der wachsenden Leistungsfähigkeit von Sprachmodellen zu einem wichtigen Forschungsbereich geworden. Sprachmodelle sind vielversprechend für das natürlichsprachliche Frage-Antworten, da sie umfangreiches Hintergrundwissen aus großen Textdaten einbeziehen. Frage-Antworten über temporale Wissensgraphen, eine Unterkategorie des natürlichsprachlichen Frage-Antwortens, zielt darauf ab, wissensintensive temporale Fragen auf Basis temporaler Wissensgraphen zu beantworten. Die meiste Forschung in diesem Bereich geht davon aus, dass Antworten auf Fragen aus dem zugrunde liegenden temporalen Wissensgraph zugänglich sind, sodass Sprachmodelle perfekte Antworten liefern können, indem sie das richtige Stück Wissen abrufen. Dies ist jedoch in realen Anwendungen nicht immer der Fall. Ein Frage-Antworten-System, das beispielsweise potenzielle politische Krisen vorhersagen soll, muss vergangene Beziehungen zwischen Ländern analysieren, um zukünftige Trends vorherzusagen. Dies führt zum Konzept des Vorhersage-Frage-Antwortens über temporale Wissensgraphen, bei dem Sprachmodelle nicht allein auf die Informationsbeschaffung angewiesen sein können, da die Wahrheit für zukünftige Ereignisse noch nicht verfügbar ist. In dieser Arbeit untersuchen wir umfassend die Aufgabe des Vorhersage-Frage-Antwortens über temporale Wissensgraphen. Wir erstellen einen Benchmark-Datensatz und ein gekoppeltes Frage-Antworten-Modell, und vergleichen frühere Methoden mit unserem Modell, um die Herausforderungen und potenziellen Lösungen für diese neue Aufgabe hervorzuheben.

Insbesondere behandelt diese Arbeit die folgenden Inhalte im Detail:

Zunächst führen wir das Problem des induktiven Entitätsrepräsentationslernens in der Modellierung der temporalen Wissensgraphen ein. Wir definieren formell die Aufgabe Few-Shot-Out-of-Graph-Link-Prädiktion für temporale Wissensgraphen, die die induktive Leistungsfähigkeit von Wissensrepräsentationslernmodellen testet. In dieser Aufgabe prädizieren Modelle Fakten, die Entitäten beinhalten, die während des Trainings nicht gesehen wurden und dann mit wenigen beobachteten Fakten auftauchen. Um dies zu

lösen, entwickeln wir ein Induktivlernmodell, das mit einem Meta-Learnen-Algorithmus trainiert wird. Unser Ansatz verwendet einen zeitbewussten Graphenkodierer, der auf Graph-Neuronale Netze basiert, sowie ein weiteres Nachrichtenweiterleitungsmodul, um Entitätskonzepte aus Wissensbasen zu extrahieren.

Als Nächstes verbessern wir das induktive Entitätsrepräsentationslernen in temporalen Wissensgraphen mit vertrauensverstärktem Bestärkendem Learnen. Wir trainieren ein Bestärkendes-Learnen-basiertes Modell im Rahmen eines Meta-Learnen-Ansatzes für die Few-Shot-Out-of-Graph-Link-Prädiktion Aufgabe. Unser Ansatz verwendet einen Transformer mit zeitbewusster Positionskodierung, um Few-Shot-Informationen zu erfassen und Repräsentationen neu aufgetauchter Entitäten zu lernen. Das Modell folgt einer gelernten Policy zur Graph-Traversierung innerhalb eines temporalen Wissensgraphs, geleitet von einem Konzeptregularisierer, der Entitätskonzepte aus Wissensbasen nutzt. Um das Problem des Datenmangels im Few-Shot-Learnen besser zu adressieren, führen wir ein Modul ein, das das Vertrauen jeder möglichen Aktion während der Graph-Traversierung berechnet und es in die Policy zur Aktionsauswahl integriert. Experimentelle Ergebnisse zeigen signifikante Verbesserungen gegenüber früheren Methoden.

Wir untersuchen dann das induktive Repräsentationslernen für Relationen in den temporalen Wissensgraphen und definieren die Zero-Shot-Vorhersage-Aufgabe, bei der Modelle Fakten vorhersagen, die zuvor unsichtbare Relationen beinhalten. Wir nutzen große Sprachmodelle, um textbasierte Relationsbeschreibungen, die von temporalen Wissensbasen bereitgestellt werden, anzureichern und sprachmodellgestützte Repräsentationen von Relationen zu generieren. Diese textbasierten Repräsentationen werden an den Graph-Repräsentationsraum ausgerichtet, um die induktiven Fähigkeiten von Wissensrepräsentationslernmethoden zu verbessern. Durch die Integration unseres Ansatzes in verschiedene Einbettungsmodelle der temporalen Wissensgraphen beobachten wir eine signifikante Steigerung ihrer induktiven Leistungsfähigkeit.

Abschließend untersuchen wir das natürlichsprachliche Frage-Antworten über temporale Wissensgraphen für Vorhersagen. Wir definieren die Aufgabe des Vorhersage-Frage-Antwortens über temporale Wissensgraphen, konstruieren einen temporalen Wissensgraph aus dem Integrated Crisis Early Warning System und generieren einen groß angelegten Benchmark-Datensatz mit 727 Tausend zeitbezogenen Fragen, die Vorhersagekraft zur Beantwortung erfordern. Der Benchmark umfasst verschiedene Fragetypen: Entitätsvorhersagen, Ja-Unbekannt-Fragen und Faktenverknüpfungsfragen. Um die neue Aufgabe durchzuführen, entwickeln wir ein spezielles Modell, ForecastTKGQA, indem wir ein vor-

trainiertes Sprachmodell mit einem für die Vorhersage trainierten Wissensrepräsentationslernmodell kombinieren. Unser Modell zeigt starke Leistungen und unterstreicht die Bedeutung des Aufbaus von wissensbewussten Frage-Antworten-Modellen mit Vorhersagefähigkeiten.

Acknowledgments

This thesis is the result of my three-year journey with Ludwig Maximilian University of Munich and Siemens AG. Throughout this period, I have been fortunate to receive support from many incredible individuals, whose contributions made the completion of this thesis possible.

First, I would like to express my sincere gratitude to my supervisor, Prof. Dr. Volker Tresp, who has consistently supported me in exploring intriguing research directions and defining my own research path. Volker has always been ready to help whenever I encountered challenges in my research and has also provided invaluable guidance, much like a parental figure, throughout these years. I also want to express my appreciation to Prof. Dr. Michael Bronstein and Prof. Dr. Quanming Yao for graciously agreeing to serve as my external thesis examiners.

I would like to extend my gratitude to Siemens AG for funding my doctoral study. I am thankful to Dr. Steffen Lamparter for welcoming me into his research group following Volker's retirement from Siemens. I also want to acknowledge the support of the European Laboratory for Learning and Intelligent Systems (ELLIS) and the European Network of AI Excellence Centres (ELISE) for facilitating my visit to the University of Oxford. Special thanks to Prof. Dr. Michael Bronstein for serving as my secondary supervisor at Oxford and for providing me with extensive research resources and connections to world-renowned scholars.

I am deeply grateful to my project collaborators, particularly Dr. Yunpu Ma, Dr. Yushan Liu, and Dr. Bo Xiong, who have guided me in producing high-quality research while effectively balancing academic and industry-related challenges. I also want to express my appreciation to the Master's students I have supervised, who have made significant contributions to my research projects. These include, in chronological order, Bailan He, Jingpei Wu, Ruoxia Qi, Zongyue Li, Jingcheng Wu, Heling Cai, Yaomengxi Han, and Yifeng Li. Additionally, I would like to thank Dr. Yuan He for being my closest friend at

Oxford and making my visit an unforgettable experience.

Last but not least, I would like to thank my parents, Rong Zhang and Jiehua Ding, for their unconditional love and encouragement. I am also deeply thankful to my beloved partner, Viktoria Novikova, who has consistently provided emotional support during times of great pressure.

List of Publications and Declaration of Authorship

- **Zifeng Ding***, Jingpei Wu*, Bailan He, Yunpu Ma, Zhen Han and Volker Tresp. Few-Shot Inductive Learning on Temporal Knowledge Graphs using Concept-Aware Information. In *4th Conference on Automated Knowledge Base Construction*, 2022. *Equal Contribution. URL: `https://www.akbc.ws/2022/assets/pdfs/6_few_shot_inductive_learning_on.pdf`

  > *I conceived of the original research contributions. Jingpei Wu and I performed implementations, experiments and evaluations. I wrote the initial draft of the manuscript and did all of the subsequent corrections. All co-authors discussed this work regularly and contributed to improving the manuscript.*

  This publication serves as Chapter 3 of this thesis.

- **Zifeng Ding***, Jingpei Wu*, Zongyue Li, Yunpu Ma, and Volker Tresp. Improving Few-Shot Inductive Learning on Temporal Knowledge Graphs using Confidence-Augmented Reinforcement Learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2023. *Equal Contribution. DOI: 10.1007/978-3-031-43418-1_33

  > *I conceived of the original research contributions. Jingpei Wu and I performed implementations, experiments and evaluations. I wrote the initial draft of the manuscript and did most of the subsequent corrections. Jingpei Wu and Zongyue Li did part of the manuscript corrections. All co-authors discussed this work regularly and contributed to improving the manuscript.*

This publication serves as Chapter 4 of this thesis.

- **Zifeng Ding***, Heling Cai*, Jingpei Wu, Yunpu Ma, Ruotong Liao, Bo Xiong, Volker Tresp. zrLLM: Zero-Shot Relational Learning on Temporal Knowledge Graphs with Large Language Models. In *2024 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2024. *Equal Contribution. `https://aclanthology.org/2024.naacl-long.104.pdf`

  I conceived of the original research contributions. Heling Cai and I performed implementations, experiments and evaluations. I wrote the initial draft of the manuscript and did most of the subsequent corrections. Bo Xiong did part of the manuscript corrections. All co-authors discussed this work regularly and contributed to improving the manuscript.

  This publication serves as Chapter 5 of this thesis.

- **Zifeng Ding***, Zongyue Li*, Ruoxia Qi*, Jingpei Wu, Bailan He, Yunpu Ma, Zhao Meng, Shuo Chen, Ruotong Liao, Zhen Han, Volker Tresp. ForecastTKGQuestions: A Benchmark for Temporal Question Answering and Forecasting over Temporal Knowledge Graphs. In *International Semantic Web Conference*, 2023. *Equal Contribution. DOI: 10.1007/978-3-031-47240-4_29

  I conceived of the original research contributions. Ruoxia Qi, Zongyue Li and I performed implementations, experiments and evaluations. I wrote the initial draft of the manuscript and did most of the subsequent corrections. Ruoxia Qi and Zongyue Li did part of the manuscript corrections. Ruoxia Qi, Zongyue Li, Jingpei Wu, Bailan He, Shuo Chen and I did data collection, data filtering and data labeling. All co-authors discussed this work regularly and contributed to improving the manuscript.

  This publication serves as Chapter 6 of this thesis.

Other Publications

- Zhen Han*, **Zifeng Ding***, Yunpu Ma, Yujia Gu, Volker Tresp. Learning Neural Ordinary Equations for Forecasting Future Links on Temporal Knowledge Graphs. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*,

2021. *Equal Contribution. https://aclanthology.org/2021.emnlp-main.658.pdf

- **Zifeng Ding**, Yunpu Ma, Bailan He, Zhen Han, Volker Tresp. A Simple But Powerful Graph Encoder for Temporal Knowledge Graph Completion. In *NeurIPS 2022 Temporal Graph Learning Workshop*, 2022. https://openreview.net/pdf?id=DYG8R\bgAIo

- Guirong Fu*, Zhao Meng*, Zhen Han*, **Zifeng Ding**, Yunpu Ma, Matthias Schubert, Volker Tresp, Roger Wattenhofer. TempCaps: A Capsule Network-based Embedding Model for Temporal Knowledge Graph Completion. In *Proceedings of the Workshop on Structured Prediction for NLP at Annual Meeting of the Association for Computational Linguistics*, 2022. *Equal Contribution. https://aclanthology.org/2022.spnlp-1.pdf#page=28

- Zhen Han*, Ruotong Liao*, Jindong Gu, Yao Zhang, **Zifeng Ding**, Yujia Gu, Heinz Koeppl, Hinrich Schütze, Volker Tresp. ECOLA: Enhancing Temporal Knowledge Embeddings with Contextualized Language Representations. In *Findings of the Association for Computational Linguistics*, 2023. *Equal Contribution. https://aclanthology.org/2023.findings-acl.335.pdf

- **Zifeng Ding***, Bailan He*, Jingpei Wu, Yunpu Ma, Zhen Han, Volker Tresp. Learning Meta-Representations of One-shot Relations for Temporal Knowledge Graph Link Prediction. In *2023 International Joint Conference on Neural Networks*, 2023. *Equal Contribution. DOI: 10.1109/IJCNN54540.2023.10191619

- Yan Xia*, Letian Shi*, **Zifeng Ding**, Joao F. Henriques, Daniel Cremers. Text2Loc: 3D Point Cloud Localization from Natural Language. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024. *Equal Contribution. https://openaccess.thecvf.com/content/CVPR2024/papers/Xia_Text2Loc_3D_Point_Cloud_Localization_from_Natural_Language_CVPR_2024_paper.pdf

- Bailan He, Yushan Liu, Marcel Hildebrandt, **Zifeng Ding**, Yaomengxi Han, Volker Tresp. An Automated Evaluation Framework for Graph Database Query Generation Leveraging Large Language Models. In *Third International Workshop on Linked Data-driven Resilience Research*, 2024. https://ceur-ws.org/Vol-3707/D2R224_paper_6.pdf

- Yaomengxi Han*, **Zifeng Ding***, Yushan Liu, Bailan He, Volker Tresp. Critical Path Identification in Supply Chain Knowledge Graphs with Large Language Models. In *Extended Semantic Web Conference*, 2024. *Equal Contribution. `https://2024.e\swc-conferences.org/wp-content/uploads/2024/05/77770216.pdf`

- Shuo Chen, Zhen Han, Bailan He, **Zifeng Ding**, Wenqian Yu, Philip Torr, Volker Tresp, Jindong Gu. Red Teaming GPT-4V: Are GPT-4V Safe Against Uni/Multi-Modal Jailbreak Attacks? In *ICLR 2024 Workshop on Secure and Trustworthy Large Language Models*, 2024. `https://openreview.net/forum?id=WubY1GeLij`

- Yize Sun, Jiarui Liu, Zixin Wu, **Zifeng Ding**, Yunpu Ma, Thomas Seidl, Volker Tresp. SA-DQAS: Self-attention Enhanced Differentiable Quantum Architecture Search. In *ICML 2024 Workshop on Differentiable Almost Everything: Differentiable Relaxations*, 2024. `https://differentiable.xyz/papers-2024/paper_20.pdf`

- Zefeng Wang*, Zhen Han*, Shuo Chen, Fan Xue, **Zifeng Ding**, Xun Xiao, Volker Tresp, Philip Torr, Jindong Gu. Stop Reasoning! When Multimodal LLMs with Chain-of-Thought Reasoning Meets Adversarial Images. In *First Conference on Language Modeling*, 2024. *Equal Contribution. `https://openreview.net/pdf?id=oqYiYG8PtY`

- Yongkang Liu*, Ercong Nie*, Shi Feng, Zheng Hua, **Zifeng Ding**, Daling Wang, Yifei Zhang, Hinrich Schütze. A Unified Data Augmentation Framework for Low-Resource Multi-domain Dialogue Generation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2024. *Equal Contribution. DOI: 10.1007/978-3-031-70344-7_10

- Ercong Nie*, Bo Shao*, Mingyang Wang, **Zifeng Ding**, Helmut Schmid, Hinrich Schütze. BMIKE-53: Investigating Cross-Lingual Knowledge Editing with In-Context Learning. Under review at *ACL Rolling Review*, 2025. *Equal Contribution. `https://arxiv.org/pdf/2406.17764`

- **Zifeng Ding***, Jingcheng Wu*, Jingpei Wu, Yan Xia, Volker Tresp. Temporal Fact Reasoning over Hyper-Relational Knowledge Graphs. In *Findings of the Association for Computational Linguistics: EMNLP*, 2024. *Equal Contribution. `https://aclanthology.org/2024.findings-emnlp.20.pdf`

- **Zifeng Ding***, Yifeng Li*, Yuan He, Antonio Norelli, Jingcheng Wu, Volker Tresp, Yunpu Ma, Michael Bronstein. DyGMamba: Efficiently Modeling Long-Term Temporal Dependency on Continuous-Time Dynamic Graphs with State Space Models. Under review at *Transactions on Machine Learning Research*. *Equal Contribution. `https://arxiv.org/pdf/2408.04713`

- Yuan Sui, Yufei He, **Zifeng Ding**, Bryan Hooi. Can Knowledge Graphs Make Large Language Models More Trustworthy? An Empirical Study over Open-ended Question Answering. Under review at *ACL Rolling Review*, 2025. `https://arxiv.org/pdf/2410.08085?`

- Yilun Liu, Yunpu Ma, Shuo Chen, **Zifeng Ding**, Bailan He, Zhen Han, Volker Tresp. PERFT: Parameter-Efficient Routed Fine-Tuning for Mixture-of-Expert Models. Under review at *Second Conference on Language Modeling*, 2025. `https://arxiv.org/pdf/2411.08212`

- Zhangdie Yuan, **Zifeng Ding**, Andreas Vlachos. The Future Outcome Reasoning and Confidence Assessment Benchmark. Under review at *ACL Rolling Review*, 2025. `https://arxiv.org/pdf/2502.19676`

Chapter 1

Introduction

1.1 Motivation

Artificial intelligence (AI) is a technology that empowers machines to replicate human intelligence, enabling them to perform tasks that require human cognition. This includes problem-solving, learning from experience, understanding complex data, and making decisions. Within the realm of AI, machine learning (ML) involves allowing machines to utilize data to automatically identify patterns. These learned patterns can then be used to make judgments and predictions about the world. In recent years, the volume of data has been growing at an unprecedented pace. This has resulted in dedicated datasets of high quality across various domains, making ML an increasingly popular topic. ML has been implemented in a wide range of real-world applications, such as facial recognition [1], product recommendation [88] and drug discovery [35].

One of the most common data structures is the graph, which consists of a set of nodes and a set of edges connecting these nodes. Various real-world scenarios, e.g., social networks [11] and protein-protein interactions [142], can be represented with graphs, leading to a growing interest in developing modern ML techniques for graph data. To better distinguish the relationships among nodes, increasing attention is being paid to representing data as relational graphs, where each edge denotes a specific relationship between its connecting nodes. One of the most popular types of relational graphs is knowledge graph (KG) [49]. KGs store world knowledge with triples. Each triple describes a factual statement in the form of (s, r, o), e.g., (*Oxford*, *located in*, *the United Kingdom*), where s, o are two entities represented as graph nodes and r is the relation type of the graph edge between the entities. In recent decades, a large number of KGs have emerged, e.g., Freebase [12] and DBpedia

[5]. They are widely used in a series of ML-related applications, including recommender system design [65] and natural language question answering (QA) [129], to advance the development of AI systems. Despite their popularity, traditional KGs are static and thus not suitable for representing time-varying knowledge. For example, the KG fact (*Angela Merkel*, *is Chancellor of*, *Germany*) is only valid from year *2005* to *2021* because after that *Olaf Scholz* became the Chancellor of Germany. Static KGs without specifying the temporal constraints of facts fail to capture such temporal information. To address this issue, temporal KGs (TKGs) have been introduced, incorporating a time identifier into each fact. Each TKG fact is represented as a quadruple (s, r, o, t), where t is the additional time identifier. Significant efforts have been made to construct TKGs, such as Wikidata [152] and ICEWS [14]. Based on these efforts, more recent works have begun to focus on ML within the scope of TKGs, including developing improved TKG modeling techniques with ML approaches and designing advanced algorithms that benefit from the abundant information contained in TKGs for various applications.

This thesis focuses on two underexplored TKG-related ML problems: inductive knowledge representation learning and natural language question answering (QA) on TKGs.

Inductive Knowledge Representation Learning on TKGs. Knowledge representation learning is currently the most successful ML paradigm for TKG modeling, where models learn low-dimensional embedding vectors, i.e., representations, to represent entities and relations based on observed TKG facts. Traditional embedding-based TKG models require large amounts of training data to learn optimal representations for entities and relations. As a result, these models struggle to produce meaningful representations for the entities and relations not seen during training, posing a huge challenge for inductive learning, which involves handling these newly-emerged entities and relations. Furthermore, real-world TKGs are always evolving, with new entities and relations constantly emerging [134]. This makes inductive learning crucial for effectively representing TKGs. In this thesis, we explore how to equip TKG embedding models with inductive capabilities by learning inductive knowledge representations using modern ML techniques.

Natural Language QA on TKGs. Natural language QA with machines has emerged as a vital research area due to the increasing capabilities of language models (LMs). QA over TKGs (TKGQA), a subcategory of natural language QA, asks machines to answer knowledge-intensive temporal questions based on TKGs. Previous research in TKGQA

often presumes that answers to questions can be directly retrieved from the underlying TKG, allowing TKGQA models to easily answer the questions by accessing the correct information from the graph [126]. However, in reality, humans frequently seek plans for the future, leading to situations where models are expected to answer questions about the future. This necessitates QA models to possess forecasting abilities. While several works have focused on forecasting QA in open-domain QA[1] [83], it is equally important to explore forecasting QA over TKGs (forecasting TKGQA). Forecasting TKGQA presents new challenges. TKGQA systems cannot rely solely on information retrieval since the ground truth for future events is not yet available. Therefore, it is crucial to equip the systems with specialized modules that enable forecasting. In this thesis, we comprehensively study the task of forecasting TKGQA, highlighting its challenges and potential solutions. Moreover, we discuss how to design a QA model that brings forecasting capabilities within the context of TKGQA.

1.2 Overview and Summary of Contributions

We give an overview of this thesis and summarize the main contributions. The remainder of this thesis is composed of six chapters.:

- In Chapter 2, we introduce the key concepts central to this thesis and provide a comprehensive overview of the existing literature. We begin by discussing the fundamentals of graphs (Section 2.1), followed by an introduction of meta-learning (Section 2.2) and LMs (Section 2.3), which are critical in inductive representation learning and natural language QA. Next, we explore KGs (Section 2.4) and TKGs (Section 2.5), focusing on representation learning, inductive learning, and natural language QA for both of them. We highlight the differences between KGs and TKGs, particularly in how temporal dynamics are incorporated into the modeling process.

- In Chapter 3, we introduce the problem of inductive entity representation learning on TKGs. We propose a new ML task TKG few-shot out-of-graph (OOG) link prediction to test the inductive power of knowledge representation learning models. In this task, models are asked to predict facts involving entities that are unseen during training and then emerge with a few observed edges. We show with experiments that

[1]Different from TKGQA that requires models to find the answers from the coupled TKGs, in open-domain QA, answers to the questions are inferred from additional text contexts.

traditional TKG representation learning models and inductive representation learning models targeted non-temporal KGs cannot effectively deal with unseen TKG entities. To address this issue, we develop a TKG inductive learning model named FILT that is trained with a meta-learning algorithm. FILT uses a time-aware graph encoder based on graph neural networks and another message-passing module to extract entity concepts from knowledge bases. Experimental results show that FILT substantially outperforms previous KG/TKG representation learning methods in inductive learning for TKG entities, marking a notable advancement in this area.

- In Chapter 4, we improve inductive entity representation learning with confidence-augmented reinforcement learning (RL). We train an RL-based model, i.e., FITCARL, within a meta-learning framework for TKG few-shot OOG link prediction. Our approach employs a Transformer [149] with a customized time-aware positional encoding to capture few-shot information for learning representations of few-shot entities. The model follows a learned policy for graph traversal within a TKG. The traversal is further guided by a parameter-free concept regularizer leveraging entity concepts from knowledge bases, following the idea of modeling entities' concept-aware information proposed in FILT. To better address the data scarcity problem in the few-shot setting, we introduce a module that computes the confidence of each candidate action during graph traversal, integrating it into the policy for action selection. Experimental results demonstrate that FITCARL achieves significant improvements over previous methods in inductive representation learning on TKG entities, including FILT.

- In Chapter 5, we explore inductive representation learning for TKG relations. We propose a new ML task zero-shot TKG forecasting where models are asked to predict the facts involving previously unseen relations. To solve this task, we propose a model zrLLM which is a plug-and-play model that can be implemented together with traditional TKG representation learning methods to enhance their inductive power over unseen relations. We use large LMs (LLMs) to enrich textual relation descriptions provided by temporal knowledge bases and then generate LLM-empowered relation representations. These text-based relation representations are aligned to the graph representation space and jointly trained with TKG models. To promote alignment between representation spaces, we propose a relation history learner that captures the temporal dynamics of historical relation patterns between each pair of entities.

Comprehensive experiments show that zrLLM significantly enhances the inductive capabilities of traditional TKG representation learning methods by leveraging textual information and LMs.

- In Chapter 6, we study natural language QA over TKGs in the forecasting setting. We define a new ML task forecasting TKGQA, constructing a TKG from the Integrated Crisis Early Warning System [14] knowledge base and generating a large-scale benchmark dataset, i.e., ForecastTKGQuestions, with 727k time-related questions based on the constructed TKG. Each question in ForecastTKGQuestions requires forecasting power to answer, posing a great challenge to previous TKGQA approcahes. The benchmark includes diverse question types, including entity prediction, yes-unknown, and fact reasoning questions, derived from the traditional tasks of TKG link prediction, as well as yes-no and multiple-choice questions in reading comprehension QA. To perform forecasting TKGQA, we develop a dedicated model, i.e., ForecastTKGQA, by combining a pre-trained LM with a TKG representation learning model trained for forecasting. Our model demonstrates strong performance, highlighting the importance of building knowledge-aware QA models with forecasting capabilities.
- In Chapter 7, we give a conclusion of the thesis. We also discuss the potential future directions that can be built upon our findings, offering valuable insights for the research community.

Chapter 2

Preliminaries and Related Work

In this chapter, we outline the fundamental concepts relevant to this thesis and provide an overview of existing works to better contextualize our contributions.

2.1 Graphs

2.1.1 Fundamental of Graphs

Definition 1 (Graph)**.** *Let $\mathcal{V}$ and $\mathcal{E}$ denote a set of vertices (i.e., nodes) and edges, respectively. A graph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. $e_{i,j} \in \mathcal{E}$ describes an edge pointing from the source node $v_i \in \mathcal{V}$ to the destination node $v_j \in \mathcal{V}$, where $i, j \in \{1, 2, ..., |\mathcal{V}|\}$.*

If a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is undirected, every edge $e_{i,j} \in \mathcal{E}$ can be implied by $e_{j,i}$. Otherwise, $\mathcal{G}$ is called a directed graph. In a graph, a node normally represents an item or a concept, and an edge describes the relationship or connection between a pair of nodes. A node $v_i \in \mathcal{V}$ can have node attributes, represented in the form of a feature vector $\mathbf{x}_{v_i} \in \mathbb{R}^{d_\mathcal{V}}$. Node attributes provide detailed information of a node, e.g., the class label of the node or various numerical numbers describing node characteristics. Similarly, an edge $e_{i,j}$ can also be associated with edge attributes in the form of a feature vector $\mathbf{x}_{e_{i,j}} \in \mathbb{R}^{d_\mathcal{E}}$. Edge attributes specify the details of an edge, e.g., edge type or edge weight indicating the importance of the edge. To ensure clarity, we will use the term *node* (*nodes*) instead of *vertex* (*vertices*) throughout this thesis.

Definition 2 (Adjacency Matrix)**.** *For a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, its adjacency matrix $\mathbf{A} \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{V}|}$ is a matrix with $\mathbf{A}_{i,j} = 1$ if $e_{i,j} \in \mathcal{E}$ and $\mathbf{A}_{i,j} = 0$ if $e_{i,j} \notin \mathcal{E}$. $\mathbf{A}_{i,j}$ denotes the entry in the i^{th} row and j^{th} column.*

If $\mathbf{A}$ is symmetric, then its associated graph is undirected. Otherwise, $\mathbf{A}$ implies a directed graph.

Definition 3 (Neighborhood)**.** *For a node $v_i \in \mathcal{V}$ in the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, its neighborhood $\mathcal{N}_{v_i}$ is defined as all the nodes existing in v_i's associated incoming edges: $\mathcal{N}_{v_i} = \{v_j | e_{j,i} \in \mathcal{E}\}$.*

In our definition, we only consider incoming edges (the edges pointing to the node of interest) as the neighbors following [162]. The ideas of incoming and outgoing edges (the edges pointing out from the node of interest) only exist for directed graphs. For undirected graphs, as long as two nodes are connected with an edge, they serve as a neighbor of each other.

2.1.2 Graph Neural Networks

Unlike the data structures with an underlying Euclidean structure, such as images, graphs are represented in a non-Euclidean manner and therefore require specialized tools for ML [15]. To address this, a new family of neural networks, i.e., graph neural networks (GNNs), has emerged. GNNs learn low-dimensional embedding vectors, i.e., representations, for graph nodes, depending on the graph structure, node features and edge features. We use $\mathbf{H} \in \mathbb{R}^{|\mathcal{V}| \times d}$ to denote the representations of all nodes in a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each row of $\mathbf{H}$ refers to the representation of a node. For example, the representation $\mathbf{h}_{v_i}$ of $v_i \in \mathcal{V}$ corresponds to the i^{th} row of $\mathbf{H}$. d is the dimension of the node representations.

A GNN consists of a number of L ($L \geq 1$) layers. In layer l ($0 \leq l \leq L-1; l \in \mathbb{N}$), it updates the node representations $\mathbf{H}^l \in \mathbb{R}^{|\mathcal{V}| \times d^l}$ [1] by aggregating the information provided by each node's neighborhood. After L layers, GNN's final output $\mathbf{H}^L$ (output of the $(L-1)^{\text{th}}$ layer) corresponds to the learned node representations. If a graph has node attributes, the node representations can be initialized with these attributes before input into a GNN.

We give an introduction of some classic GNN models. For simplicity, we omit the edge feature vectors:

- Graph Convolutional Network (GCN) [89] stems from graph signal processing and generalizes from the ChebNet [37] by using its first-order approximation. Each GCN

[1] The superscript of $\mathbf{H}^l$ indicates that it is the input of the l^{th} layer of GNN. d^l denotes the dimension of node representations in the l^{th} layer of GNN. We use such style of notation following previous works about GNNs such as [89].

layer updates node representations as

$$\mathbf{H}^{l+1} = \sigma\left(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{l}\mathbf{W}^{l}\right). \tag{2.1}$$

$\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$, where $\mathbf{I}$ is an identity matrix in the same size as the adjacency matrix $\mathbf{A} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$. $\mathbf{W}^{l} \in \mathbb{R}^{d^{l}\times d^{l+1}}$ is a trainable weight matrix of the l^{th} GCN layer. Each element of $\tilde{\mathbf{D}} \in \mathbb{R}^{|\mathcal{V}|\times|\mathcal{V}|}$ is computed as $\tilde{\mathbf{D}}_{i,i} = \sum_{j}\tilde{\mathbf{A}}_{i,j}$ so that $\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}$ represents the normalized adjacency matrix. $\sigma(\cdot)$ denotes an activation function.

- Message Passing Neural Network (MPNN) [61] employs the idea of neural message passing to update node representations, where information can be passed from one node to another along edges directly. It can be viewed as generalizing the convolution operation in convolutional neural networks (CNNs) [96] for images to graph learning scenarios. Each MPNN layer is defined as

$$\mathbf{h}_{v_i}^{l+1} = \text{Update}^{l}\left(\mathbf{h}_{v_i}^{l}, \mathbf{m}_{v_i}^{l+1}\right), \text{ where } \mathbf{m}_{v_i}^{l+1} = \sum_{v_j\in\mathcal{N}_{v_i}} \text{Msg}^{l}\left(\mathbf{h}_{v_i}^{l}, \mathbf{h}_{v_j}^{l}\right). \tag{2.2}$$

  $\text{Update}^{l}(\cdot)$ and $\text{Msg}^{l}(\cdot)$ are the update function and the message function of the l^{th} MPNN layer, respectively. Equation 2.2 is a process of aggregation, where a node is updated by summing over the information provided by the adjacent nodes in its neighborhood. [61] further finds that MPNN subsumes various GNN models, including GCN [89]. In recent years, MPNN has become a significant framework of GNNs and inspired a large number of follow-up works.

- Graph Attention Network (GAT) [150] introduces an attention mechanism to distinguish the importance of different neighboring nodes during message passing. Each GAT layer is defined as

$$\begin{aligned} \mathbf{h}_{v_i}^{l+1} &= \sigma\left(\sum_{v_j\in\mathcal{N}_{v_i}} \alpha_{i,j}\mathbf{W}\mathbf{h}_{v_j}^{l}\right), \\ \text{where } \alpha_{i,j} &= \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^{\top}\left(\mathbf{W}\mathbf{h}_{v_i}^{l} \| \mathbf{W}\mathbf{h}_{v_j}^{l}\right)\right)\right)}{\sum_{v_k\in\mathcal{N}_{v_i}}\exp\left(\text{LeakyReLU}\left(\mathbf{a}^{\top}\left(\mathbf{W}\mathbf{h}_{v_i}^{l} \| \mathbf{W}\mathbf{h}_{v_k}^{l}\right)\right)\right)}. \end{aligned} \tag{2.3}$$

  $\alpha_{i,j}$ represents the importance of node v_j to v_i during information aggregation. $\mathbf{W} \in \mathbb{R}^{d^{l+1}\times d^{l}}$ is a weight matrix. $\mathbf{a} \in \mathbb{R}^{2d^{l+1}}$ is a parameter vector used for computing attention. $\sigma(\cdot)$ denotes the activation function.

For discussions on a broader range of GNN types, please refer to the surveys [162, 85]. The representations learned by GNNs can be applied to various downstream ML tasks. More detailed discussions on leveraging GNNs, with an emphasis on KG and TKG representation learning, are provided in Section 2.4.3 and Section 2.5.3.

2.2 Meta-Learning

Few-shot learning (FSL) is a type of ML problem in which models are required to generalize effectively to new data for each class, provided with only a small number of labeled class-specific data examples. Meta-learning approaches aim to quickly grasp new concepts using only a few related data examples by generalizing from previously encountered learning tasks [92] and therefore are suitable for FSL. A classic meta-learning framework is episodic training [151], where an ML model is trained over a series of episodes, each serving as a mini-training process on a specific task T. In this framework, the training and test data for each task are referred to as the support set S and the query set Q, respectively. The model is trained on a set of tasks to "learn how to learn" from the support set, with the objective of minimizing a loss over the query set. Assume we have a set of N training tasks $\mathbb{T} = \{T_i\}_{i=1}^{N}$, where $T_i = \{\mathcal{S}_i, \mathcal{Q}_i\}$, the training objective of a model in episodic training is written as

$$\theta = \arg\min_{\theta} \mathbb{E}_{T_i \sim \mathbb{T}} \left[\frac{1}{|\mathcal{Q}_i|} \sum_{q \in \mathcal{Q}_i} [l_\theta(q|\mathcal{S}_i)] \right]. \tag{2.4}$$

$l_\theta(\cdot)$ is a task-specific loss function decided by the downstream task and θ denotes the model parameters. When support sets are limited to containing only a few data examples, episodic training functions as a form of few-shot training. As a result, episodic training is a powerful meta-learning tool for addressing FSL problems. Episodic training has been employed in various studies on inductive learning for KGs and TKGs. We will discuss in details in Section 2.4.6 and 2.5.6.

2.3 Language Models

A language model (LM) is a probabilistic model that estimates the likelihood of word sequences in natural language. Early LMs originate from statistical approaches, such as the N-gram model [131], which predicts the N^{th} word based on the preceding $N-1$ words. This concept has influenced the design of modern LMs, which are built to generate

the next token[2] by conditioning on the preceding tokens. Later, representation learning is introduced into language modeling, where each word is represented by a learned low-dimensional embedding vector that captures its semantic meaning. For example, Word2Vec [110] proposes two strategies for learning word embeddings: the Continuous Bag-of-Words (CBOW) model, which predicts the current word based on its surrounding context, and the Skip-gram model, which predicts surrounding words given the current word. The word embeddings learned through these strategies enable words to be contextually aware, making words with similar contexts seen in the training set close to each other in the representation space. After that, encoder-decoder structured LMs start to gain attention. An LM encoder takes the input text and outputs an encoded representation providing comprehensive understanding of the input, while the decoder takes this representation as input and generates the output token sequence (text is generated by combining tokens) depending on the computed token probabilities. One famous LM with encoder-decoder structure is Seq2Seq [141]. Both of its encoder and decoder are based on a separate long short-term memory (LSTM) network [73], which is an effective variant of recurrent neural network (RNN) [124] that is able to capture long-range information in the sequences. One advantage of using RNNs to do language modeling is that RNNs can deal with sequences with undefined lengths. An RNN recurrently incorporates the information of each element in a sequence into its hidden state in the form of a vector representation, which can be used to encode sentences. During decoding, RNN recurrently outputs a vector at each step that can be transformed into a probability distribution over token vocabulary, which enables Seq2Seq to do text generation. Building on the success of encoder-decoder architecture, the Transformer model [149] is proposed. Transformers outperform RNNs in language modeling and have become the foundational building blocks of modern LMs. We discuss in details in the next section.

2.3.1 Transformers

A Transformer [149] follows the encoder-decoder architecture. The encoder contains several identical layers, each of which is further comprised of two sub-layers: a self-attention layer and a fully connected feed-forward neural network. Self-attention layer is the core of Transformer. It can be described as follows. Consider a sequence of tokens $(w_1, ..., w_N)$, where each element w_i in this sequence is represented by a token representation (i.e., token

[2]In the context of LMs, token is a unit of text that the model processes individually. A token can be a word (e.g., dog) or subword (e.g., a character or a combination of several characters).

embedding[3]) $\mathbf{h}_{w_i} \in \mathbb{R}^d$ (d is the dimension of token embeddings). The self-attention layer updates w_i's representation by considering all tokens in the input sequence

$$\begin{aligned} \mathbf{h}_{w_i} &:= \sum_{j=1}^{N} \alpha_{i,j} \mathbf{W}_v \mathbf{h}_{w_j}, \\ \text{where } \alpha_{i,j} &= \frac{\exp\left(\frac{(\mathbf{W}_q \mathbf{h}_{w_i})^\top (\mathbf{W}_k \mathbf{h}_{w_j})}{\sqrt{d_k}}\right)}{\sum_{l=1}^{N} \exp\left(\frac{(\mathbf{W}_q \mathbf{h}_{w_i})^\top (\mathbf{W}_k \mathbf{h}_{w_l})}{\sqrt{d_k}}\right)}. \end{aligned} \tag{2.5}$$

$\alpha_{i,j}$ is called the attention of w_j when updating w_i's representation. $\mathbf{W}_q \in \mathbb{R}^{d_k \times d}$, $\mathbf{W}_k \in \mathbb{R}^{d_k \times d}$ and $\mathbf{W}_v \in \mathbb{R}^{d_v \times d}$ are three matrices computing so-called queries, keys, and values, respectively. In summary, the self-attention mechanism treats the token of interest as a query, computes attention scores based on the keys corresponding to surrounding tokens, and then produces a weighted sum according to these attention scores and the values associated with these surrounding tokens. In practice, self-attention layers are implemented in a multi-head manner. Assume there are m heads, and then in each head, the output of the weighted sum will be a representation with a dimension of d/m. The final updated representation will be a linear transformation of the concatenation of all m heads' outputs. Each head has its own set of parameters, enabling the model to capture diverse information. [149] shows that multi-head attention is effective and thus it is widely adopted in later works based on Transformers.

The decoder also contains several identical layers. Each decoding layer consists of three sub-layers: a self-attention layer performing attention over the previously generated output[4], another self-attention layer performing attention over the output of the encoder stack and a fully connected feed-forward neural network. During decoding, attention is computed over the previously generated tokens and the input sequence, rooting from the idea of next token prediction [33].

[3]Token embedding in Transformer is a combination of original token representation representing tokens' characteristics and a positional representation denoting the positions of tokens in input sequences. In [149], the positional representations are learning-free and initialized with sinusoidal functions of different frequencies.

[4]Note that Transformer is an autoregressive [63] model, meaning that it generates output token by token and each generated token is based on the input as well as the previously generated token.

2.3.2 Transformer-Based Language Models

Most successful modern LMs employ Transformers as basic building blocks. Due to the increasing availability of large-scale data, a growing number of Transformer-based LMs are first pre-trained on a large corpus with a series of pre-training tasks such as next token prediction [33] and masked language modeling (MLM) [38]. Then, a fine-tuning process is used to adjust model parameters to adapt to specific downstream tasks, requiring only relatively small tuning datasets. Pre-training enables LMs to acquire strong generalization capabilities across various tasks, as well as semantic and syntactic knowledge from the extensive pre-training corpus. We discuss several representative Transformer-based LMs here:

- BERT [38] pre-trains a bidirectional Transformer encoder on two pre-training tasks, i.e., MLM and next sentence prediction (NSP). MLM masks several randomly sampled tokens in a sequence and asks the model to generate them based on the unmasked tokens which provide textual contexts. Since the unmasked tokens can be before or after each to-be-predicted token, BERT achieves bidirectional encoding of token representations. NSP requires models to predict whether the second sentence in the input follows the first one. This task is claimed to be beneficial in several downstream tasks such as question answering. Additionally, to make the model aware of the position of tokens, BERT introduces learnable position embeddings that is added on the original token embeddings. It also prepends a [CLS] token at the beginning of every input sequence which is used to aggregate the information from the entire sequence. This allows BERT to do sentence-level downstream tasks more easily.

- RoBERTa [104] is an optimized version of BERT. It first employs a byte-level Byte-Pair Encoding (BPE) [120] in its tokenization process, which helps to represent any input text with a moderate size of vocabulary. It then employs dynamic masking when the model is pre-trained on the MLM task. BERT uses static masking for each input sequence, meaning that in different training epochs, the same input sequence is masked in the same way. Dynamic masking masks different tokens when the model is trained on the same input sequence in different epochs, leading to a more robust training process and enabling maximal utilization of data. In addition, RoBERTa collects a much larger corpus for pre-training. Experimental results show that larger size of pre-training data helps to improve model's capabilities. Finally, RoBERTa verifies that the NSP pre-training task is not necessarily beneficial for the downstream

tasks. Therefore, the loss regarding this task is discarded.

- T5 [121] is a text-to-text LM. It unifies a wide range of natural language processing (NLP) tasks as a text-to-text framework. T5 takes the input text that describes the task along with task-specific input data, and then generates a text response directly specifying the answer. To further improve model's ability of natural language understanding, T5 is pre-trained on a huge corpus, i.e., Colossal Clean Crawled Corpus (C4), consisting of hundreds of gigabytes of clean English text scraped from the web. It also proposes a new pre-training task called span-corruption. In span-corruption, spans of text are masked and the model is trained to predict the missing text. [121] shows that span-corruption is particularly effective in improving model performance on various NLP tasks. Besides these contributions, T5 tries to scale up the model size, introducing more training parameters. It demonstrates that larger LMs pre-trained on larger corpora can lead to better performance on downstream tasks due to a stronger capability in transfer learning. Note that T5 keeps an encoder-decoder architecture. This allows users to extract the output of its encoder for downstream tasks just as how BERT and RoBERTa are implemented.

- GPT-3 [16] scales up to 175 billion parameters and is pre-trained on a vast and diverse 570GB corpora . It shows that a large pre-trained LM can be viewed as a few-shot learner. Given a small number of examples and a prompt[5] related to a downstream task as input, GPT-3 can reason over the provided examples and generate answers of high quality. This process is named as in-context learning, which has become a popular pipeline in modern NLP. GPT-3 is a decoder-only model, meaning that it does not generate intermediate representations of the input text, unlike the encoders from previous works, e.g., BERT and T5. This allows GPT-3 to focus on generating text autoregressively.

- LLaMA [144] is designed to be a decoder-only LM that is significantly smaller[6] than large models like GPT-3. It shows that by optimizing the training method and model structure, LMs with fewer parameters can also achieve competitive performance. One advantage of smaller language models is that they are much easier to

[5] A prompt in this context can be understood as an input aiming to elicit desired responses from LMs.

[6] However, LLaMA is still large compared with the previous generation of LMs, e.g., BERT. LLaMA has four variants, i.e., LLaMA-7B, LLaMA-13B, LLaMA-33B and LLaMA-65B, that contain 7 billion, 13 billion, 33 billion and 65 billion trainable parameters, respectively.

implement for inference. Compared with previous excessively large LMs, LLaMA is less demanding on hardware, making it more accessible for research and practical applications. LLaMA demonstrates that pre-normalization before each Transformer sub-layer, switching the activation function from ReLU to SwiGLU [132] and adopting rotary positional embeddings [135] can help improve model performance. It also demonstrates that pre-training LMs on larger datasets enables smaller models to continually improve, allowing them to achieve performance comparable to very large models.

- Mixtral 8×7B [82], a recent decoder-only LM, employs a mixture-of-experts (MoE) structure, where each expert is a separate feed-forward neural network with 7 billion parameters. During inference, 2 out of 8 experts are activated to generate response based on a trained router that decides which experts should be used. This enables the model to achieve high computational efficiency during inference while still preserving strong capabilities of language modeling.

As the development of LM progresses, we witness a trend towards increasing model sizes and more diverse, larger-scale training data, leading to the emergence of large language models (LLMs). LLMs have gained great attention due to their potential as foundational tools in building intelligent AI agents and assisting humans in various real-world applications. This motivation has driven a surge in the number of new LLMs (or multimodal LLMs[7]) built upon Transformer, such as Gemini [3]. Meanwhile, efforts have been made to develop more advanced versions of existing models, e.g., the recent GPT-4 [117], Claude 3[8] and LLaMA 3 [48]. These advancements highlight the ongoing innovation and rapid growth in the field of language modeling.

In this thesis, we leverage Transformer-based LMs to solve ML tasks related to TKGs. In Chapter 6, we discuss how we use them together with underlying temporal knowledge bases to achieve natural language question answering (See Section 2.5.5 for preliminaries and task definition). And in Chapter 5, we show how we use LLMs to promote inductive learning on TKGs (See Section 2.5.6 for preliminaries and task definition).

[7]Recent advances lead to the development of multimodal LLMs, which integrate text with other modalities such as images, videos, and speech. Despite their expanded capabilities, we still classify them as LLMs since they originate from language modeling.

[8]https://www.anthropic.com/news/claude-3-family

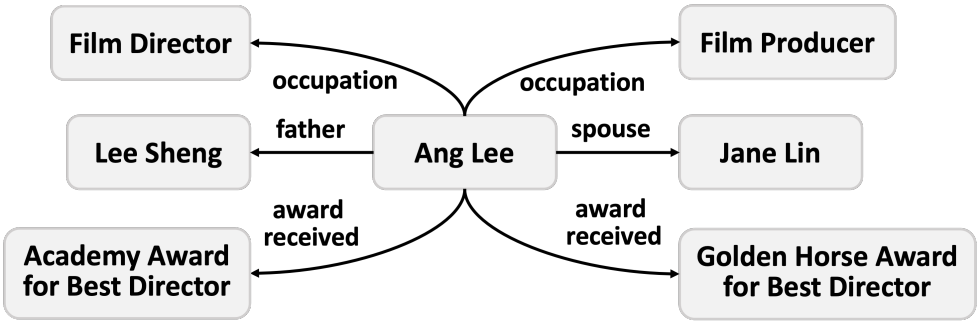


Figure 2.1: An example KG. All the facts are taken from Wikidata [152]. Each grey rectangle denotes a node (here is an entity) and the content beside each directed arrow serves as an edge label, i.e., relation type.

2.4 Knowledge Graphs

2.4.1 Fundamental of Knowledge Graphs

Definition 4 (Knowledge Graph)**.** *Let $\mathcal{E}$ and $\mathcal{R}$ denote a set of entities and relations, respectively. A knowledge graph $\mathcal{G} = \{(s, r, o)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is a directed graph consisting of a set of facts. Each fact is represented with a triple (s, r, o), denoting a directed edge pointing from the subject entity $s \in \mathcal{E}$ to the object entity $o \in \mathcal{E}$. $r \in \mathcal{R}$ is the edge type, i.e., the relation type, describing the relationship between s and o.*

Figure 2.1 shows an example knowledge graph (KG). Each node in a KG corresponds to an entity (e.g., *Ang Lee*). For each directed edge (s, r, o), s and o can also be viewed as the source node and the destination node, respectively. To accurately capture the relationship between two entities that cannot be described by a single relation type, KGs store multiple edges with different relation types between entities. Thus, KGs are considered as multi-relational graphs [128, 148]. Note that throughout Section 2.4, we adhere to the notation commonly used in KG research, where $\mathcal{E}$ denotes the set of entities, rather than the set of edges defined in prior works on general graphs (Section 2.1).

Definition 5 (Neighborhood in Knowledge Graphs)**.** *For an entity $e_i \in \mathcal{E}$ in the graph $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, its neighborhood $\mathcal{N}_{e_i}$ is defined as all (subject entity, relation type) pairs extracted from the facts where e_i serves as the object entity: $\mathcal{N}_{e_i} = \{(e_j, r_j) | (e_j, r_j, e_i) \in \mathcal{G}, e_j, e_i \in \mathcal{E}, r_j \in \mathcal{R}\}$.*

The idea of entity neighborhood in KGs is similar to node neighborhood in general graphs. A key difference is that KGs specify edge labels, i.e., relation types. Therefore, we

define each neighbor of a KG entity e_i as a pair consisting of a neighboring entity and its corresponding relation type associated with e_i.

2.4.2 Relational Learning on Knowledge Graphs

Relational learning on KGs is the process of learning patterns that encapsulate relational knowledge within KGs. KG relational learning methods can be categorized into symbolic and subsymbolic approaches. Symbolic approaches employ logic-based frameworks, such as description logic [6] and fuzzy logic [171], to capture the association among facts and extract logical rules for KG modeling. On the other hand, subsymbolic methods, also known as knowledge representation learning (KRL) approaches, learn low-dimensional embedding vectors, i.e., representations, of KG entities and relations. Embedding-based KRL has gained significant popularity in recent years and has become the dominant paradigm in KG relational learning. In this thesis, we focus on these embedding-based representation learning approaches and discuss the related ML problems corresponding to them.

2.4.3 Knowledge Representation Learning on Knowledge Graphs

In this section, we provide an overview of KRL methods on KGs. KRL refers to learning low-dimensional embedding vectors for KG entities and relations. Embeddings are learnable continuous vector representations. Within the context of ML and deep learning, they can be trained from scratch via backpropagation. In KRL, Each KG entity or relation is mapped to a unique embedding vector representation which is learnable through the training data.

Inverse Relations. It is worth noting that a common practice in KRL is to include inverse relations for each KG relation type. This involves expanding the original KGs by adding the facts containing inverse relations and then performing representation learning on these expanded graphs. For each fact (s, r, o), the corresponding inverse fact triple is represented as (o, r^{-1}, s), where r^{-1} denotes the inverse relation of r. For example, for the fact (*Oxford*, *located in*, *the United Kingdom*), its inverse fact triple is (*the United Kingdom*, *located in*$^{-1}$, *Oxford*), meaning that *the United Kingdom* has a region named *Oxford*. Inverse relations are included into the KG relation set. As a result, the size of the KG as well as the size of its relation set are doubled. Incorporating inverse relations enhances graph connectivity during representation learning and leads to better model performance

as well as theoretically higher expressiveness as shown in recent works [87, 76]

KG Score Functions

A majority of embedding-based KRL methods propose KG score functions to compute the plausibility score of each fact triple (s, r, o), indicating the likelihood of its veracity. A KG score function $\phi(\cdot)$ takes the fact (s, r, o) as input, finds the corresponding embedding vector representations $\mathbf{h}_s$, $\mathbf{h}_r$, $\mathbf{h}_o$ of s, r, o, and then outputs a real-valued number as the computed score. In the early stage, lots of works take the Euclidean space as the representation space of knowledge representations. For example:

- RESCAL [115] models a fact triple as a three-way tensor. It specifies a unique parameter matrix[9] $\mathbf{H}_r$ for each relation r and computes the score for the fact (s, r, o) as

$$\phi((s, r, o)) = \mathbf{h}_s{}^\top \mathbf{H}_r \mathbf{h}_o, \quad \mathbf{h}_s, \mathbf{h}_o \in \mathbb{R}^d, \mathbf{H}_r \in \mathbf{R}^{d \times d}. \tag{2.6}$$

  d is the dimension of entity representations. $\mathbf{H}_r$ is a full-rank matrix.

- TransE [13] takes relations as translations from the subject entities to the object entities in the Euclidean space, i.e.,

$$\phi((s, r, o)) = -\|\mathbf{h}_s + \mathbf{h}_r - \mathbf{h}_o\|, \quad \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o \in \mathbb{R}^d. \tag{2.7}$$

  d is the dimension of entity and relation representations. $\|\cdot\|$ denotes the norm (either L1 norm or L2 norm in TransE). After translation, the smaller the norm is, the greater the plausibility of the fact (s, r, o).

- DistMult [169] follows RESCAL, using a similar form of tensor factorization-based function to compute scores. It restricts each relation-specific parameter matrix $\mathbf{H}_r$ to be diagonal, making it possible to use a vector $\mathbf{h}_r$ to represent each relation. The complete form of DistMult is

$$\phi((s, r, o)) = \langle \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o \rangle, \quad \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o \in \mathbb{R}^d. \tag{2.8}$$

  d is the dimension of entity and relation representations. $\langle \cdot, \cdot, \cdot \rangle$ is a function that first computes the element-wise product of three input vectors, and then does a sum over all elements.

[9] We take the parameter matrix as a distinct form of embedding vector.

- TuckER [9] is another tensor factorization-based KG score function inspired by Tucker decomposition [146]. It specifies a learnable core tensor $\mathcal{W}$ and performs tensor product in three modes, i.e.,

$$\phi((s,r,o)) = \mathcal{W} \times_1 \mathbf{h}_s \times_2 \mathbf{h}_r \times_3 \mathbf{h}_o, \quad \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o \in \mathbb{R}^d, \mathcal{W} \in \mathbb{R}^{d \times d \times d}. \tag{2.9}$$

  d is the dimension of entity and relation representations. $\times_1$, $\times_2$ and $\times_3$ denote tensor product in three different modes.

To increase models' expressiveness, there also exist a number of methods learning knowledge representations in the complex space. For example:

- ComplEx [145] extends DistMult to the complex space

$$\phi((s,r,o)) = \mathrm{Re}\left(\langle \mathbf{h}_s, \mathbf{h}_r, \bar{\mathbf{h}}_o \rangle\right), \quad \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o \in \mathbb{C}^d. \tag{2.10}$$

  d is the dimension of entity and relation representations. Each element in $\mathbf{h}_s$, $\mathbf{h}_r$ and $\mathbf{h}_o$ is a complex number and $\mathrm{Re}(\cdot)$ denotes a function taking the real part of its input. $\bar{\mathbf{h}}_o$ is the complex conjugate of the object entity representation $\mathbf{h}_o$.

- RotatE [140] is a rotational model taking relation as a rotation from the subject entity to the object entity in the complex space

$$\phi((s,r,o)) = -\|\mathbf{h}_s \circ \mathbf{h}_r - \mathbf{h}_o\|, \quad \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o \in \mathbb{C}^d. \tag{2.11}$$

  d is the dimension of entity and relation representations. $\circ$ is the element-wise product. Each element in $\mathbf{h}_r$ has a modulus of 1, leading to a counterclockwise rotation around the origin of the complex plane only affecting the phases of entity representations.

- QuatE [176] extends RotatE's complex representations to hypercomplex ones consisting of quaternions [67]. A quaternion contains one real component and three imaginary components, therefore the quaternion space can be viewed as a hypercomplex space. The complete form of QuatE is

$$\phi((s,r,o)) = \mathbf{h}_s \otimes \frac{\mathbf{h}_r}{\|\mathbf{h}_r\|}\mathbf{h}_o, \quad \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o \in \mathbb{H}^d. \tag{2.12}$$

  d is the dimension of entity and relation representations. Each element in $\mathbf{h}_s$, $\mathbf{h}_r$ and $\mathbf{h}_o$ is a quaternion and $\otimes$ is the Hamilton product operation. $\|\cdot\|$ denotes the norm of its input.

In addition to the methods mentioned above, more recent works leverage the hyperbolic space for knowledge representation learning due to its advantage in representing hierarchical structures. For example:

- MuRP [8] embeds multi-relational graph data in the Poincaré ball model of the hyperbolic space. A Poincaré ball of radius $1/\sqrt{c}$ is a d-dimensional manifold $\mathbb{B}_c^d = \{x \in \mathbb{R}^d : c\|x\|^2 < 1\}$ equipped with the Riemannian metric $g^{\mathbb{B}}$. $g^{\mathbb{B}} = (2/(1 - c\|x\|^2))^2 g^{\mathbb{E}}$, where $g^{\mathbb{E}}$ is the Euclidean metric [114]. $\|\cdot\|$ is the Euclidean norm. The complete form of MuRP is defined as

$$\phi((s,r,o)) = -\text{dist}_{\mathbb{B}_c^d}\left(\exp_0^c\left(\mathbf{H}_r \log_0^c(\mathbf{h}_s)\right), \mathbf{h}_o \oplus_c \mathbf{h}_r\right)^2 + b_s + b_o. \tag{2.13}$$

  $\mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o \in \mathbb{B}_c^d$ are hyperbolic embeddings. $\mathbf{H}_r \in \mathbb{R}^{d\times d}$ is a diagonal parameter matrix specific to the relation r. $\log_0^c(\cdot)$ and $\exp_0^c(\cdot)$ are two operations projecting the subject entity representation to the tangent space of the Poincaré ball and performing back projection, respectively. $\oplus_c$ denotes Möbius addition [147] and b_s, b_o are entity-related biases. $\text{dist}_{\mathbb{B}_c^d}(x,y) = (2/\sqrt{c})\tanh^{-1}(\sqrt{c}\|-x \oplus_c y\|)$ represents the distance between $x \in \mathbb{B}_c^d$ and $y \in \mathbb{B}_c^d$.

- ATTH [20] introduces relation-specific hyperbolic rotation and reflection into the Poincaré ball model. Different from MuRP, ATTH also learns relation-specific hyperbolic curvatures c_r for different relations. The complete form of ATTH is defined as

$$\begin{aligned}
\phi((s,r,o)) = -\text{dist}_{\mathbb{B}_{c_r}^d}\left(\text{Att}\left(\mathbf{h}_s^{\text{Rot}}, \mathbf{h}_s^{\text{Ref}}; \mathbf{a}_r\right) \oplus_{c_r} \mathbf{h}_r, \mathbf{h}_o\right)^2 + b_s + b_o, \\
\text{where } \mathbf{h}_s^{\text{Rot}} = \mathbf{W}_r^{\text{Rot}}\mathbf{h}_s,\ \mathbf{h}_s^{\text{Ref}} = \mathbf{W}_r^{\text{Ref}}\mathbf{h}_s; \\
\text{Att}\left(\mathbf{h}_s^{\text{Rot}}, \mathbf{h}_s^{\text{Ref}}; \mathbf{a}_r\right) = \exp_0^{c_r}\left(\alpha_{\mathbf{h}_s^{\text{Rot}}} \log_0^{c_r}\left(\mathbf{h}_s^{\text{Rot}}\right) + \alpha_{\mathbf{h}_s^{\text{Ref}}} \log_0^{c_r}\left(\mathbf{h}_s^{\text{Ref}}\right)\right), \\
\left(\alpha_{\mathbf{h}_s^{\text{Rot}}}, \alpha_{\mathbf{h}_s^{\text{Ref}}}\right) = \text{Softmax}\left(\mathbf{a}^\top \log_0^{c_r}\left(\mathbf{h}_s^{\text{Rot}}\right), \mathbf{a}^\top \log_0^{c_r}\left(\mathbf{h}_s^{\text{Ref}}\right)\right).
\end{aligned} \tag{2.14}$$

  $\mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o \in \mathbb{B}_{c_r}^d$ are hyperbolic embeddings. $\mathbf{W}_r^{\text{Rot}} \in \mathbb{R}^{d\times d}$ and $\mathbf{W}_r^{\text{Ref}} \in \mathbb{R}^{d\times d}$ are the block-diagonal rotation and reflection matrices specific to the relation r, respectively. $\mathbf{a} \in \mathbb{R}^d$ is an attention vector. $\log_0^{c_r}(\cdot)$ and $\exp_0^{c_r}(\cdot)$ are two operations projecting the subject entity representation to the tangent space of the relation-specific Poincaré ball and performing back projection, respectively. $\oplus_{c_r}$ denotes Möbius addition and b_s, b_o are entity-related biases. $\text{dist}_{\mathbb{B}_{c_r}^d}(x,y) = (2/\sqrt{c_r})\tanh^{-1}(\sqrt{c_r}\|-x \oplus_{c_r} y\|)$ represents the distance between $x \in \mathbb{B}_{c_r}^d$ and $y \in \mathbb{B}_{c_r}^d$.

We have discussed a number of classic KG score functions in this section. For a more comprehensive investigation of other KG score functions, please refer to the following surveys [78, 19]. KG score functions often overlook the structural information inherent in the graph structure of KGs, e.g., node neighborhoods. To address this limitation, there has been a growing trend in designing graph encoders based on GNNs to learn contextualized knowledge representations.

Multi-Relational Graph Neural Networks

To accommodate to KGs, a number of works design GNN models for multi-relational graphs. Note that in KGs, each node corresponds to an entity, so learning KG entity representations is equivalent to learning node representations within the context of GNNs.

We give an introduction of some classic multi-relational GNN models. GNN models update the representations of all entities at once in each layer. We show how they update the entity representation $\mathbf{h}_{e_i}$ for the entity $e_i \in \mathcal{E}$:

- Relational Graph Convolutional Network (R-GCN) [128] extends GCN [89] to multi-relational data and learns entity representations based on entities' neighborhood. An R-GCN layer is defined as

$$\mathbf{h}_{e_i}^{l+1} = \sigma\left(\sum_{r_k}\sum_{e_j \in \mathcal{N}_{e_i,r_k}} \frac{1}{c(e_i, r_k)} \mathbf{W}_{r_k}^{l} \mathbf{h}_{e_j}^{l} + \mathbf{W}_0^{l} \mathbf{h}_{e_i}^{l}\right). \tag{2.15}$$

  $\mathcal{N}_{e_i,r_k} = \{e_j | (e_j, r_k, e_i) \in \mathcal{G}, e_j, e_i \in \mathcal{E}, r_k \in \mathcal{R}\}$ is the relation-specific neighborhood of the entity e_i corresponding to the relation r_k. Note that it is slightly different from Definition 5 since in R-GCN the whole neighborhood of e_i is split into several relation-specific neighborhoods. $\mathbf{W}_{r_k}^{l} \in \mathbb{R}^{d^{l+1} \times d^{l}}$ is a trainable weight matrix related to r_k in the l^{th} R-GCN layer. $\mathbf{W}_0^{l} \in \mathbb{R}^{d^{l+1} \times d^{l}}$ is another trainable weight matrix in the l^{th} R-GCN layer deciding how much information e_i preserves from itself during update. $c(e_i, r_k)$ is a normalization constant that can either be learned or set manually. $\sigma(\cdot)$ denotes the activation function.

- Structure-Aware Convolutional Network (SACN) [130] introduces a weighted graph convolutional network (WGCN) that assigns a simple relation-specific parameter to each relation type

$$\mathbf{h}_{e_i}^{l+1} = \sigma\left(\sum_{r_k}\sum_{e_j \in \mathcal{N}_{e_i,r_k}} \alpha_{r_k}^{l} \mathbf{W}^{l} \mathbf{h}_{e_j}^{l} + \mathbf{W}_0^{l} \mathbf{h}_{e_i}^{l}\right). \tag{2.16}$$

$\mathcal{N}_{e_i,r_k}$ follows the same definition as in R-GCN [128]. $\mathbf{W}^l, \mathbf{W}_0^l \in \mathbb{R}^{d^{l+1} \times d^l}$ are two trainable weight matrices in the l^{th} layer. $\alpha_{r_k}^l \in \mathbb{R}$ is the parameter specific to the relation r_k in the l^{th} layer. $\sigma(\cdot)$ denotes the activation function.

- Composition-Based Multi-relational Graph Convolutional Network (CompGCN) [148] jointly learns representations of KG entities and relations with composition functions. Each CompGCN layer is defined as

$$\begin{aligned} \mathbf{h}_{e_i}^{l+1} =& \sigma\left(\sum_{(e_j,r_j)\in\mathcal{N}_{e_i}} \mathbf{W}_{\lambda(r_j)}^l \psi\left(\mathbf{h}_{e_j}^l, \mathbf{h}_{r_j}^l\right)\right), \ \mathbf{h}_{r_j}^{l+1} = \mathbf{W}_{\text{rel}}^l \mathbf{h}_{r_j}^l, \\ \text{where } \mathbf{W}_{\lambda(r_j)}^l =& \begin{cases} \mathbf{W}_O \in \mathbb{R}^{d^{l+1}\times d^l}, & r \in \mathcal{R}_{\text{orig}} \\ \mathbf{W}_I \in \mathbb{R}^{d^{l+1}\times d^l}, & r \in \mathcal{R}_{\text{inv}} \\ \mathbf{W}_S \in \mathbb{R}^{d^{l+1}\times d^l}, & r = \text{self-loop}. \end{cases} \end{aligned} \tag{2.17}$$

  $\psi(\cdot,\cdot)$ is a composition function that can be in various forms such as multiplication or subtraction. $\mathbf{W}_{\lambda(r_j)}^l$ is a direction-specific weight matrix in the l^{th} layer, depending on whether r_j is an original relation ($\mathcal{R}_{\text{orig}}$), an inverse relation ($\mathcal{R}_{\text{inv}}$), or denoting self-loop (connecting the node itself). $\mathbf{W}_{\text{rel}}^l$ is a weight matrix for updating relation representations in the l^{th} CompGCN layer. $\sigma(\cdot)$ is the activation function. CompGCN subsumes R-GCN [128] and WGCN [130] and has become a popular architecture in KRL.

In practice, multi-relational GNNs are coupled with KG score functions, in order to compute the plausibility scores of KG fact triples. To be specific, the output of GNNs will serve as the input representations of KG score functions. KRL methods aim to learn expressive representations for KG modeling. We will then discuss how to leverage these learned representations for downstream ML tasks. We particularly focus on two tasks, i.e., link prediction (Section 2.4.4) and natural language question answering (Section 2.4.5). In Section 2.4.6, we further explore methods for conducting inductive learning to achieve inductive link prediction on KGs.

2.4.4 Link Prediction on Knowledge Graphs

Although mainstream KGs are large-scale, they suffer from incompleteness [111], meaning they usually do not contain all ground truth facts. Hence, predicting unobserved facts in KGs is crucial, leading to the task of link prediction on KGs, also known as KG completion (KGC). KGC is the most popular ML task in the field of KRL.

Definition 6 (Knowledge Graph Completion/Link Prediction)**.** *Assume we have a ground truth KG $\mathcal{G}_{gt}$ that contains all the true facts, and an observed KG $\mathcal{G}_{ob}$ containing all the observed facts, where $\mathcal{G}_{ob} \subset \mathcal{G}_{gt}$. Given a link prediction query $(s_q, r_q, ?)$ (or $(?, r_q, o_q)$) derived from a ground truth fact $(s_q, r_q, o_q) \in \mathcal{G}_{gt} \setminus \mathcal{G}_{ob}$, KG completion (i.e., KG link prediction) aims to predict the missing object o_q (or subject s_q) based on $\mathcal{G}_{ob}$.*

As discussed in Section 2.4.3, it is common to augment the original KG with the facts including inverse relations. By incorporating inverse relations, subject entity prediction can be transformed into object entity prediction. For example, the link prediction query $(?, r_q, o_q)$ can be rewritten as $(o_q, {r_q}^{-1}, ?)$. This allows KGC to be framed as an object entity prediction problem.

In our definition, we focus on predicting missing entities rather than relations, following most previous works that formulate KGC as an entity prediction problem. We treat relation prediction as a separate problem and concentrate exclusively on entity prediction in this thesis.

Evaluation

KGC is formed as a ranking task for evaluation. For example, assume we want to predict the missing object entity o_q from a link prediction query $(s_q, r_q, ?)$, a KG model is asked to compute a score for each triple in $\{(s_q, r_q, o') | o' \in \mathcal{E}\}$, where o' is called candidate entity and can be any entity within the KG entity set $\mathcal{E}$. The candidate entities are ranked according to the scores of their associated triples (e.g., if (s_q, r_q, o_q) has the highest score then o_q ranks top 1). Ideally, a model should assign the highest score to the triple (s_q, r_q, o_q) compared with other triples containing other candidate entities $o' \in \mathcal{E} \setminus \{o_q\}$. To measure the quality of ranking, two evaluation metrics are widely adopted, i.e., mean reciprocal rank (MRR) and Hits@k. MRR computes the mean of the reciprocal ranks of ground truth missing entities

$$\text{MRR} = \frac{1}{|\mathcal{Q}|} \sum_{q \in \mathcal{Q}} \frac{1}{\psi_q}. \tag{2.18}$$

$\mathcal{Q}$ denotes the set of all link prediction queries and q represents each link prediction query. ψ_q is the rank of the ground truth missing entity in each query. Hits@k denotes the proportion of the predicted links where ground truth missing entities are ranked as top k. Another point worth noting is that it has been a common practice to adopt a filtering setting (proposed in [13]) during ranking evaluation. Filtering refers to removing from the entity set all the candidate entities that form ground truth fact triples together with the

link prediction query during ranking, where these facts can appear either in the training, validation or test set (except the test triple of interest). For example, assume two facts exist in the KG: (*Shanghai*, *located in*, *China*), (*Shanghai*, *located in*, *Asia*). To predict the missing object entity of the link prediction query (*Shanghai*, *located in*, ?), derived from the fact (*Shanghai*, *located in*, *China*), the candidate entity *Asia* will be filtered because (*Shanghai*, *located in*, *Asia*) is also correct and *Asia* will impose negative influence on the ranking result, leading to an unfair evaluation of models.

2.4.5 Natural Language Question Answering on Knowledge Graphs

Natural language question answering on KGs (KGQA) can be defined as

Definition 7 (Question Answering on Knowledge Graphs)**.** *Assume we have an underlying KG* $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$*. Given a natural language question* q *and an annotated topic entity* $s_q \in \mathcal{E}$ *existing in* q*, KGQA aims to find the entity* $o_q \in \mathcal{E}$ *that answers* q*.*

Natural language questions in KGQA are proposed based on the facts in the underlying KG, meaning that all the KGQA questions are answerable if a model can perform perfect reasoning over the KG and no additional information source is needed.

Different from link prediction, KGQA does not provide the relation types for models. Models have to understand the natural language questions for answer inference. Another difference is that as introduced in a series of works such as [127], KGQA may involve multi-hop questions that require models to perform reasoning over multiple edges. For example, a 2-hop natural language question *Who is the other child of Christopher Hemsworth's father?* can be answered only by reasoning over two KG facts (*Craig Hemsworth*, *is father of*, *Christopher Hemsworth*) and (*Craig Hemsworth*, *is father of*, *Liam Hemsworth*)[10]. The answer entity *Liam Hemsworth* is a 2-hop neighbor of the topic entity *Christopher Hemsworth*. These differences mean that the approaches introduced in Section 2.4.3 cannot be directly used to solve KGQA.

Mainstream KGQA methods can be divided into two types, i.e, semantic parsing-based and KG embedding-based. Semantic parsing-based methods (e.g., [10, 101, 27, 139, 36]) parse the questions into logic forms, e.g., SPARQL query, that are executable within knowledge bases. Two major drawbacks of them are that: (1) they need to annotate expensive logic forms as supervision signal; (2) they fail to achieve accurate prediction when KGs

[10]In this example, we assume that there is no fact such as (*Liam Hemsworth*, *is sibling of*, *Christopher Hemsworth*) that makes *Liam Hemsworth* an 1-hop neighbor of *Christopher Hemsworth*.

miss important facts regarding the ground truth answers (i.e., they suffer from KG incompleteness, which is a common problem for real-world KGs) [175, 94]. KG embedding-based KGQA has recently gained popularity. One classic method is EmbedKGQA [127]. EmbedKGQA consists of three modules: (1) the KRL module (KG embedding module) trains entity representations over the fact triples in the underlying KG by using the ComplEx [145] score function; (2) the question representation module enables natural language understanding by leveraging an LM, i.e., RoBERTa [104], to encode each natural language question q into a low-dimensional question representation; (3) the answer selection module selects the predicted answer e_{ans} with a score function in the same form as ComplEx

$$e_{\text{ans}} = \text{argmax}_{o' \in \mathcal{E}} \text{Re}\left(\langle \mathbf{h}_s, \mathbf{h}_q, \bar{\mathbf{h}}_{o'} \rangle\right) \quad \mathbf{h}_s, \mathbf{h}_q, \mathbf{h}_{o'} \in \mathbb{C}^d. \tag{2.19}$$

where $\mathbf{h}_q$ is the question representation encoded by LM[11] and d is the dimension size. $o' \in \mathcal{E}$ is an candidate entity that is same as in KGC. EmbedKGQA does not constrain the candidate answer to be within the limited neighborhood of the topic entity, so it is suitable for Multi-hop KGQA. It is also the foundation of various recent works on temporal KGQA that aims to perform QA on temporal KGs, which will be discussed in Section 2.5.5.

Based on EmbedKGQA, many follow-up works are done aiming to improve KGQA performance by extracting more relevant information to the questions from the underlying KGs or implementing better complex query reasoning techniques. For example:

- LEGO [123] consists of a latent space executor and a query synthesizer. It starts the answer inference with a query originating from the topic entity and then iteratively synthesizes and executes the query in the embedding space, conditioned on the question representation encoded by an LM. A relation pruner is trained to prune the search space in the query synthesis process, improving LEGO's efficiency and performance.

- BeamQA [4] develops a sequence-to-sequence path generation module, and a beam search execution algorithm to search and rank candidate entity answers to the questions. The path generation module takes a question as input and generates a sequence of relations based on the BART [97] LM. The search execution algorithm leverages the representations of KG entities and relations as well as the ComplEx [145] KG score function for candidate ranking.

[11] LM-encoded representations are real-valued. Here $\mathbf{h}_q$ has already been transformed into complex-valued vectors where each element is a complex number, as implemented in ComplEx [145].

Evaluation

Same as KGC, KGQA can also be framed as a ranking task. For each natural language question q together with its annotated topic entity s_q, a KGQA model is asked to compute a score based on s_q, q and any candidate entity $o' \in \mathcal{E}$. An ideal KGQA model should assign the highest score for the ground truth answer entity o_q. In this sense, KGQA evaluation can be viewed as a slight variant from KGC evaluation, by changing the query relation to the natural language question. Therefore, MRR and Hits@k are also used to evaluate models on this task.

2.4.6 Inductive Representation Learning on Knowledge Graphs

Inductive learning on KGs refers to the ability of a model in making inferences about the entities and relations unseen in the training data. As KRL gains popularity, there is a growing interest in enhancing KRL approaches with inductive capabilities. KRL approaches require a substantial amount of training data to learn expressive representations of KG entities and relations, making them inherently limited in inductive learning. Meanwhile, real-world KGs are evolving, with new entities and relations constantly emerging. This highlights the importance of effectively managing these newly-emerged elements (entities and relations). Therefore, it is crucial to study how to equip KRL approaches with strong inductive power.

The rest of this section is mainly divided into three parts: the first focuses on inductive learning for unseen entities, the second on unseen relations, and the third discusses the recent studies in addressing both unseen entities and relations together. As pointed out in [23], past and current research on inductive learning often assesses models' inductive power by conducting KG link prediction involving unseen elements. In this thesis, we follow previous works by focusing on the problem of inductive KG link prediction and centering our discussions around it. We also give a brief discussion about symbolic approaches for KG inductive learning at the end of this section.

Inductive Learning on Knowledge Graph Entities

Inductive KG link prediction on KG entities can be split into three problem settings: semi-inductive (SI), fully-inductive (FI) and mixture of both (SI + FI) [2].

SI refers to predicting the unobserved facts containing one seen entity and an unseen entity, i.e., either the subject or object entity in the to-be-predicted fact is not seen in the

training data. We can define SI KG link prediction as

Definition 8 (Semi-Inductive Knowledge Graph Link Prediction)**.** *Given an observed background KG $\mathcal{G}_{back} \subseteq \mathcal{E}_{back} \times \mathcal{R} \times \mathcal{E}_{back}$ and a set of unseen entities $\mathcal{E}'$, where $\mathcal{E}' \cap \mathcal{E}_{back} = \emptyset$. Semi-inductive KG link prediction aims to predict the missing entity from each link prediction query $(s_q, r_q, ?)$ (or $(?, r_q, o_q)$) derived from an unobserved fact where either $s_q \in \mathcal{E}'$, $o_q \in \mathcal{E}_{back}$ or $s_q \in \mathcal{E}_{back}$, $o_q \in \mathcal{E}'$, and $r_q \in \mathcal{R}$.*

Representative works of SI link prediction leverage KGs' structural information, e.g., entity neighborhoods, to transfer knowledge from seen entities to unseen entities. They assume that new entities emerge together with a number of auxiliary facts connecting them to seen entities for computing inductive representations. SI Link prediction is then conducted on other facts (which do not exist in the auxiliary set) based on the computed unseen entity representations. We introduce three classic methods:

- MEAN [66] adopts GNNs to propagate information from the seen entities to unseen entities, based on the auxiliary facts. It also shows the effectiveness of mean pooling when GNNs are used in inductive entity representation learning.
- LAN [157] improves MEAN by introducing an attentional GNN for knowledge transfer. The attention weights are decided not only by the entity representations but also the logical rules indicating the relation dependencies between the relation in entity neighbors and the query relation r_q.
- CFAG [153] first leverages two GNN-based aggregators, i.e., global and local aggregators, to transfer information from seen entities to unseen entities, and then uses a conditional generative adversarial network [113] to incorporate the information of query relation r_q to output query-specific representations for unseen entities.

FI refers to predicting the unobserved facts containing two unseen entities, i.e., both subject and object entities in the to-be-predicted fact are not seen in the training data. We can define FI KG link prediction as

Definition 9 (Fully-Inductive Knowledge Graph Link Prediction)**.** *Given an observed background KG $\mathcal{G}_{back} \subseteq \mathcal{E}_{back} \times \mathcal{R} \times \mathcal{E}_{back}$ and a set of unseen entities $\mathcal{E}'$, where $\mathcal{E}' \cap \mathcal{E}_{back} = \emptyset$. Fully-inductive KG link prediction aims to predict the missing entity from each link prediction query $(s_q, r_q, ?)$ (or $(?, r_q, o_q)$) derived from an unobserved fact where $s_q, o_q \in \mathcal{E}'$ and $r_q \in \mathcal{R}$.*

FI KG link prediction assumes that unseen entities $\mathcal{E}'$ are not connected to seen entities $\mathcal{E}_{\text{back}}$ and form a completely new KG. Some works leverage structural information of KGs, e.g., the neighborhood of unseen entities, to achieve FI link prediction. For example:

- NodePiece [53] represents each unseen entity by encoding its distances to several randomly selected anchor entities and its relational context with neural networks. The anchor distances and the connected relations around the unseen entity are treated as the entity-independent information for inductive representations learning.

- MorsE [26] employs meta-learning [93], learning to learn the FI setting during model training. It uses an entity initializer and a GNN modulator to extract the entity type information as well as the relational contexts, which is called meta knowledge in [26].

Some other methods try to encode the subgraph between each pair of unseen entities for prediction. For example:

- GraIL [143] treats the unseen subject and object entities as a whole in an unobserved fact and encodes the relational subgraph (named as enclosing subgraph) between them. It extracts the subgraph on top of the multi-hop neighborhood of unseen entities, uses an attentional GNN to capture the contextualized information within the subgraph, and outputs a subgraph representation by average pooling over all the entities in the subgraph. The subgraph representation is then used in a score function for prediction.

A line of works follows [143] and proposes different subgraph encoding strategies for inductive learning, such as CoMPILE [107], TACT [21], and SRNI [166]. However, such kind of subgraph-based approaches have one limitation: they are limited to performing relation prediction, i.e., predicting the relation between two unseen entities. It is hard for them to be generalized to entity prediction since enclosing subgraphs can be extracted only when two entities are fixed, meaning that the subgraphs are strongly bounded to the entity pairs and to perform entity prediction, models have to extract a subgraph for each candidate entity, which introduces a heavy computational burden. Besides, some works, e.g., NBFNet [182], model the relational paths between unseen entities to achieve inductive learning. A relational path can be viewed as a simple subgraph [23]. Reasoning over paths requires no subgraph construction and therefore can improve model's efficiency [182]. Based on it, RED-GNN [179] proposes relational directed graphs that helps to preserve the local

neighborhood information between entity pairs and also benefit from the directional path for efficient subgraph encoding.

The mixture of both SI and FI, i.e., SI + FI, refers to predicting the unobserved facts containing two unseen entities/one unseen entity, i.e., both subject and object entities/either the subject or object entity in the to-be-predicted fact are/is not seen in the training data. We can define SI + FI KG link prediction as

Definition 10 (Semi-Inductive + Fully-Inductive Knowledge Graph Link Prediction)**.** *Given an observed background KG $\mathcal{G}_{back} \subseteq \mathcal{E}_{back} \times \mathcal{R} \times \mathcal{E}_{back}$ and a set of unseen entities $\mathcal{E}'$, where $\mathcal{E}' \cap \mathcal{E}_{back} = \emptyset$. SI + FI KG link prediction aims to predict the missing entity from each link prediction query $(s_q, r_q, ?)$ (or $(?, r_q, o_q)$) derived from an unobserved fact where either $s_q \in \mathcal{E}'$, $o_q \in \mathcal{E}_{back} \cup \mathcal{E}'$ or $s_q \in \mathcal{E}_{back} \cup \mathcal{E}'$, $o_q \in \mathcal{E}'$, and $r_q \in \mathcal{R}$.*

SI + FI KG link prediction is more realistic. It assumes that each unseen entity in $\mathcal{E}'$ can be either connected to a seen entity in $\mathcal{E}_{\text{back}}$ or an unseen entity in $\mathcal{E}'$, which is closer to the real-world scenarios when KGs expand. Similar to SI and FI methods, some SI + FI methods try to accurately encode structural information of unseen entities. For example:

- GEN [7] assumes that each unseen entity is associated with only a few facts as it emerges (unlike SI methods such as MEAN [66] that equip each unseen entity with a much larger number of associated auxiliary facts). It proposes a few-shot learning task, i.e., KG few-shot out-of-graph (OOG) link prediction, and designs a meta-learning framework to transfer knowledge from seen entities to unseen entities. KG few-shot OOG link prediction belongs to SI + FI link prediction and it constrains models to learning strong inductive representations for unseen entities with only few-shot data examples, which is challenging and more realistic.

Apart from them, a number of recent works have explored introducing additional information sources, e.g., textual descriptions of KG entities, for inductive entity representation learning. For example:

- QBLP [2] solves inductive KG link prediction by using the additional information provided by the qualifiers of hyper-relational KG facts (proposed in [54]). Each hyper-relational fact consists of a main fact triple and a group of qualifiers describing it. [2] assumes that qualifiers are comprised of seen entities and relations and thus they can help to transfer knowledge for inductive representation learning.

- KEPLER [159] proposes a KG dataset, i.e., Wikidata5M, that equips each entity with text descriptions. It fine-tunes a pre-trained LM on two objectives: KG link prediction (with the KG embedding loss adopted from RotatE [140]) and masked language modeling (with the loss adopted from BERT [38]). It leverages the additional textual information and LM's ability in natural language understanding to achieve inductive learning.

- SimKGC [156] follows KEPLER's setting and employs contrastive learning with different negative samplings to enhance text-based KG reasoning. It uses BERT to encode entities to enhance model's inductive capacity.

Additional information sources enable expressive representations of unseen entities, regardless of how many related facts are encountered during model training. If a model can effectively utilize these external sources, it can achieve strong inductive capabilities. Therefore, such methods are capable of solving SI + FI KG link prediction.

For a more comprehensive discussion about other inductive entity representation learning methods, please refer to the following surveys [23, 77]

Inductive Learning on Knowledge Graph Relations

Previous works on unseen relations typically consider the few-shot scenario, where each new relation is introduced with a small number of associated facts that models can use for inference. This problem is referred to as few-shot relational learning[12] [164], and we can formulate it into the following task

Definition 11 (Few-Shot Knowledge Graph Link Prediction)**.** *Given an observed background KG* $\mathcal{G}_{back} \subseteq \mathcal{E} \times \mathcal{R}_{back} \times \mathcal{E}$ *and a set of unseen relations* $\mathcal{R}'$*, where* $\mathcal{R}' \cap \mathcal{R}_{back} = \emptyset$*. Assume we further observe* K *quadruples* $\mathcal{S}_{r'} = \{(s_i, r', o_i)\}_{i=1}^{K}$ *corresponding to each unseen relation* r'*, where* $r' \in \mathcal{R}'$*,* $s_i, o_i \in \mathcal{E}$*. Based on* $\mathcal{S}_{r'}$ *and the whole background graph* $\mathcal{G}_{back}$*, few-shot KG link prediction aims to predict the missing entity of each link prediction query, i.e.,* $(s_q, r', ?)$ *or* $(?, r', o_q)$*, derived from the unobserved quadruple* $(s_q, r', o_q) \in \mathcal{Q}_{r'}$ *containing* r'*, where* $s_q, o_q \in \mathcal{E}$*.* $\mathcal{S}_{r'}$ *and* $\mathcal{Q}_{r'} = \{(s_i, r', o_i)\}_{i=K+1}^{M_{r'}}$ *are the support set and the query set for the unseen relation* r'*, respectively.* $M_{r'}$ *denotes the number of all the fact triples associated with* r'*.*

[12]Here we follow previous works (e.g., [164, 25]) and only refer to few-shot relations.

Few-shot KG link prediction is usually taken by previous works as a meta-learning problem. Here we discuss several classic approaches. Some works are metric-based meta-learning approaches that use metric functions to do similarity matching of the few-shot examples and the to-be-predicted links. For example:

- GMatching [164] is a model trained with episodic training [151]. In each episode, GMatching samples a relation as an unseen relation and simulates a one-shot learning problem focused on it. GMatching uses a GNN to learn contexualized entity representations based on the background graph and combines the representations of the subject and object entities in a KG fact. The representations of the support entity pair and the query entity pair are matched with a metric processor based on an LSTM network [73] for link inference.

- FSRL [172] improves GMatching by first using a relation-aware heterogeneous neighbor encoder, and then designing an aggregator based on RNNs [124] to combine the information provided by few-shot examples. It addresses GMatching's limitation in effectively learning from multiple support fact triples.

- FAAN [133] proposes an adaptive neighbor encoder to distinguish the importance of different neighbors of an entity based on their relatedness to the unseen relation in the link prediction query. It further uses Transformer [149] to encode support entity pairs and devises an adaptive matching processor that introduces attention between each support fact and the query fact during the support information aggregation.

Besides these methods, some other works employ the Model-Agnostic Meta-Learning (MAML) [50] framework for few-shot learning. For example:

- MetaR [25] uses a neural network to compute a relation meta based on each support fact triple of an unseen relation. It then utilizes the averaged relation meta as the initialization of the representation of the unseen relation. With this representation, it computes a loss with the TransE [13] score function. One step of gradient descent is applied to the relation meta to output the final unseen relation representation, which provides a good generalization power following MAML.

- GANA [116] proposes a gated and attentive neighbor aggregator (GNN-based) to capture the contextualized information of entities. It then uses a bidirectional LSTM to integrate all the information provided by the entity pairs in support facts. It

switches the TransE-like score function in MetaR to a score function based on TransH [160] to handle more complex relationships.

Apart from them, there also exist several works relying on additional information sources for inductive relation representation learning. These methods can also deal with the zero-shot setting, i.e., $\mathcal{S}_{r'} = \emptyset$. For example:

- ZSGAN [119] equips relations with textual descriptions. It trains a GAN [62] to generate unseen relation representations conditioned on their encoded textual descriptions, in order to achieve zero-shot relational learning.

- OntoZSL [59] leverages the ontology of KGs. It synthesizes the features of unseen relations by using the representations within an ontological schema including the semantics of the KG relations, which also enables zero-shot relational learning.

For a more comprehensive discussion about other inductive relation representation learning methods, please refer to the following surveys [23, 77]

Inductive Learning on Knowledge Graph Entities & Relations

Recently, some works study how to address both unseen entities and relations simultaneously. For example:

- MaKEr [24] employs meta-learning to jointly handle unseen entities and relations. It proposes a training task that simulates the situation where new entities and relations emerge together, in order to enable model's inductive capability[13]

- RMPI [60] leverages the subgraph encoding technique (used in FI methods, e.g., GraIL [143]) to handle unseen entities and achieves reasoning over unseen relations by passing information between the relations sharing common entities. The unseen relation representations can be further enhanced based on KGs' ontological schema (similar to OntoZSL [59]).

[13]Few-shot setting is not imposed in [24], meaning that each unseen relation and entity has a substantial number of support facts during evaluation (similar to the auxiliary facts in several previous works such as MEAN [66]). MaKEr leverages a GNN to aggregate information provided by the support set and is evaluated on the query set.

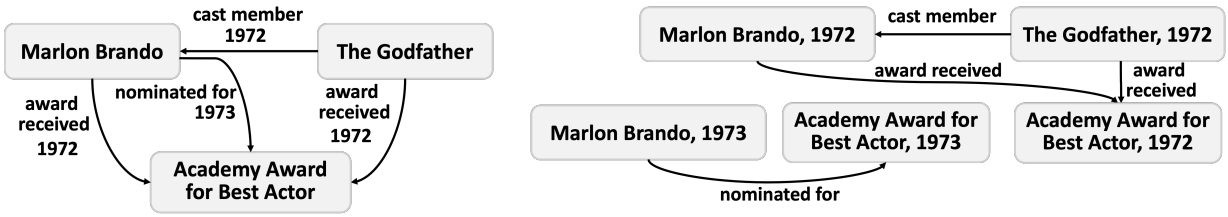
- ULTRA [55] builds upon RMPI by constructing a multi-relational meta-graph of fundamental relation interactions (tail-to-head, head-to-head, head-to-tail, and tail-to-tail[14]). It learns relative relation representations based on these fundamental interactions and leverages conditional message passing to enhance performance. It handles unseen entities with an encoder based on NBFNet [182].

Symbolic Approaches for Inductive Learning on Knowledge Graphs.

Different from KRL-based methods, symbolic KG relational learning approaches (such as AMIE [52], AMIE+ [51] and AnyBURL [109]) are naturally capable of inductive learning. They learn entity-agnostic symbolic rules to reason KGs so they can handle unseen entities. However, one drawback of them is that rules are strongly bounded to KG relation types, which makes them not generalizable to unseen relations. As this thesis lays emphasis on KRL-based approaches, we will not go into more details about symbolic approaches. Please refer to the following surveys for a better understanding [174, 23].

2.5 Temporal Knowledge Graphs

2.5.1 Fundamental of Temporal Knowledge Graphs



(a) (b)

Figure 2.2: An example TKG. Figure 2.2a represents the TKG by coupling timestamps with edges, while Figure 2.2b couples timestamps with entities to form temporal nodes. The facts in the example TKG are extracted from Wikidata [152].

[14]Fundamental relation interactions are between the nodes in a relation graph where each node in the graph corresponds to a KG relation type. Tail-to-head means that the entity connecting two nodes in the relation graph serves as the object entity of the first relation node and the subject entity of the second relation node. Similar meaning applies for other fundamental interactions.

Definition 12 (Temporal Knowledge Graph)**.** *Let $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ denote a set of entities, relations and timestamps, respectively. A temporal knowledge graph $\mathcal{G} = \{(s, r, o, t)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$ is a directed graph consisting of a set of temporal facts. Each fact is represented with a quadruple (s, r, o, t), denoting a directed edge pointing from the subject entity $s \in \mathcal{E}$ to the object entity $o \in \mathcal{E}$ at the timestamp t. $r \in \mathcal{R}$ is the edge type, i.e., the relation type, describing the relationship between s and o.*

Figure 2.2a shows an example TKG. Each node in a TKG corresponds to an entity. For each directed edge (s, r, o, t), s and o can be viewed as the source node and the destination node, respectively, and a timestamp t is coupled with the edge to specify the time validity of the fact. Alternatively, we can decouple timestamps from edges and couple them with entities to form temporal nodes. In this way, we can reframe the example TKG into the form presented in Figure 2.2b.

Some real-world TKGs, e.g., Wikidata [152], specify time constraints of facts with time periods. For example, $(s, r, o, [t_1, t_2])$ indicates that the fact remains valid from timestamp t_1 to t_2. In this thesis, we follow [84] and decompose each of such fact into a group of consecutive facts $\{(s, r, o, t_1), ..., (s, r, o, t_2)\}$ to represent its validity at all the timestamps between t_1 to t_2. This enables Definition 12 to represent any TKG, including the ones labeled with time periods.

A number of existing works denote a TKG as a sequence of TKG snapshots $\mathcal{G} = \{\mathcal{G}_1, ..., \mathcal{G}_T\}$, where $T = |\mathcal{T}|$ is the number of timestamps. Each snapshot contains all the concurrent facts happening at the same timestamp and can be viewed as a static KG. We will discuss how this formulation is used in GNN-based TKG representation learning methods in Section 2.5.3.

Definition 13 (Temporal Neighborhood in Temporal Knowledge Graphs)**.** *For an entity $e_i \in \mathcal{E}$ in the graph $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$, its neighborhood $\mathcal{N}_{e_i}$ is defined as all (subject entity, relation type, timestamp) triples extracted from the facts where e_i serves as the object entity: $\mathcal{N}_{e_i} = \{(e_j, r_j, t_j) | (e_j, r_j, e_i, t_j) \in \mathcal{G}, e_j, e_i \in \mathcal{E}, r_j \in \mathcal{R}, t_j \in \mathcal{T}\}$.*

Different from static KGs, when considering the temporal neighborhood of an entity e_i, TKGs further include the timestamps extracted from the facts associated with e_i.

2.5.2 Relational Learning on Temporal Knowledge Graphs

Relational learning on TKGs can also be categorized into symbolic and subsymbolic (i.e., KRL) approaches. Similar to the discussion about static KGs, we only focus on the

embedding-based representation learning approaches and discuss the related ML problems corresponding to them. Compared with KG relational learning methods, approaches for TKG reasoning lay greater emphasis on temporal reasoning. In many cases, temporal information is crucial for decision-making. For example, assume we have a natural language question for KGQA: *In which university did Albert Einstein start his study in 1901?*. And we have two observed related TKG facts, i.e., (*Albert Einstein*, *start study*, *University of Zurich*, *1901*), (*Albert Einstein*, *start study*, *ETH Zurich*, *1896*), in the underlying TKG. To answer the question, models should be able to reason over the underlying TKG and distinguish the ground truth answer *University of Zurich* from the negative entity *ETH Zurich*. Without the ability of temporal reasoning, static KG models (discussed in Section 2.4) are not able to achieve this. In contrast, TKG models are designed to incorporate temporal information, enabling more accurate decision-making that accounts for the temporal aspect.

2.5.3 Knowledge Representation Learning on Temporal Knowledge Graphs

In this section, we provide an overview of KRL methods on TKGs. Similar to KRL approaches for static KGs, these methods learn low-dimensional embedding vectors for entities and relations.

Inverse Relations. KRL methods for TKGs also include inverse relations for each relation type. This involves expanding the original TKGs by adding the facts containing inverse relations and then performing representation learning on these expanded graphs. For each fact (s, r, o, t), the corresponding inverse fact quadruple is (o, r^{-1}, s, t), where r^{-1} denotes the inverse relation of r. Similar to KRL on static KGs, inverse relations are included into the TKG relation set, leading to doubled TKG size and relation set size.

TKG Score Functions

A majority of embedding-based KRL methods are based on the KG score functions and develop TKG score functions to compute the plausibility score of each fact quadruple (s, r, o, t), indicating the likelihood of its veracity. A TKG score function $\phi(\cdot)$ takes the fact (s, r, o, t) as input, finds the corresponding embedding vector representations for s, r, o, t, and then outputs a real-valued number as the computed score. For example:

- TTransE [95] is based on TransE [13] and considers temporal information as a translation in the Euclidean space, i.e.,

$$\phi((s,r,o,t)) = -\|\mathbf{h}_s + \mathbf{h}_r + \mathbf{h}_t - \mathbf{h}_o\|, \quad \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o, \mathbf{h}_t \in \mathbb{R}^d. \tag{2.20}$$

  $\mathbf{h}_s$, $\mathbf{h}_r$, $\mathbf{h}_o$, $\mathbf{h}_t$ are the embedding vector representations of s, r, o, t, respectively. d is the dimension of entity, relation and time representations. $\|\cdot\|$ denotes the norm (same as TransE). After relational and temporal translation, the smaller the norm is, the greater the plausibility of the fact (s,r,o,t).

- TA-DistMult [58] builds upon DistMult. It decomposes the timestamp into a sequence consisting of self-defined temporal tokens and concatenates it with the relation token. Then the concatenated sequence is input into an LSTM to encode the time-aware relation representation. Finally, TA-DistMult uses DistMult score function to compute the final score

$$\phi((s,r,o,t)) = \langle \mathbf{h}_s, \mathbf{h}_{r,t}, \mathbf{h}_o \rangle, \quad \mathbf{h}_s, \mathbf{h}_{r,t}, \mathbf{h}_o \in \mathbb{R}^d. \tag{2.21}$$

  $\mathbf{h}_{r,t}$ is the time-aware relation representation of the relation r at timestamp t. d is the dimension of representations. $\langle \cdot, \cdot, \cdot \rangle$ is a function as same in DistMult.

- TNTComplEx [90] is inspired by ComplEx [145]. It assumes that some relations are affected by temporal aspects and some are not. To this end, TNTComplEx uses a combination of time-aware and time-invariant relation representations to embed TKG relations. The complete form of this method is defined as

$$\phi((s,r,o,t)) = \mathrm{Re}\left(\langle \mathbf{h}_s, \mathbf{h}_r + \mathbf{h}_{r,t} \circ \mathbf{h}_t, \bar{\mathbf{h}}_o \rangle\right), \quad \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_{r,t}, \mathbf{h}_o, \mathbf{h}_t \in \mathbb{C}^d. \tag{2.22}$$

  Here we use $\mathbf{h}_r$ to denote the time-invariant part of relation representation, and $\mathbf{h}_{r,t}$ is the time-aware part. d is the dimension of representations. $\mathrm{Re}(\cdot)$ denotes a function taking the real part of the its input. $\circ$ is the element-wise product operation.

- TeRo [145] implements temporal rotation based on the operation in RotatE [140] to jointly encode entity and time information. It then uses a TransE-like score function as the final form

$$\begin{aligned} &\phi((s,r,o,t)) = -\|\mathbf{h}_{s,t} + \mathbf{h}_r - \bar{\mathbf{h}}_{o,t}\|, \\ &\text{where } \mathbf{h}_{s,t} = \mathbf{h}_s \circ \mathbf{h}_t,\ \mathbf{h}_{o,t} = \mathbf{h}_o \circ \mathbf{h}_t;\ \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o, \mathbf{h}_t \in \mathbb{C}^d. \end{aligned} \tag{2.23}$$

$\mathbf{h}_s$, $\mathbf{h}_r$, $\mathbf{h}_o$, $\mathbf{h}_t$ are the embedding vector representations of s, r, o, t, respectively. d is the dimension of representations. Each element in $\mathbf{h}_s$, $\mathbf{h}_r$ and $\mathbf{h}_o$ is a complex number and $\mathrm{Re}(\cdot)$ denotes a function taking the real part of its input. $\bar{\mathbf{h}}_{o,t}$ is the complex conjugate of the time-aware object entity representation $\mathbf{h}_{o,t}$.

- ChronoR [125] is also inspired by RotateE. It treats the combination of relation and timestamp as a rotation from the subject entity to the object entity in the complex space. Following TNTComplEx, it uses two separate embedding vectors to model the time-aware and the time-invariant relation representations.

$$\phi((s,r,o,t)) = \mathrm{Re}\left(\langle \mathbf{h}_s \circ (\mathbf{h}_{r,t} \| \mathbf{h}_t) \circ \mathbf{h}_r, \mathbf{h}_o \rangle\right), \quad \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_{r,t}, \mathbf{h}_o, \mathbf{h}_t \in \mathbb{C}^d. \tag{2.24}$$

$\mathbf{h}_r$, $\mathbf{h}_{r,t}$ are the time-invariant, time-aware part of relation representation, respectively. d is the dimension of representations. $\mathrm{Re}(\cdot)$ denotes a function taking the real part of its input. $\circ$ is the element-wise product operation and $\|$ means concatenation.

- RotateQVS [22] extends TeRo and represents temporal information using rotations in the quaternion vector space. Temporal rotations happen within the imaginary axes of the quaternion representing each entity and the real part is used to describe the time-invariant part of entity representation. Compared with TeRo that performs rotation in the complex space, quaternions in RotateQVS have three imaginary axes and therefore are more expressive in modeling. The complete form of RotateQVS is defined as

$$\begin{aligned} &\phi((s,r,o,t)) = -\|\mathbf{h}_{s,t} + \mathbf{h}_r - \bar{\mathbf{h}}_{o,t}\|, \\ &\text{where } \mathbf{h}_{s,t} = \mathbf{h}_t \mathbf{h}_s {\mathbf{h}_t}^{-1},\ \mathbf{h}_{o,t} = \mathbf{h}_t \mathbf{h}_o {\mathbf{h}_t}^{-1},\ \mathbf{h}_s, \mathbf{h}_r, \mathbf{h}_o, \mathbf{h}_t \in \mathbb{H}^d. \end{aligned} \tag{2.25}$$

$\mathbf{h}_s$, $\mathbf{h}_r$, $\mathbf{h}_o$, $\mathbf{h}_t$ are the embedding vector representations of s, r, o, t, respectively. d is the dimension of representations. RotateQVS constrains the time representations $\mathbf{h}_t$ as unit quaternions.

- DyERNIE [69] roots from MuRP [8] and embeds TKG entities with time-aware entity representations on Riemannian manifolds. For an entity at a timestamp, DyERNIE uses a collection of N embedding vectors on N different manifolds. Each of these vectors is a combination of a time-invariant part and a velocity vector specifying the

time-dependent information. Its complete form is defined as

$$\begin{aligned}\phi((s,r,o,t)) = \sum_{i=1}^{N} - \mathrm{dist}_{\mathbb{B}_{c_i}^{d_i}}\left(\mathbf{H}_r^i \otimes_{c_i} \mathbf{h}_{s,t}^i, \mathbf{h}_{o,t}^i \oplus_{c_i} \mathbf{h}_r^i\right)^2 + b_s^i + b_o^i. \\ \text{where } \mathbf{h}_{s,t}^i = \exp_0^{c_i}\left(\log_0^{c_i}(\mathbf{h}_s^i) + \mathbf{h}_s^{\mathrm{vel},i} t\right), \\ \mathbf{h}_{o,t}^i = \exp_0^{c_i}\left(\log_0^{c_i}(\mathbf{h}_o^i) + \mathbf{h}_o^{\mathrm{vel},i} t\right).\end{aligned} \tag{2.26}$$

$\mathbf{h}_s^i, \mathbf{h}_r^i, \mathbf{h}_o^i \in \mathbb{B}_{c_i}^{d_i}$ are hyperbolic embeddings on the i^{th} manifold. $\mathbf{h}_s^{\mathrm{vel},i} \in \mathbb{B}_{c_i}^{d_i}$ and $\mathbf{h}_o^{\mathrm{vel},i} \in \mathbb{B}_{c_i}^{d_i}$ denote the velocity vectors of s and o on the i^{th} manifold, respectively. d_i and c_i are the representation dimension and the curvature of the i^{th} manifold, respectively. $\mathbf{H}_r^i \in \mathbb{R}^{d_i \times d_i}$ is a parameter matrix specific to the relation r on the i^{th} manifold. $\log_0^{c_i}(\cdot)$ and $\exp_0^{c_i}(\cdot)$ are two operations projecting representations to the tangent space of the manifold and performing back projection, respectively. $\otimes_{c_i}$ and $\oplus_{c_i}$ denote Möbius multiplication and addition [147, 56], respectively. b_s^i, b_o^i are entity-related biases on the i^{th} manifold. The distance function of DyERNIE $\mathrm{dist}_{\mathbb{B}_{c_i}^{d_i}}(\cdot,\cdot)$ is in the same form as in MuRP.

In addition to the classic methods mentioned above, there are a wide variety of other TKG score functions. Please refer to the following surveys [17, 155, 18] to have a better overview. Similar to the problem for score functions of static KGs, TKG score functions only pay attention to the fact quadruples and neglect the structural information in graphs. Therefore, recent TKG modeling approaches aim to enhance model capabilities by utilizing various GNN-based graph encoders, which will be discussed next.

Graph Neural Network-Based TKG Models

A number of GNN-based TKG modeling approaches apply GNNs in the following way: (1) first, the concurrent TKG facts happening at the same timestamp will be grouped together and taken as a static KG snapshot; (2) a GNN graph encoder (in most cases multi-relational GNN) is then used to encode each snapshot; (3) the encoded representations at different timestamps are input into a sequence encoder, e.g., LSTM or Transformer, to learn time-aware representations at the timestamp of interest[15]; (4) finally, a TKG decoder (normally a KG or TKG score function) is employed to leverage the learned representations

[15]In this thesis, timestamp of interest means the timestamp involved in model inference. For example, if we wish to compute the plausibility score of a TKG fact happening at timestamp t, then t is the timestamp of interest.

for performing downstream ML tasks. Here we discuss several classic methods using such framework:

- TeMP [161] employs R-GCN [128] as its snapshot encoder. It adopts two strategies for temporal encoding and develops two corresponding model variants, i.e., TeMP-GRU and TeMP-SA. TeMP-GRU encodes temporal information by using a gated recurrent unit (GRU) [32] to sequentially process the outputs of the snapshot encoder conditioned on different timestamps, while TeMP-SA utilizes the self-attention mechanism proposed in [149] to integrate time-aware entity representations with learnable attentions to different timestamps. Both model variants consider a time window around the timestamp of interest as the information pool for extracting temporal information. The graph information outside the time window is discarded. Any KG score function can be used as the decoder of TeMP, where time-aware entity representations serve as the input to enable temporal reasoning. TeMP's snapshot encoder can be any multi-relational GNN, making it an important prototype for TKG representation learning.

- RE-GCN [100] designs a relation-aware GNN for snapshot encoding. The l^{th} layer encoding the graph snapshot at the timestamp t_i can be written as

$$\mathbf{h}_{e_i,t_i}^{l+1} = \sigma\left(\sum_{(e_j,r_j)\in\mathcal{N}_{e_i,t_i}} \frac{1}{c(e_i,t_i)}\mathbf{W}_1^l(\mathbf{h}_{e_j,t_i}^l + \mathbf{h}_{r_j}^l) + \mathbf{W}_2^l\mathbf{h}_{e_i,t_i}^l\right). \tag{2.27}$$

  Here, $\mathcal{N}_{e_i,t_i} = \{(e_j,r_j)|(e_j,r_j,e_i,t_i)\in\mathcal{G}, e_j,e_i\in\mathcal{E}, r_j\in\mathcal{R}, t_i\in\mathcal{T}\}$ denotes the neighborhood of e_i in the snapshot at timestamp t_i. $\mathbf{W}_1^l, \mathbf{W}_2^l \in \mathbb{R}^{d^{l+1}\times d^l}$ are two trainable weight matrices in the l^{th} layer. $c(e_i,t_i)$ denotes the number of facts happening at t_i that take e_i as the object entity. $\sigma(\cdot)$ is the activation function. Note that although we write the representations with timestamp subscripts, the GNN here does not inject temporal information evolving along the time axis. It is because this GNN encoder is applied on each graph snapshot which is a static KG. We use this style of notation to indicate that the GNN incorporates the structural information of entities at each specific timestamp. Based on the output of GNN at two neighboring timestamps, RE-GCN uses a time gate recurrent component to compute a weighted sum of entity representations to capture the temporal evolution of entities. For representing each relation in a time-aware manner, RE-GCN first performs mean pooling over the representations of the entities connecting to this relation at each timestamp, and then

uses a GRU to capture temporal dynamics. Besides, RE-GCN models time-invariant information of entities and relations provided by the TKGs originating from the Integrated Crisis Early Warning System (ICEWS) knowledge base. In ICEWS-based TKGs, each entity's name string contains time-invariant property information. For example, the entity *Police (Australia)* implies that this entity belongs to Australia and the general concept of Police. RE-GCN constructs two time-invariant facts (in our example, the constructed facts are (*Police (Australia)*, *is a*, *Police*) and (*Police (Australia)*, *country*, *Australia*)) according to such entities and collect all of them to form a time-invariant multi-relational graph (can also be viewed as a static KG). It uses R-GCN to model this entity property graph and ensures that the time-aware representation of an entity remains closely aligned with its representation learned from the property graph. For decoding, RE-GCN uses the ConvTransE [130] score function.

- TANGO [71] designs a customized residual multi-relational graph convolutional layer to encode the structural information at each graph snapshot

$$\mathbf{h}_{e_i,t_i}^{l+1} = \mathbf{h}_{e_i,t_i}^{l} + \delta\sigma\left(\sum_{(e_j,r_j)\in\mathcal{N}_{e_i,t_i}} \frac{1}{c(e_i,t_i)}\mathbf{W}^l(\mathbf{h}_{e_j,t_i}^{l} \circ \mathbf{h}_{r_j}^{l})\right). \tag{2.28}$$

$\mathbf{W}^l \in \mathbb{R}^{d^{l+1}\times d^l}$ is a trainable weight matrix in the l^{th} layer. δ is a learnable weight deciding how much aggregated information is integrated in each layer. Other notations follow RE-GCN's graph encoder in Eq. 2.27. $\circ$ is the element-wise product operation. To better capture the temporal transition of TKGs, TANGO models the formation and dissolution of temporal edges by using another graph encoder. Each layer of this encoder is defined as

$$\bar{\mathbf{h}}_{e_i,t_i}^{l+1} = \sigma\left(\sum_{(e_j,r_j)\in\mathcal{N}_{e_i,t_i}^{\bar{A}}} \frac{1}{|\mathcal{N}_{e_i,t_i}^{\bar{A}}|}\mathbf{W}_{\text{trans}}^{l}\bar{A}_{e_j,r_j,e_i}^{t_i}(\mathbf{h}_{e_j,t_i}^{l} \circ \mathbf{h}_{r_j}^{l})\right). \tag{2.29}$$

$\mathbf{W}_{\text{trans}}^{l} \in \mathbb{R}^{d^{l+1}\times d^l}$ is a trainable weight matrix. $\bar{A}_{e_j,r_j,e_i} \in \{-1,0,1\}$ is an entry of a transition tensor $\bar{\mathbf{A}}_{t_i} \in \{-1,0,1\}^{|\mathcal{E}|\times|\mathcal{V}|\times|\mathcal{E}|}$ corresponding to e_j, r_j and e_i. $\bar{\mathbf{A}}_{t_i}$ is computed with the three-way adjacency tensors $\mathbf{A}_{t_i} \in \{0,1\}^{|\mathcal{E}|\times|\mathcal{V}|\times|\mathcal{E}|}$ and $\mathbf{A}_{t_i-\Delta t} \in \{0,1\}^{|\mathcal{E}|\times|\mathcal{V}|\times|\mathcal{E}|}$ from the timestamps t_i and $t_i-\Delta t$, i.e., $\bar{\mathbf{A}}_{t_i} = \mathbf{A}_{t_i} - \mathbf{A}_{t_i-\Delta t}$. Take $\mathbf{A}_{t_i}$ as an example, if (e_j, r_j, e_i, t_i) exists in the graph snapshot at t_i, then $A_{e_j,r_j,e_i}^{t_i} = 1$, otherwise $A_{e_j,r_j,e_i}^{t_i} = 0$. Finally, TANGO models temporal evolution recurrently with

a neural ordinary differential equation (NODE)

$$\mathbf{H}_{t_i+\Delta t} - \mathbf{H}_{t_i} = \int_{t_i}^{t_i+\Delta t} \Big(f_{\text{MGCN}}(\mathbf{H}_\tau, \mathcal{G}_\tau, \tau) + \omega f_{\text{trans}}(\mathbf{H}_\tau, \bar{\mathbf{A}}_\tau, \tau) \Big)\, d\tau. \tag{2.30}$$

$f_{\text{MGCN}}(\cdot,\cdot,\cdot)$ and $f_{\text{trans}}(\cdot,\cdot,\cdot)$ are two functions doing multi-relational graph convolution (Eq. 2.28) and temporal transition modeling (Eq. 2.29), respectively. $\mathbf{H}_\tau \in \mathbb{R}^{(|\mathcal{E}|+2|\mathcal{R}|)\times d}$ denotes the entity and relation (including inverse relations) representations at time τ. $\mathcal{G}_\tau$ is the graph snapshot at τ. ω is a hyperparameter controlling how much the model learns from edge formation and dissolution. The integration is modulated by an ODE solver and the time difference (Δt) decides how deep is the neural functions applied between a pair of neighboring timestamps, making TANGO able to model continuous temporal information. For decoding, TANGO uses DistMult and TuckER and builds two model variants TANGO-DistMult and TANGO-TuckER based on them.

- TiRGN [99] builds upon RE-GCN. It jointly computes the entity and relation representations based on the graph snapshot at timestamp t_i with the following GNN graph encoder

$$\mathbf{h}^{l+1}_{e_i,t_i} = \sigma\left(\sum_{(e_j,r_j)\in\mathcal{N}_{e_i,t_i}} \frac{1}{c(e_i,t_i)} \mathbf{W}^l_{r_j} \psi(\mathbf{h}^l_{e_j,t_i}, \mathbf{h}^l_{r_j,t_i}) + \mathbf{W}^l \mathbf{h}^l_{e_i,t_i} \right). \tag{2.31}$$

$\mathbf{W}^l_{r_j} \in \mathbb{R}^{d^{l+1}\times d^l}$ and $\mathbf{W}^l \in \mathbb{R}^{d^{l+1}\times d^l}$ are two trainable weight matrices and $\mathbf{W}^l_{r_j}$ is specific to the relation r_j. $\psi(\cdot,\cdot)$ is a function that performs the one-dimensional convolution. Other notations follow RE-GCN's graph encoder in Eq. 2.27. TiRGN uses an entity-oriented GRU and a relation-oriented GRU (named as local recurrent encoder jointly) to learn the temporal evolution patterns of entities and relations, respectively, in a similar way as RE-GCN. It then develops a new TKG score function Time-ConvTransE as a decoder, where periodical and non-periodical time representations are specified to enable more fine-grained temporal reasoning. To effectively capture the influence of repetitive global facts, TiRGN designs a global history encoder that checks the existence of facts in the history and performs masking on the link prediction[16] candidate entities. With the masked entity set, TiRGN uses Time-ConvTransE to output a global score, which is used to compute a weighted sum with the local score from the output of the decoder before masking.

[16]We will introduce link prediction on TKGs in Section 2.5.4, including formal definitions and further discussions.

Another line of works develops temporal GNNs that directly incorporate temporal information and temporal evolution during the graph information aggregation process. For example:

- TARGCN [44] explores the temporal context of each entity to learn its entity representation and encodes temporal information by learning representations based on time differences between the timestamp of interest and the timestamps of entity's temporal neighbors

$$\mathbf{h}_{e_i,t_i} = \sigma\left(\sum_{(e_j,r_j,t_j)\in\bar{\mathcal{N}}_{e_i}} \frac{1}{|\bar{\mathcal{N}}_{e_i}|}\mathbf{W} f\left(\mathbf{h}_{e_j}\|\psi(t_i,t_j)\|\mathbf{h}_{r_j}\right)\right). \tag{2.32}$$

  $\psi(t_i,t_j) = \sqrt{\frac{1}{d_t}}[cos(\omega_1(t_i-t_j)+\phi_1),...,cos(\omega_{d_t}(t_i-t_j)+\phi_{d_t}))]$ is a time difference encoder taken from [165], where d_t is the dimension of time representations. $\|$ means concatenation operation. $f(\cdot)$ is a feed-forward neural network. $\bar{\mathcal{N}}_{e_i} \subseteq \mathcal{N}_{e_i}$ is the sampled version of the complete temporal neighborhood $\mathcal{N}_{e_i}$. The probability of each neighbor being sampled is $\exp(-|t_i-t_j|)/\sum_{(e_k,r_k,t_k)\in\mathcal{N}_{e_i}} exp(-|t_i-t_k|)$. Sampling neighborhood helps TARGCN to focus on the temporal context near the timestamp of interest and avoid paying too much attention to the redundant temporal information. TARGCN has high parameter efficiency by achieving strong performance with much fewer parameters compared with TeMP [161] and T-GAP [86]. It uses DistMult as its decoder.

- xERTE [70] proposes a temporal GNN based on the temporal relational attention mechanism to solve link prediction on TKGs. Given a link prediction query $(s_q, r_q, ?, t_q)$[17], xERTE iteratively expands an inference graph for L times centered on the temporal node (s_q, t_q). In each iteration, xERTE extracts the prior temporal neighbors from the temporal nodes in the inference graph and forms edges between each temporal node and its sampled neighbors. Each extracted prior neighbor is taken from a historical fact that connects the entity of the temporal node and happens before the timestamp of the temporal node. After the construction of inference graph, xERTE develops a TRGA GNN-based layer that updates node representations as follows. Take node v as an example. xERTE computes the attention score associated with each edge from temporal node $u = (e_u, t_u)$ to $v = (e_v, t_v)$ in the

[17]Here, t_q is the timestamp of interest, similar to t_i in the introduction of previous methods.

inference graph as

$$\alpha_{v,u}^{l+1}(k) = \frac{\exp(\mathbf{a}_{v,u}^{l+1}(k))}{\sum_{w \in \hat{\mathcal{N}}_v} \sum_{z=1}^{K} \exp(\mathbf{a}_{v,w}^{l+1}(z))}, \\ \text{where } \mathbf{a}_{v,u}^{l+1}(k) = \mathbf{W}_{\text{sub}}^{l} \left(\mathbf{h}_v^l \| \mathbf{h}_{r_{v,u}^k}^l \| \mathbf{h}_{s_q}^l \| \mathbf{h}_{r_q}^l \right) \mathbf{W}_{\text{obj}}^{l} \left(\mathbf{h}_u^l \| \mathbf{h}_{r_{v,u}^k}^l \| \mathbf{h}_{s_q}^l \| \mathbf{h}_{r_q}^l \right). \tag{2.33}$$

$\hat{\mathcal{N}}_v$ is all the prior neighbors of temporal node v. $\mathbf{W}_{\text{sub}}^{l} \in \mathbb{R}^{d^{l+1} \times d^l}$ and $\mathbf{W}_{\text{obj}}^{l} \in \mathbb{R}^{d^{l+1} \times d^l}$ are two trainable weight matrices. K is the number of all relations between the entities of v and u during the inference graph construction. The node representation of v used in TRGA is initialized as $\mathbf{h}_v^0 = \mathbf{h}_{e_v} \| \mathbf{h}_{t_v}$, which also applies for node u. $\mathbf{h}_{e_v} \in \mathbb{R}^{d_e}$ and $\mathbf{h}_{t_v} \in \mathbb{R}^{d_t}$ denote the entity and timestamp representations of e_v and t_v, respectively. With the attention scores, xERTE aggregates information for v

$$\mathbf{h}_v^{l+1} = \sigma \left(\mathbf{W}^l \left(\gamma \mathbf{h}_v^l + (1-\gamma) \sum_{u \in \hat{\mathcal{N}}_v} \sum_{k=1}^{K} \alpha_{v,u}^{l+1}(k) \mathbf{h}_u^l + \mathbf{b}^l \right) \right). \tag{2.34}$$

$\mathbf{W}^l \in \mathbb{R}^{d^{l+1} \times d^l}$ is a weight matrix and $\mathbf{b}^l \in \mathbb{R}^{d^l}$ is a trainable weight vector. γ is a hyperparameter. After L layers of TRGA, xERTE aggregates the attention scores of the edges to generate temporal node attention scores and further computes entity attention scores by summing over all the nodes containing them. The link prediction results are decided by these entity scores, with higher scores leading to higher rankings for the corresponding entities

For the discussion of other TKG modeling methods based on temporal GNNs, please refer to the following surveys [17, 155, 18].

Other Methods

Apart from the two types of mainstream methods mentioned above, there are other approaches for KRL on TKGs. We discuss several representatives here.

- CyGNet [180] proposes a copy-generation framework to solve link prediction on TKGs. To predict a fact, CyGNet first operates in the copy mode, calculating probabilities for the object entities that have previously appeared with the link prediction query's subject and relation in the graph history. It then switches to the generation mode, computing probabilities for all candidate entities in the complete entity set. Finally, CyGNet combines the probability distributions from both modes to determine the final scores for fact inference. The probability generation process utilizes

multi-layer perceptrons (MLPs) on the embedding vector representations of entities, relations, and timestamps.

- TITer [138] is a reinforcement learning-based method for TKG link prediction. It searches for the ground truth missing entity with an agent traversing in the historical graph prior to the prediction timestamp. During graph traversal, the agent travels among the temporal nodes (defined same as in xERTE, i.e., (entity, timestamp) pairs). Every step of transition refers to travelling from one temporal node $u = (e_u, t_u)$ to another temporal node $v = (e_v, t_v)$ along the temporal edge between u and v where v is a prior neighbor of u defined same as in the inference graph of xERTE. Given a link prediction query $(s_q, r_q, ?, t_q)$, the agent starts traversal from (s_q, t_q) and do L steps of state transition. The entity of the node where the agent lands ultimately is taken as the prediction answer. TITer's agent follows a policy based on modeling the transition probabilities at each traversal step. Transition probabilities are computed using MLPs on the embedding vector representations of entities, relations and time differences, along with the path information encoded by an LSTM.

- ECOLA [72] enhances temporal knowledge embedding with temporally relevant textual information. For each fact in a TKG, ECOLA retrieves its relevant textual description from backend knowledge bases and concatenates it with the fact to create a textual input sequence. This sequence is then fed into a BERT [38] model for fine-tuning. The BERT model is trained on TKG facts using the knowledge-text prediction (KTP) task, a modified version of MLM. Given a fact quadruple and its corresponding textual description, KTP randomly masks some input tokens and asks the model to predict the original tokens based on their context. Unlike MLM, KTP specifically masks entities and relations to better align knowledge and text representations. In addition to KTP, ECOLA optimizes knowledge representations using a TKG score function. By combining these two training objectives, ECOLA effectively incorporates textual information into knowledge representations, enhancing the performance of previous TKG scoring functions.

- CENET [167] is a TKG link prediction method that leverages historical contrastive learning. It accounts for the impact of both historical repetitive facts and non-historical facts. A contrastive learning framework is used to learn representations that differentiate between historical and non-historical facts. During inference, CENET

first determines whether the to-be-predicted facts are repeated and then computes a probability distribution over a masked candidate entity set for link prediction.

- GenTKG [102] fine-tunes an LLM, i.e., Llama2-7B [144] for TKG link prediction. Given a link prediction query, GenTKG retrieves related historical facts with a group of learned temporal logical rules. An instruction text is augmented with the retrieval result to prompt the LLM to generate the prediction answer. The reasoning power of LLM is leveraged for answer inference.

Please refer to the following surveys [17, 155, 18] for the introduction of more types of KRL approaches on TKGs. In the next sections, we will discuss how to leverage the learned TKG representations for downstream ML tasks. Same as for static KGs, we particularly focus on two tasks on TKGs, i.e., link prediction (Section 2.5.4) and natural language question answering (Section 2.5.5).

2.5.4 Link Prediction on Temporal Knowledge Graphs

Similar to static KGs, TKGs also suffer from incompleteness. Therefore, it is also crucial to predict unobserved facts in TKGs, i.e., TKG link prediction. TKG link prediction can be categorized into two types: interpolated link prediction and extrapolated link prediction. TKG interpolated link prediction is commonly referred to as TKG completion or TKG interpolation, while TKG extrapolated link prediction is often called TKG forecasting or TKG extrapolation. The definitions of both tasks are provided below.

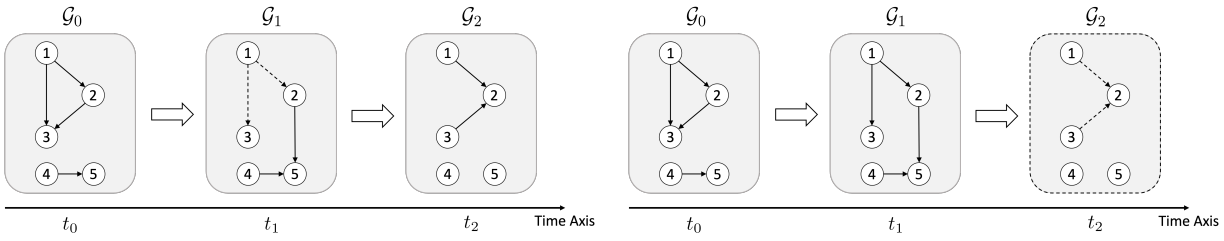
Definition 14 (Temporal Knowledge Graph Interpolated Link Prediction)**.** *Assume we have a ground truth TKG $\mathcal{G}_{gt}$ that contains all the true facts, and an observed TKG $\mathcal{G}_{ob}$ containing all the observed facts, where $\mathcal{G}_{ob} \subset \mathcal{G}_{gt}$. Given a link prediction query $(s_q, r_q, ?, t_q)$ (or $(?, r_q, o_q, t_q)$) derived from a ground truth fact $(s_q, r_q, o_q, t_q) \in \mathcal{G}_{gt} \backslash \mathcal{G}_{ob}$, TKG interpolated link prediction aims to predict the missing object o_q (or subject s_q) based on all the observed graph information in $\mathcal{G}_{ob}$.*

To predict a link in TKG interpolation, models can leverage the observed facts happening at any timestamp. This allows fact inference based on the evidence from the prediction timestamp t_q or the future.

Definition 15 (Temporal Knowledge Graph Extrapolated Link Prediction)**.** *Assume we have a ground truth TKG $\mathcal{G}_{gt}$ that contains all the true facts. Given a link prediction query*

$(s_q, r_q, ?, t_q)$ (or $(?, r_q, o_q, t_q)$) derived from a ground truth fact $(s_q, r_q, o_q, t_q) \in \mathcal{G}_{gt}$, TKG extrapolated link prediction aims to predict the missing object o_q (or subject s_q) based on the ground truth graph information prior to t_q, i.e., $\{(s_i, r_i, o_i, t_i) \in \mathcal{G}_{gt} | t_i < t_q\}$.

TKG extrapolation restricts models to only leveraging the facts happening before the prediction timestamp t_q for link inference. This setting simulates a forecasting scenario, which is important in various applications that require future planning. The difference between interpolated and extrapolated link prediction is illustrated with Figure 2.3.

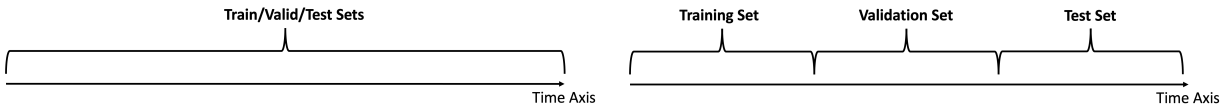


(a) TKG interpolation. (b) TKG extrapolation.

Figure 2.3: An example explaining the difference between TKG interpolation (Figure 2.3a) and extrapolation (Figure 2.3b). We depict the TKG as a series of snapshots. The edges marked with dashed lines are to-be-predicted links and the ones marked with solid lines are observed facts. The graph nodes are labeled with node IDs. In our example, TKG interpolation asks models to predict the unobserved links in $\mathcal{G}_1$, given the observed facts along the whole time axis. By contrast, TKG extrapolation asks models to predict all the links in $\mathcal{G}_2$, given prior facts in $\mathcal{G}_0$ and $\mathcal{G}_1$.

As discussed in Section 2.5.3, it is common to augment the original TKG with the facts including inverse relations. Similar to static KGs, each subject prediction query $(?, r_q, o_q, t_q)$ on TKGs can be rewritten as $(o_q, {r_q}^{-1}, ?, t_q)$. This allows both TKG interpolated and extrapolated link prediction to be framed as object entity prediction problems. We also focus only on predicting missing entities rather than relations. Relation and time prediction are treated as separate problems and this thesis concentrates exclusively on entity prediction, following most previous works on TKG link prediction.

Discussion. TKG interpolation methods typically differ from extrapolation methods in model design. Besides, their training strategies vary as well. Extrapolation methods follow the restriction of the forecasting setting and therefore design their models to only utilize



(a) TKG interpolation. (b) TKG extrapolation.

Figure 2.4: Difference between TKG interpolation (Figure 2.4a) and extrapolation (Figure 2.4b) in the temporal orders of train/valid/test sets.

the information prior to the link prediction queries for representation learning, while interpolation methods can have access to any information stored in the observed TKG facts for link inference. In the standard ML pipeline, models are first trained on a training dataset, validated using a validation set to assess their performance, and then evaluated on a test set. Due to the constraint in the forecasting scenario, the temporal order among train/valid/test sets for interpolated and extrapolated link prediction are different. For extrapolation, the maximum timestamp of the fact quadruples in the training set is smaller than the minimum timestamp of the facts in the validation set, and the maximum timestamp of the facts in the validation set is smaller than the minimum timestamp of the facts in the test set. For interpolation, all train/valid/test sets share the same time period. Figure 2.4 illustrates the difference of temporal orders among the train/valid/test sets between TKG interpolation and extrapolation. The varying temporal orders result in different training strategies for the methods addressing two tasks. Interpolation methods are trained with the facts that share the same time period with test data, while extrapolation methods need to generalize to the timestamps unseen in the training process during evaluation. To enable models' forecasting capabilities, TKG extrapolation methods simulate the forecasting process during training. For example, as TKG extrapolation methods, RE-GCN and TANGO (discussed in section 2.5.3) recurrently encode the graph information along the time axis from the past to the prediction timestamp to model the temporal dynamics for forecasting. Such simulation is proven effective and widely adopted in various recent works, e.g., [99, 103]. Another feasible practice to achieve forecasting is to introduce time difference representations. Although extrapolation models cannot learn expressive representations of the timestamps in the test set, they can learn the representations of time differences instead. Some extrapolation models combine time difference representation learning with GNNs in order to generate expressive time-aware entity representations at unseen timestamps during evaluation, e.g., [42]. Among all the methods discussed in Section 2.5.3, all the TKG score functions are interpolation methods since

they do not consider the forecasting setting during model design. For the rest of them, RE-GCN, TANGO, TiRGN, xERTE, CyGNet, TITer and CENET are developed for TKG extrapolation, while TeMP and TARGCN are designed for TKG interpolation. In general, TKG extrapolation is harder than interpolation because of the restricted amount of available information for link inference. Therefore, in recent years, TKG extrapolation has gained more popularity than interpolation, with increasing efforts dedicated to this topic.

Evaluation

Both TKG interpolation and extrapolation are formed as ranking tasks for evaluation. They share the same evaluation protocol. For example, assume we want to predict the missing object entity o_q from a link prediction query $(s_q, r_q, ?, t_q)$, a TKG model is asked to compute a score for each quadruple in $\{(s_q, r_q, o', t_q)|o' \in \mathcal{E}\}$, where o' is a candidate entity and can be any entity within the TKG entity set $\mathcal{E}$. The candidate entities are ranked according to the scores of their associated quadruples (e.g., if (s_q, r_q, o_q, t_q) has the highest score then o_q ranks top 1). To measure the quality of ranking, two evaluation metrics, i.e., MRR and Hits@k, introduced in Section 2.4.4 are widely adopted. Please refer to Section 2.4.4 for detailed definitions. Similar to the filtering setting used in the evaluation process of static KG link prediction, TKG link prediction also employs filtering during evaluation. Two types of filtering settings are commonly used. The first type of filtering is called time-unaware filtering [84], which is as same as the filtering setting proposed in [13] for static KG link prediction. The second type of filtering is called time-aware filtering [70]. It refers to removing from the entity set all the candidate entities that form ground truth fact quadruples together with the link prediction query during ranking, where these facts can only appear at the prediction timestamp of the query t_q (except the test fact quadruple of interest). Time-aware filtering is more reasonable than time-unaware filtering in the context of TKG link prediction. For example, assume we have a test fact of interest (*Albert Einstein*, *study at*, *University of Zurich*, *1902*) in the test set, and we derive an object prediction query (*Albert Einstein*, *study at*, ?, *1902*) from this quadruple where the query time is *1902*. Additionally, we have another fact (*Albert Einstein*, *study at*, *ETH Zurich*, *1896*) in the test set. According to the time-unaware filtering setting, *ETH Zurich* will be filtered because it appears in the test set. However, it is unreasonable because *Albert Einstein* did not study at *ETH Zurich* in *1902*. In practice, a lot of related works provide both time-aware and time-unaware filtered results, in order to make the evaluation more comprehensive.

2.5.5 Natural Language Question Answering on Temporal Knowledge Graphs

Following [126], natural language question answering on TKGs (TKGQA) can be defined as

Definition 16 (Question Answering on Temporal Knowledge Graphs)**.** *Assume we have an underlying TKG $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$. Given a natural language question q, annotated entities and timestamps (or time periods) existing in q (there can be no annotated entity or time constraint), TKGQA aims to find the entity $e_q \in \mathcal{E}$ or $t_q \in \mathcal{T}$ that answer q.*

Natural language questions in TKGQA are proposed based on the facts in the underlying TKG, meaning that all the TKGQA questions are answerable if a model can perform perfect reasoning over the TKG and no additional information source is needed. Similar to KGQA on static KGs, TKGQA also does not provide the relation types for models. Models have to understand the natural language questions for answer inference. Besides, one key difference between KGQA and TKGQA is that TKGQA requires models to have the ability of temporal reasoning. This poses a great challenge to previous works on TKG KRL and KGQA since they are not capable of either natural language understanding or temporal reasoning.

Mainstream works on TKGQA can also be divided into semantic parsing-based and TKG embedding-based methods [136]. Different from the semantic parsing-based methods of KGQA, in the context of TKGQA, such methods (e.g., [80, 40, 30]) consider temporal operators during the parsing process to enable temporal reasoning. However, most of them still carry over the drawbacks of the semantic parsing methods in non-temporal KGQA, such as the need for costly logic form annotations and the requirement of complete TKGs for accurate predictions. By contrast, based on the success of KG embedding-based KGQA methods, various recent TKGQA models leverage TKG representations learned from KRL approaches to enable embedding-based TKGQA. Here we discuss several classic embedding-based TKGQA methods

- CronKGQA [126] is the extension of EmbedKGQA [127]. It consists of three modules: (1) the KRL module (TKG embedding module) trains entity/relation/time representations over the fact quadruples in the underlying TKG by using the TComplEx [91] score function; (2) the question representation module enables natural language understanding by leveraging a pre-trained BERT LM [38] together with projection

layers to encode each natural language question q into two separate low-dimensional question representations, i.e., $\mathbf{h}_q^{\text{ent}}$ for answering entities and $\mathbf{h}_q^{\text{time}}$ for answering the timestamps; (3) the answer selection module selects the predicted answer e_{ans} or t_{ans} with a score function in the same form as TComplEx

$$\begin{aligned} e_{\text{ans}} &= \text{argmax}_{o' \in \mathcal{E}} \text{Re}\left(\langle \mathbf{h}_s, \mathbf{h}_q^{\text{ent}}, \bar{\mathbf{h}}_{o'}, \mathbf{h}_t \rangle\right) \quad \mathbf{h}_s, \mathbf{h}_q^{\text{ent}}, \mathbf{h}_{o'}, \mathbf{h}_t \in \mathbb{C}^d, \\ t_{\text{ans}} &= \text{argmax}_{t' \in \mathcal{T}} \text{Re}\left(\langle \mathbf{h}_s, \mathbf{h}_q^{\text{time}}, \bar{\mathbf{h}}_o, \mathbf{h}_{t'} \rangle\right) \quad \mathbf{h}_s, \mathbf{h}_q^{\text{time}}, \mathbf{h}_o, \mathbf{h}_{t'} \in \mathbb{C}^d. \end{aligned} \tag{2.35}$$

Similar to the LM-encoded question representation in EmbedKGQA, entity-specific and time-specific question representations ($\mathbf{h}_q^{\text{ent}}$ and $\mathbf{h}_q^{\text{time}}$) have been transformed into complex-valued vectors where each element is a complex number, as implemented in ComplEx [145]. d is the dimension size. $o' \in \mathcal{E}$ and $t' \in \mathcal{T}$ are an candidate entity and an candidate timestamp as the answer to the question, respectively. $\text{Re}(\cdot)$ is the function taking the real part of its input. $\bar{\mathbf{h}}_o$ and $\bar{\mathbf{h}}_{o'}$ are the complex conjugate of $\mathbf{h}_o$ and $\mathbf{h}_{o'}$, respectively. $\langle \cdot, \cdot, \cdot, \cdot \rangle$ is a function that first computes the element-wise product of four input vectors, and then does a sum over all elements.

- EXAQT [81] consists of two stages. In the first stage, it first detects the entities in a natural language question and picks out the facts containing them from the underlying KG to form a subgraph corresponding to the question by using Group Steiner Trees (GST) [39]. Then a BERT LM [38] is employed to identify additional question-related temporal facts. The identified facts are included into the subgraph to output an answer graph for the next stage. In the second stage, EXAQT first summarizes four temporal categories for natural language questions, i.e., EXPLICIT, IMPLICIT, TEMPORAL ANSWER and ORDINAL[18], and tags them with their corresponding labels. It then finds the temporal signals in each question following the policy specified in [79]. After that, EXAQT encodes the temporal categories as well as the temporal signals into multi-hot vectors. The textual information of a question is encoded by an LSTM and further concatenated with multi-hot vectors to generate an initialization of question representation. The entity representations are initialized with Wikipedia2Vec [168] and updated with an L layer time-aware relational GNN. To compute the time-aware representation of an entity within the answer graph, in each layer, EXAQT leverages a sinusoidal position encoding function [178] to represent the temporal information of the temporal facts related to this entity and

[18]Please refer to [81] for a detailed explanation of categorization.

uses another LSTM to aggregate over all related temporal facts. An attention module is further designed to differentiate the contributions of repeated facts during the update of entity representations. The predicted answer to each question is selected from the elements (entities or timestamps) in the answer graph by computing element probabilities with an MLP.

- TempoQR [108] encodes a natural language question into a sequence of token representations with a pre-trained BERT [38]. The representations of the tokens linked with entities are first substituted by the TKG representations from a TComplEx [91] model pre-trained on the underlying TKG. To retrieve the time scope associated with the question, TempoQR proposes two strategies: hard supervison and soft supervision. Hard supervision selects all the facts involving annotated question entities and collects the timestamps of them. The maximum and minimum timestamps among them lead to the time scope of the question. Soft supervision leverages the learned entity representations from TComplEx and recomputes the approximate timestamp representations specifying the time scope (which consists of a start time representation and an end time representation) based on interchanged question entities. The representations of time scope are added on the representations of the entity tokens to achieve temporal positional encoding. And the updated sequence of token representations is fed into an L-layer Transformer [149], where the output serves as the final question representation. The TComplEx score function is finally used as a score function for answer prediction.

Many recent TKGQA methods are proposed based on these works. Please refer to this survey [136] for the introduction of them.

Evaluation

Same as KGQA, TKGQA can also be framed as a ranking task. For each natural language question q together with its annotated entities and timestamps, a TKGQA model is asked to compute a score for any candidate entity $o' \in \mathcal{E}$ or candidate timestamp $t' \in \mathcal{T}$. An ideal TKGQA model should assign the highest score for the ground truth answer entity or timestamp. Thus, MRR and Hits@k are also used to evaluate models on TKGQA.

Limitations of Previous TKGQA

Previous TKGQA have limitations, which we discuss from three perspectives: task setting, datasets and models.

Perspective of Task Setting. Previous related works aim to develop TKGQA systems that answer temporal questions based on the facts from a fixed time period, where an underlying TKG spanning this period is observable and can be fully used for answer inference. However, in the real world, forecasting is also a common and critical situation. For example, predicting potential political shifts around the globe, such as the rise of authoritarian regimes and the outbreak of conflicts, is highly valuable because this allows governments and international organizations to take preventive measures to address emerging threats and promote long-term peacekeeping. To this end, it is crucial to develop TKGQA systems that are capable of answering questions about the future based on historical knowledge. In Chapter 6, we will thoroughly discuss how we propose a new task of TKGQA, namely forecasting TKGQA, to address the above-mentioned limitation. In forecasting TKGQA, every natural language question can be answered only when models are capable of future inference. We also show with comprehensive experiments that the TKGQA systems for non-forecasting TKGQA are not suitable for answering forecasting questions, demonstrating the challenge of this new task.

Perspective of Datasets. The earliest datasets for temporal KGQA are TempQuestions [79] and TimeQuestions [81]. Although all the questions in them are temporal questions, their underlying KGs are not temporal KGs. To solve this problem, CronQuestions [126], MultiTQ [31] and MusTQ [177] are proposed, taking two TKGs, i.e., Wikidata [152] and ICEWS [58], as their underlying knowledge bases, respectively. One limitation of CronQuestions, MultiTQ and MusTQ is that they are not built for the forecasting scenario. TKGQA models can access the ground truth fact from the underlying TKG that directly indicates the answer to each temporal question. For example, the TKG facts from *2003*, including (*Stephen Robert Jordan*, *member of sports team*, *Manchester City*, *2003*), are all observable to answer the question *Which team was Stephen Robert Jordan part of in 2003?*. This means that as long as the QA model can perform extensive search on the underlying TKG based on an accurate understanding of the natural language question, they can find the ground truth answer. By contrast, in the forecasting scenario, QA systems should be evaluated on their ability to predict the future, meaning that they

should not be able to directly retrieve answers from the observed facts in the underlying TKG. This requires new dedicated datasets to simulate the forecasting setting to drive the development of TKGQA models for future predictions. Another point worth noting is that previous TKGQA datasets only contain entity and time prediction questions that are closely bounded to the task of link prediction over TKGs, greatly suppressing the diversity of question types. Meanwhile, in a more general domain of reading comprehension QA, yes-no questions and multiple-choice questions have been extensively studied, e.g., [83]. To this end, we propose a new dataset, i.e., ForecastTKGQuestions, that is exclusively developed for the new task of forecasting TKGQA. ForecastTKGQuestions not only contains entity prediction questions, it also contains yes-no type questions and multiple-choice questions. It is large-scale, including 717k forecasting questions, and is around 1.7 times as large as CronQuestions. We give detailed discussions about it in Chapter 6.

Perspective of Models. Previous TKGQA models are not designed for answering forecasting questions. For example, several TKGQA methods leverage TComplEx as the backend to learn entity representations for answer inference (e.g., CronKGQA and TempoQR). Since TComplEx is a TKG score function developed for interpolated link prediction, the TKG embedding vectors provided by it are not useful in the forecasting setting. Therefore, it is important to develop specific modules that equip TKGQA systems with forecasting capabilities. In Chapter 6, we discuss the design of a new model, ForecastTKGQA, tailored for the forecasting setting. ForecastTKGQA utilizes a TKG extrapolation module to enable the whole system to achieve future prediction.

2.5.6 Inductive Representation Learning on Temporal Knowledge Graphs

Same as static KGs, TKGs are also ever-evolving, with new entities and relations constantly emerging. Inductive learning on TKGs refers to the ability of a model in making inferences about the entities and relations unseen in the training data, in particular in a temporal context. Building on inductive learning over static KGs (discussed in Section 2.4.6), TKGs introduce additional challenges due to the dynamic nature of temporal data. As TKGs capture evolving relationships among entities over time, the ability to generalize from limited data becomes even more critical. Inductive learning on TKGs must handle the introduction of new entities and relations while leveraging the temporal information to

make accurate predictions, making it more challenging than inductive learning on static KGs.

Compared with inductive representation learning on static KGs, there are only a limited number of works studying how to equip KRL methods on TKGs with inductive capabilities. In this section, we give an introduction of the current advancement of inductive representation learning on TKGs. Following Section 2.4.6, we divide the rest of this section into three parts: the first focuses on inductive learning for unseen entities, the second on unseen relations, and the third discusses addressing both unseen entities and relations simultaneously. We also follow previous works of inductive KG representation learning and focus exclusively on the problem of inductive TKG link prediction.

Inductive Learning on Temporal Knowledge Graph Entities

In this thesis, we present the first approach of inductive entity representation learning on TKGs, i.e., FILT [46]. We give a brief introduction of it in this section to lay a foundation for the discussion about follow-up works. More details about [46] can be found in Chapter 3. As the first work studying inductive learning on unseen entities, FILT is inspired by GEN [7] (an inductive entity representation learning method discussed in Section 2.4.6) and formulates TKG inductive learning into a meta-learning task, i.e., TKG few-shot out-of-graph (OOG) link prediction. TKG few-shot OOG link prediction is based on TKG interpolation and thus does not involve future forecasting. It is also a task combining the SI and FI settings where newly-emerged entities can be connected to either seen entities or other unseen entities. We can define TKG few-shot OOG link prediction as

Definition 17 (Temporal Knowledge Graph Few-Shot Out-of-Graph Link Prediction)**.** *Given an observed background TKG $\mathcal{G}_{back} \subseteq \mathcal{E}_{back} \times \mathcal{R} \times \mathcal{E}_{back} \times \mathcal{T}$, an unseen entity e' is an entity $e' \in \mathcal{E}'$, where $\mathcal{E}' \cap \mathcal{E}_{back} = \emptyset$. Assume we further observe K associated quadruples (support set) for each unseen entity e' in the form of $(e', r, \tilde{e}, t)$ (or $(\tilde{e}, r, e', t)$), where $\tilde{e} \in (\mathcal{E}_{back} \cup \mathcal{E}')$, $r \in \mathcal{R}$, $t \in \mathcal{T}$, and K is a small number denoting the shot size, e.g., 1 or 3. TKG few-shot out-of-graph link prediction aims to predict the missing entities from the link prediction queries $(e', r_q, ?, t_q)$ (or $(?, r_q, e', t_q)$) derived from unobserved quadruples (query set) containing unseen entities, where $r_q \in \mathcal{R}$, $t_q \in \mathcal{T}$.*

FILT designs a meta-learning framework to transfer knowledge from seen entities to unseen entities. It employs episodic training [151] to train a time-aware GNN-based model to "learn how to learn" entity representations of new-emerged entities based on observed

few-shot associated facts. To enhance model's inductive power, FILT also leverages entity concepts provided by the temporal knowledge bases. FILT learns concept representations and augments GNN's output with them to inject conceptual knowledge into entity representations. Although few-shot entities are observed in only a few, i.e., K, facts when they are added into a knowledge base, their concepts are pre-defined. To this end, conceptual knowledge can serve as a strong information source for inductive entity representation learning. We comprehensively discuss FILT in Chapter 3. Please refer to the corresponding chapter for more details, including the model structure of FILT and how the entity concepts are defined and encoded.

To better solve TKG few-shot OOG link prediction, we also present another method, i.e., FITCARL [47], in this thesis. FITCARL improves inductive entity representation learning with confidence-augmented RL. We train an RL-based model with episodic training to enable it to search for the link prediction answer entity given only a few observed facts associated with few-shot entities. FITCARL employs a Transformer [149] with time-aware positional encoding to capture few-shot information. It follows a learned policy for graph traversal among TKG entities, guided by a concept regularizer leveraging entity concepts introduced in [46]. One challenge of graph traversal under the few-shot setting is that when the agent lands on a few-shot entity, the action space will be highly limited because only K edges are connected to this entity. This would potentially lead to unreasonable traversal paths. To address this problem, we introduce a module that computes the confidence of each candidate action during graph traversal, integrating it into the policy for action selection. Please refer to Chapter 4 for more details of FITCARL, where we hold a comprehensive discussion about it.

A concurrent work [158] draws attention to inductive learning over unseen entities in the setting of TKG extrapolation. It formulates a few-shot learning task named as few-shot TKG reasoning

Definition 18 (Few-Shot Temporal Knowledge Graph Reasoning)**.** *Given a TKG* $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$*, an unseen entity* $e' \in \mathcal{E}'$ *(*$\mathcal{E}' \cap \mathcal{E} = \emptyset$*) is an entity that joins* $\mathcal{G}$ *at* $t' \in \mathcal{T}'$*, where the minimum timestamp in* $\mathcal{T}'$ *is greater than the maximum timestamp in* $\mathcal{T}$*. Assume we further observe the first* K *associated fact quadruples* $\{(e', r_i, \tilde{e}_i, t_i)$ *or* $(\tilde{e}_i, r_i, e', t_i)\}_{i=1}^{K}$ *(support set) for each unseen entity* e'*, where* $\tilde{e}_i \in (\mathcal{E} \cup \mathcal{E}')$*,* $r_i \in \mathcal{R}$*,* $t_i \in \mathcal{T}'$*, and* K *is a small number denoting the shot size, e.g., 1 or 3. Few-shot TKG reasoning aims to predict the missing entities from the link prediction queries* $(e', r_q, ?, t_q)$ *(or* $(?, r_q, e', t_q)$*) derived from unobserved quadruples (query set) containing unseen entities, where* $r_q \in \mathcal{R}$*,* $t_q \in \mathcal{T}'$*.*

To solve few-shot TKG reasoning, a model named MetaTKGR is proposed in [158]. MetaTKGR first designs a GNN-based attentional graph encoder that aggregates historical information related to each unseen entity. Multi-hop temporal neighbors are considered to overcome the data scarcity problem brought by the few-shot setting. MetaTKGR utilizes the MAML [50] framework for meta-learning, where an inner loop updates the parameters using the support set, and an outer loop calculates the loss over the query set.

Another closely related work is MetaTKG [163]. MetaTKG is a plug-and-play approach that enables TKG extrapolation methods to deal with newly-emerged entities. It groups each pair of neighboring TKG snapshots and treats the pairs as meta-learning tasks. In each pair, the snapshot at the former timestamp serves as the support set and the one at the latter timestamp is taken as the query set. MetaTKG uses a Temporal Meta Learner that introduces MAML into the training process, aiming to let the backbone TKG model quickly learn the evolutionary meta-knowledge over time. It achieves enhancement of model performance on the entities with little historical information. One point worth noting is that MetaTKG does not involve FSL. Each new entity is not limited to having only a few observed associated facts, which means MetaTKG may not be fully applicable in more extreme scenarios.

Apart from these methods, SST-BERT [28] explores leveraging pre-trained LMs' ability of natural language understanding to enable inductive entity representation learning. It first fine-tunes a BERT [38] model based on the facts in the background TKG. Each fact for fine-tuning is transformed into a group of textual sentences describing the fact, including the historical description of subject and object entities and the description of relation path between them. The fine-tuning objective is based on the classic masked language modeling task, but is specifically adapted by [28] to emphasize temporal expressions within the sentences. While the fine-tuned model can be applied to relation prediction among unseen entities, it falls short in performing entity prediction.

Inductive Learning on Temporal Knowledge Graph Relations

OAT [112] and MOST [42] are the earliest works studying inductive learning on TKG relations. They consider the one-shot scenario, where each new relation is introduced with only one associated fact that can be leveraged for inference. OAT first proposes a task named one-shot link prediction on TKGs. It refers to predicting future facts related to one-shot relations, aligning with the TKG extrapolation setting. MOST redefines the task proposed in OAT and formulates one-shot relational learning into two separate tasks

covering both interpolation and extrapolation settings, i.e., one-shot TKG interpolated and extrapolated link prediction, which can be defined as

Definition 19 (One-Shot Temporal Knowledge Graph Interpolated Link Prediction)**.** *Given an observed background TKG* $\mathcal{G}_{back} \subseteq \mathcal{E} \times \mathcal{R}_{back} \times \mathcal{E} \times \mathcal{T}$ *and a set of unseen relations* $\mathcal{R}'$*, where* $\mathcal{R}' \cap \mathcal{R}_{back} = \emptyset$*. Assume we further observe only one quadruple* (s_0, r', o_0, t_0) *corresponding to each unseen relation* r'*, where* $r' \in \mathcal{R}'$*,* $s_0, o_0 \in \mathcal{E}$*. Based on* (s_0, r', o_0, t_0) *(support quadruple) and the whole background graph* $\mathcal{G}_{back}$*, one-shot TKG interpolated link prediction aims to predict the missing entity of each link prediction query, i.e.,* $(s_q, r', ?, t_q)$ *or* $(?, r', o_q, t_q)$*, derived from the unobserved quadruples* (s_q, r', o_q, t_q) *(query set) containing* r'*, where* $s_q, o_q \in \mathcal{E}$ *and* $t_q \in \mathcal{T}$*.*

Definition 20 (One-Shot Temporal Knowledge Graph Extrapolated Link Prediction)**.** *Given an observed background TKG* $\mathcal{G}_{back} \subseteq \mathcal{E} \times \mathcal{R}_{back} \times \mathcal{E} \times \mathcal{T}$ *and a set of unseen relations* $\mathcal{R}'$*, where* $\mathcal{R}' \cap \mathcal{R}_{back} = \emptyset$*. Assume we further observe only one quadruple* (s_0, r', o_0, t_0) *corresponding to each unseen relation* r'*, where* $r' \in \mathcal{R}'$*,* $s_0, o_0 \in \mathcal{E}$*. Based on* (s_0, r', o_0, t_0) *(support quadruple) and the facts happening prior to* t_0 *in the background graph* $\mathcal{G}_{back}$*, one-shot TKG extrapolated link prediction aims to predict the missing entity of each link prediction query, i.e.,* $(s_q, r', ?, t_q)$ *or* $(?, r', o_q, t_q)$*, derived from the unobserved quadruples* (s_q, r', o_q, t_q) *(query set) containing* r'*, where* $s_q, o_q \in \mathcal{E}$*,* $t_q \in \mathcal{T}$ *and* $t_0 < t_q$*.*

OAT achieves inductive learning in the extrapolation setting by using episodic training to train an attentional GNN together with a similarity-based decoding function. The decoding function compares the to-be-predicted links with the support facts to determine the link prediction answer. Based on a similar training paradigm, MOST develops a model that first extracts meta information with a time-aware graph encoder, and then learns the meta representation of each newly-emerged relation for link inference with a customized decoding function inspired by RotatE [140]. Time-aware entity representations are input into the decoding function to achieve temporal reasoning. To adapt to different link prediction settings, MOST further develops two model variants, i.e., MOST-TA and MOST-TD. MOST-TA learns representations for timestamps, while MOST-TD learns time difference representations. Experimental results show that MOST-TA is more suitable for the interpolation setting and MOST-TD better suits extrapolation.

One limitation of the FSL methods is that they require at least one observed fact related to each unseen relation to achieve link prediction on other associated facts. In this thesis,

we propose a zero-shot relational learning method for TKGs [41], specifically designed to address the zero-shot scenario. We first define a zero-shot TKG forecasting task as follows

Definition 21 (Zero-Shot Temporal Knowledge Graph Forecasting)**.** *Assume we have a ground truth TKG* $\mathcal{G}_{gt} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$*, where* $\mathcal{R}$ *can be split into seen* $\mathcal{R}_{se}$ *and unseen* $\mathcal{R}_{un}$ *relations (*$\mathcal{R} = \mathcal{R}_{se} \cup \mathcal{R}_{un}, \mathcal{R}_{se} \cap \mathcal{R}_{un} = \emptyset$*). Given a link prediction query* $(s_q, r_q, ?, t_q)$ *(or* $(o_q, r_q, ?, t_q)$*) whose query relation* $r_q \in \mathcal{R}_{un}$*, models are asked to predict the missing object* o_q *(or subject* s_q*) based on the facts* $\mathcal{O} = \{(s, r_i, o, t_i) \in \mathcal{G}_{gt} | t_i < t_q, r_i \in \mathcal{R}_{se}\}$ *containing seen relations and happening before* t_q*.*

To solve this task, we propose a plug-and-play model named zrLLM. zrLLM first uses an LLM, i.e., GPT-3.5[19], to produce enriched relation descriptions (ERDs) for TKG relations (whether seen or unseen) based on their textual descriptions provided in TKG datasets. It then generates relation representations by using the encoder of another LLM, i.e., T5-11B [122]. The ERDs are input into T5-11B's encoder and the output relation representations can be directly implemented with traditional TKG forecasting models (e.g., TANGO [71]). Finally, zrLLM uses a relation history learner to capture historical relation patterns based on LLM-empowered relation representations. Through these steps, zrLLM aligns the natural language space of LLMs with the embedding space of TKG forecasting models, instead of relying on the models to learn relation representations purely from the observed graph contexts. Zero-shot relations can be represented using LLM-enhanced representations that incorporate semantic information, even without any observed associated facts. For more details of zrLLM, please refer to Chapter 5.

Inductive Learning on Knowledge Graph Entities & Relations

Recently, there is one preliminary work, i.e., MTKGE [29], studying how to address both unseen entities and relations simultaneously on TKGs. MTKGE deals with unseen relations by using two GNN-based modules to capture information from Relative Position Pattern Graphs (RPPGs) and Temporal Sequence Pattern Graph (TSPGs). An RPPG treats relations as nodes and labels the edges with the relative position features, which is the same as the multi-relational meta-graphs of fundamental relation interactions proposed in ULTRA[55] (as discussed in Section 2.4.6). A TSPG also treats relations as nodes, but focus on the temporal order between a pair of relations. Since each fact in TKG has a timestamp label, the timestamps of two facts connected by the same entity can

[19]https://platform.openai.com/docs/model-index-for-researchers

indicate the temporal order of these facts, and it also specifies the temporal order of their corresponding relations. TSPG labels edges with three meta-time relations, i.e., forward (source node happens before destination node), backward (source node happens after destination node) and meantime (source and destination nodes happen together). For unseen entities, MTKGE initializes their representations by utilizing their surrounding relations in the original TKG. The meta information learned from RPPGs and TSPGs helps to enrich unseen entities with reasonable characters. Finally, MTKGE proposes an L-layer multi-relational GNN to further update representations for entities, relations and timestamps. A decoding function (can be any TKG score function) is used to compute plausibility scores of temporal facts. One limitation of MTKGE is that it requires a support graph containing a substantial number of data examples related to the unseen entities and relations to learn expressive representations, which is not always available in real-world scenarios. Further efforts should be made to generalize MTKGE to the few-shot, or even zero-shot, setting.

We have introduced a number of important works within the field of inductive representation learning on TKGs. For discussions of more related works, please refer to the following surveys [155, 18].

Symbolic Approaches for Inductive Learning on Knowledge Graphs.

Symbolic TKG relational learning approaches (such as TLogic [106]) are naturally capable of inductive learning on unseen entities because they learn entity-agnostic temporal rules based on TKG relations. Same as the symbolic KG reasoning methods, their inductive capability is only shown on unseen entities because of the strong bound between relations and exploited symbolic rules. As this thesis lays emphasis on KRL-based approaches, we will not go into more details about symbolic approaches. Please refer to the following surveys for a better understanding [17, 155].

Chapter 3

Few-Shot Inductive Learning on Temporal Knowledge Graphs using Concept-Aware Information

This chapter contains the publication

> **Zifeng Ding**[*], Jingpei Wu[*], Bailan He, Yunpu Ma, Zhen Han and Volker Tresp. Few-Shot Inductive Learning on Temporal Knowledge Graphs using Concept-Aware Information. In *4th Conference on Automated Knowledge Base Construction*, 2022. [*]Equal Contribution. URL: `https://www.akbc.ws/2022/assets/pdfs/6_few_shot_inductive_learning_on.pdf`

Few-Shot Inductive Learning on Temporal Knowledge Graphs using Concept-Aware Information

Zifeng Ding[*1,2] ZIFENG.DING@CAMPUS.LMU.DE
Jingpei Wu[*3] JINGPEI.WU@TUM.DE
Bailan He[1] BAILAN.HE@CAMPUS.LMU.DE
Yunpu Ma[1,2] COGNITIVE.YUNPU@GMAIL.COM
Zhen Han[†1,2] ZHEN.HAN@CAMPUS.LMU.DE
Volker Tresp[†1,2] VOLKER.TRESP@SIEMENS.COM
[1]*Ludwig Maximilian University of Munich*
[2]*Corporate Technology, Siemens AG*
[3]*Technical University of Munich*

Abstract

Knowledge graph completion (KGC) aims to predict the missing links among knowledge graph (KG) entities. Though various methods have been developed for KGC, most of them can only deal with the KG entities seen in the training set and cannot perform well in predicting links concerning novel entities in the test set. Similar problem exists in temporal knowledge graphs (TKGs), and no previous temporal knowledge graph completion (TKGC) method is developed for modeling newly-emerged entities. Compared to KGs, TKGs require temporal reasoning techniques for modeling, which naturally increases the difficulty in dealing with novel, yet unseen entities. In this work, we focus on the inductive learning of unseen entities' representations on TKGs. We propose a few-shot out-of-graph (OOG) link prediction task for TKGs, where we predict the missing entities from the links concerning unseen entities by employing a meta-learning framework and utilizing the meta-information provided by only few edges associated with each unseen entity. We construct three new datasets for TKG few-shot OOG link prediction, and we propose a model that mines the concept-aware information among entities. Experimental results show that our model achieves superior performance on all three datasets and our concept-aware modeling component demonstrates a strong effect.

1. Introduction

Knowledge graphs (KGs) store factual information in the form of triples, i.e., (s, r, o), where s, o, r denote the subject entity, the object entity, and the relation between them, respectively. KGs have already been widely used in a series of downstream tasks, e.g., question answering [Saxena et al., 2020, Ding et al., 2022b] and recommender systems [Wang et al., 2019c,a]. While KG triples are capable of representing facts, they cannot express their time validity. World knowledge is ever-changing, which means many facts have their own time validity, e.g., the fact (*Angela Merkel*, *is chancellor of*, *Germany*) is valid only before (*Olaf Scholz*, *is chancellor of*, *Germany*). To this end, temporal knowledge graphs (TKGs) are introduced to consider the time validity of facts by representing every fact with a quadruple, i.e., (s, r, o, t), where t denotes the time when the fact is valid.

∗. Equal contribution.
†. Corresponding author.

KGs and TKGs are known to suffer from incompleteness [Min et al., 2013, Leblay and Chekol, 2018]. Therefore, various methods have been developed for automatically completing KGs [Nickel et al., 2011, Bordes et al., 2013, Trouillon et al., 2016, Sun et al., 2019, Guo and Kok, 2021b] and TKGs [Tresp et al., 2015, Leblay and Chekol, 2018, Ma et al., 2019, Jung et al., 2021, Ding et al., 2021]. Though these methods achieve superior performance on knowledge graph completion (KGC) and temporal knowledge graph completion (TKGC), they have their limitations. In real-world scenarios, KGs and TKGs evolve over time, indicating that new (unseen) entities may emerge constantly [Shi and Weninger, 2018]. Besides, real-world KGs exhibit long-tail distributions, where a large portion of entities only have few edges [Baek et al., 2020]. This also applies to TKGs, e.g., the entity frequency distribution of ICEWS datasets (Appendix A). Traditional KGC and TKGC methods learn the representations of the observed (seen) entities, and perform link prediction over a fixed set of entities. To learn the optimal representations of the observed entities, these methods require a large number of training examples associated with each of them. [Baek et al., 2020] shows that traditional KGC methods show poor performance when they are used to predict the links concerning newly-emerged, yet unseen entities. In our work, we also observe that traditional TKGC methods share the same problem (Section 5.3).

To tackle the limitations of traditional TKGC methods, we propose the TKG few-shot out-of-graph (OOG) link prediction task and a TKG reasoning model for better learning the inductive representations of newly-emerged entities in TKGs. Inspired by recent work that mines shared concepts of stocks for improving stock prediction [Li et al., 2020, Xu et al., 2021b], we devise a module, taking advantage of the entity concepts provided by the temporal knowledge bases. The contribution of our work is three-folded:

- We propose the TKG few-shot out-of-graph (OOG) link prediction task. To better learn the inductive representations of unseen entities and predict their links, we propose a meta-learning-based model. To the best of our knowledge, this is the first work aiming to improve the link prediction performance concerning unseen entities in TKGs.

- We extract the entity concepts from the temporal knowledge bases and take them as additional information to boost our model performance. We design an effective module to learn concept-aware information. The experimental results show that introducing such information helps to learn better representations for unseen entities in the inductive setting.

- We propose three new datasets for TKG few-shot OOG link prediction, i.e., ICEWS14-OOG, ICEWS18-OOG and ICEWS0515-OOG. We compare our model with several baseline methods. Experimental results show that our model outperforms all the baselines on all three datasets.

2. Related Work

Knowledge graph embedding methods. Knowledge graph embedding (KGE) methods can be split into two categories. Some methods design scoring functions to compute the plausibility scores of KG facts [Bordes et al., 2013, Trouillon et al., 2016, Sun et al., 2019, Guo and Kok, 2021b], while other KGE methods employ neural-based structures, e.g., graph neural networks (GNNs), to better capture the structural dependencies of KGs [Schlichtkrull et al., 2018, Vashishth et al., 2020, Yu et al., 2021]. By combining neural-based graph encoders with KG scoring functions, these methods achieve superior performance in KG reasoning tasks.

Temporal knowledge graph embedding methods. To deal with the temporal constraints in TKG facts, two lines of temporal knowledge graph embedding (TKGE) methods have been developed. The first line of methods designs novel time-aware scoring functions for characterizing extra time information [Leblay and Chekol, 2018, Ma et al., 2019, Lacroix et al., 2020, Sadeghian et al., 2021, Han et al., 2020a, 2021c]. The second line of methods models temporal information by employing neural structures, e.g., GNNs and recurrent models. [Han et al., 2021a, Jung et al., 2021, Ding et al., 2021, Han et al., 2020b, Sun et al., 2021] sample every entity's temporal neighbors and use GNNs to learn time-aware representations of them. [Wu et al., 2020] and [Han et al., 2021b] model structural information with GNNs, and they achieve temporal reasoning by utilizing a gated recurrent unit [Cho et al., 2014] and a neural ordinary differential equation [Chen et al., 2018], respectively.

Inductive learning on knowledge graphs. Traditional KGE and TKGE methods require a large number of training examples to learn entity representations. However, in real-world scenarios, KGs and TKGs are ever-evolving, and they exhibit long-tail distributions. New entities and relations emerge and a huge portion of them only have very few associated facts, thus causing traditional methods unable to learn optimal representations. To alleviate this problem, a line of work [Xiong et al., 2018, Chen et al., 2019, Sheng et al., 2020, Mirtaheri et al., 2021, Ding et al., 2022a] tries to employ meta-learning to learn inductive representations of unseen KG (or TKG) relations. Nevertheless, they are unable to deal with novel entities. Several methods try to deal with unseen (out-of-graph) entities in an inductive setting [Hamaguchi et al., 2017, Wang et al., 2019b, He et al., 2020]. They first learn representations of seen entities, and then use an auxiliary set to transfer knowledge from seen to unseen entities during inference. [Baek et al., 2020] proposes a more realistic task: few-shot out-of-graph (OOG) link prediction, where the links among unseen entities are also considered during evaluation and the representation of every unseen entity can only be derived from very few (number of shot size) edges. Baek et al. simulate the unseen entities in the training phase and introduce meta-learning for learning unseen entities' representations. Based on it, [Zhang et al., 2021] proposes a model using hyper-relation features to improve performance on few-shot OOG link prediction. Another series of work tries to include external information of entities, e.g., textual descriptions, to solve this problem [Xie et al., 2016, Wang et al., 2019d] and it turns out to be effective in modeling unseen entities. Though there exist various methods dealing with OOG unseen entities in KGs, there is still no method specifically designed to embed unseen entities inductively for TKGs.

3. Preliminaries and Task Formulation

Entity concepts in temporal knowledge graphs. Entity concepts describe the characteristics of KG entities. They are manually defined by humans and assigned to every KG entity. In the ICEWS database [Boschee et al., 2015], entities belong to several sectors, e.g., *Government*, *Executive Office*. Each entity's sectors are specified in the ICEWS weekly event data[1]. We treat the sectors of an entity as its concepts and learn concept representations as additional information. We observe that some region entities in the ICEWS database, e.g., South Korea and North America, have no specified sectors. We manually assign a new sector *Region* to them. We ensure that every entity has its own sectors. More details about concept extraction is presented in Appendix F.

Task formulation. We first give the definition of a temporal knowledge graph, then we formulate the TKG few-shot out-of-graph link prediction task.

1. https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/QI2T9A

Definition 1 (Temporal Knowledge Graph (TKG)). Let $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ denote a finite set of entities, relations and timestamps, respectively. A temporal knowledge graph (TKG) $\mathcal{G}$ can be taken as a finite set of TKG facts represented by their associated quadruples, i.e., $\mathcal{G} = \{(s, r, o, t)|s, o \in \mathcal{E}, r \in \mathcal{R}, t \in \mathcal{T}\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$.

Definition 2 (Temporal Knowledge Graph Few-Shot Out-of-Graph Link Prediction). Given an observed background TKG $\mathcal{G}_{\text{back}} \subseteq \mathcal{E}_{\text{back}} \times \mathcal{R} \times \mathcal{E}_{\text{back}} \times \mathcal{T}$, an unseen entity e' is an entity $e' \in \mathcal{E}'$, where $\mathcal{E}' \cap \mathcal{E}_{\text{back}} = \emptyset$. Assume we further observe K associated quadruples for each unseen entity e' in the form of $(e', r, \tilde{e}, t)$ (or $(\tilde{e}, r, e', t)$), where $\tilde{e} \in (\mathcal{E}_{\text{back}} \cup \mathcal{E}')$, $r \in \mathcal{R}$, $t \in \mathcal{T}$, and K is a small number denoting the shot size, e.g., 1 or 3. TKG few-shot out-of-graph link prediction aims to predict the missing entities from the link prediction queries $(e', r_q, ?, t_q)$ (or $(?, r_q, e', t_q)$) derived from unobserved quadruples containing unseen entities, where $r_q \in \mathcal{R}$, $t_q \in \mathcal{T}$.

We further formulate the TKG few-shot OOG link prediction task into a meta-learning problem. For a TKG $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$, we first select a group of entities $\mathcal{E}'$, where each entity's number of associated quadruples is between a lower and a higher threshold. We aim to pick out the entities that are not frequently mentioned in TKG facts since newly-emerged entities normally are coupled with only several edges. We randomly split these entities into three groups $\mathcal{E}'_{\text{meta-train}}$, $\mathcal{E}'_{\text{meta-valid}}$ and $\mathcal{E}'_{\text{meta-test}}$. For each group, we treat the union of all the quadruples associated to this group's entities as the corresponding meta-learning set, e.g., the meta-training set $\mathbb{T}_{\text{meta-train}}$ is formulated as $\{(e', r, \tilde{e}, t)|\tilde{e} \in \mathcal{E}, r \in \mathcal{R}, e' \in \mathcal{E}'_{\text{meta-train}}, t \in \mathcal{T}\} \cup \{(\tilde{e}, r, e', t)|\tilde{e} \in \mathcal{E}, r \in \mathcal{R}, e' \in \mathcal{E}'_{\text{meta-train}}, t \in \mathcal{T}\}$. We ensure that there exists no link between every two of the meta-learning sets. The associated quadruples of the rest entities form a background graph $\mathcal{G}_{\text{back}} \subseteq \mathcal{E}_{\text{back}} \times \mathcal{R} \times \mathcal{E}_{\text{back}} \times \mathcal{T}$, where $\mathcal{E}' \cap \mathcal{E}_{\text{back}} = \emptyset$ and $\mathcal{E} = (\mathcal{E}_{\text{back}} \cup \mathcal{E}')$. We take the meta-training entities $\mathcal{E}'_{\text{meta-train}}$ as simulated unseen entities and try to learn how to transfer knowledge from seen entities $\mathcal{E}_{\text{back}}$ to them during meta-training. The entities in $\mathcal{E}'_{\text{meta-valid}}$ and $\mathcal{E}'_{\text{meta-test}}$ are real unseen entities that are used to evaluate the model performance.

Based on [Baek et al., 2020], we define a meta-training task T as follows. In each task T, we first randomly sample N simulated unseen entities $\mathcal{E}_T$ from $\mathcal{E}'_{\text{meta-train}}$. Then we randomly select K associated quadruples for each $e' \in \mathcal{E}_T$ as its support quadruples $\mathcal{S}_{e'} = \{(e', r_i, \tilde{e}_i, t_i) \text{ or } (\tilde{e}_i, r_i, e', t_i)\}_{i=1}^{K}$, where K is the shot size and $\tilde{e}_i \in (\mathcal{E}_{\text{back}} \cup \mathcal{E}')$. The rest of e''s quadruples are taken as its query quadruples $\mathcal{Q}_{e'} = \{(e', r_i, \tilde{e}_i, t_i) \text{ or } (\tilde{e}_i, r_i, e', t_i)\}_{i=K+1}^{M_{e'}}$, where $M_{e'}$ denotes the number of e''s associated quadruples in $\mathbb{T}_{\text{meta-train}}$ and $\tilde{e}_i \in (\mathcal{E}_{\text{back}} \cup \mathcal{E}')$. For every meta-training task T, the aim of TKG few-shot OOG link prediction is to simultaneously predict the missing entities from the link prediction queries derived from the query quadruples associated to all the entities from $\mathcal{E}_T$, e.g., $(e', r_i, ?, t_i)$ or $(?, r_i, e', t_i)$. In this way, we simulate the situation that we simultaneously observe a bunch of unseen entities and each of them has only few edges, which is similar to how emerging entities appear in temporal knowledge bases. After meta-training, we validate our model on a meta-validation set $\mathbb{T}_{\text{meta-valid}}$ and test our model on a meta-test set $\mathbb{T}_{\text{meta-test}}$, where they contain all the quadruples associated to the entities in $\mathcal{E}'_{\text{meta-valid}}$ and $\mathcal{E}'_{\text{meta-test}}$, respectively. We do not sample N entities during meta-validation and meta-test. Instead, we treat all the entities in $\mathcal{E}'_{\text{meta-valid}}$ (or $\mathcal{E}'_{\text{meta-test}}$) as appearing at the same time. For a better understanding, we present Figure 5 to illustrate how we formulate the TKG few-shot OOG link prediction task into a meta-learning problem. We also discuss the difference between our proposed task and traditional TKGC in Appendix D.

We summarize the challenge of TKG few-shot OOG link prediction as follows: (1) TKG reasoning models are asked to predict the links concerning the newly-emerged entities that are completely unseen during the training process; (2) Only a small number (K) of edges associated with

each newly-emerged entity are observable to support predicting the unobserved links concerning this entity.

4. Our Method

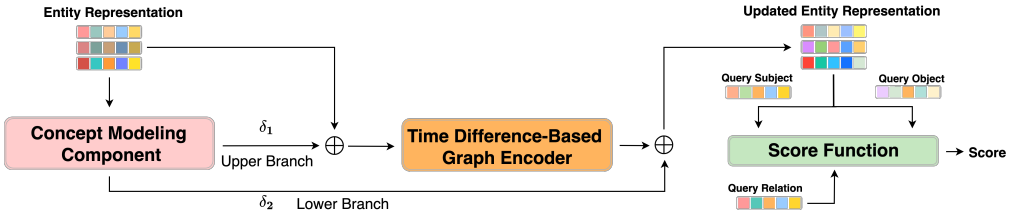


Figure 1: Model structure of FILT. Assume we have an unseen entity e', and we want to predict a link corresponding to $(e', r_i, \tilde{e}_i, t_i) \in \mathcal{Q}_{e'}$. We derive the concept representations in the concept modeling component and use a time difference-based graph encoder for learning e''s time-aware representation. We take the representations of r_i and $\tilde{e}_i$ to compute the plausibility score of the link.

We propose a model dealing with **f**ew-shot **i**nductive **l**earning on **T**KGs (FILT). Figure 1 shows the model structure of FILT. It consists of three components: (1) Concept modeling component that represents entity concepts based on seen entities' representations; (2) Time difference-based graph encoder that learns the contextualized representations of unseen entities; (3) KG scoring function that computes the plausibility scores of the TKG quadruples concerning unseen entities.

4.1 Concept Modeling Component

When a new entity emerges in a TKG, though there might be only few observed associated edges, some of its concepts, e.g., which sectors it belongs to, are already known. Since every entity concept is shared across all the entities in this TKG, we can learn concept information from seen entities and transfer it to newly-emerged entities.

Inspired by [Xu et al., 2021b] that mines concept-aware information for stock prediction, we develop a concept modeling component to learn TKG entity concepts as follows. First, we pre-train our background graph with ComplEx [Trouillon et al., 2016]. Note that only seen entities $\mathcal{E}_{\text{back}}$ are involved in the pre-training process. Assume we have a set of entity concepts $\mathcal{C}$, then we initialize the representation of every entity concept $c \in \mathcal{C}$ with its associated entities by averaging these entities' representations:

$$\mathbf{h}_c = \frac{1}{|\mathcal{N}_c|} \sum_{e \in \mathcal{N}_c} \mathbf{h}_e, \tag{1}$$

where $\mathbf{h}_c$ and $\mathbf{h}_e$ denote the representations of the concept c and the entity e, respectively. $\mathcal{N}_c$ denotes the neighborhood of the entity concept c. For example, if two TKG entities *Angela Merkel* and *Xi Jinping* both belong to the concept *Elite*, they will be included into *Elite*'s neighborhood. Since we want to distinguish the contributions of different entities to an entity concept, we then correct the concept representations as follows:

$$\mathbf{h}_c = \sum_{e_i \in \mathcal{N}_c} \alpha_c^{e_i} \mathbf{h}_{e_i}, \quad \alpha_c^{e_i} = \frac{\exp(\mathbf{h}_{e_i}^\top \mathbf{h}_c)}{\sum_{e_j \in \mathcal{N}_c} \exp(\mathbf{h}_{e_j}^\top \mathbf{h}_c)}. \tag{2}$$

After we correct the concept representations, we compute an entity's concept-aware information by aggregating the representations of its associated concepts:

$$\mathbf{h}_e^{\mathcal{C}_e} = \sum_{c_i \in \mathcal{C}_e} \beta_e^{c_i} \mathbf{h}_{c_i}, \quad \beta_e^{c_i} = \frac{\exp(\mathbf{h}_{c_i}^\top \mathbf{h}_e)}{\sum_{c_j \in \mathcal{C}_e} \exp(\mathbf{h}_{c_j}^\top \mathbf{h}_e)}. \tag{3}$$

$\mathcal{C}_e \subseteq \mathcal{C}$ denotes the set of all concepts associated to e. As shown in Figure 1, we inject the concept-aware information into two branches. We use two separate layers of feed forward neural network and project the concept-aware information onto two branches. The upper branch adds the concept information to the entity representations $\mathbf{h}_e := \mathbf{h}_e + \delta_1 \sigma(\mathbf{W}_c^1 \mathbf{h}_e^{\mathcal{C}_e})$ and take them as the input of our graph encoder. The lower branch processes the concept information $\delta_2 \sigma(\mathbf{W}_c^2 \mathbf{h}_e^{\mathcal{C}_e})$ and adds it to the entity representations after the graph aggregation step. δ_1 and δ_2 are two trainable weights deciding how much concept-aware information should be injected. $\mathbf{W}_c^1$ and $\mathbf{W}_c^2$ are two weight matrices and σ is an activation function. By employing the double branch structure, we not only include the concept information into the graph encoder, but also directly infuse it into the final entity representations for link prediction.

4.2 Time Difference-Based Graph Encoder

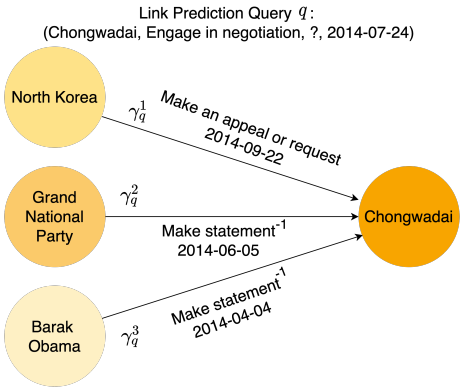


Figure 2: The structure of the time difference-based graph encoder. Assume we have an unseen entity *Chongwadai*, and we have a link prediction query (*Chongwadai*, *Engage in negotiation*, ?, *2014-07-24*), given three support quadruples, i.e., (*North Korea*, *Make an appeal or request*, *Chongwadai*, *2014-09-22*), (*Chongwadai*, *Make statement*, *Grand National Party*, *2014-06-05*), and (*Chongwadai*, *Make statement*, *Barak Obama*, *2014-04-04*). We use our graph encoder to compute the time-aware contextualized representation of *Chongwadai* at *2014-07-24*. For each temporal neighbor from a support quadruple, we compute its importance according to the time difference between *2014-07-24* and the timestamp of its corresponding support quadruple. We denote the temporal neighbors with colored circles. The color darkness of the circles implies the importance of the temporal neighbors during aggregation in Equation 4. The darker circle a temporal neighbor is represented with, the more important it is, i.e., $\gamma_q^2 > \gamma_q^1 > \gamma_q^3$.

To compute the contextualized representations of the unseen entities, we employ a time difference-based graph encoder. For each unseen entity e', assume we have a link prediction query $(e', r_q, ?, t_q)$

derived from a query quadruple $(e', r_q, \tilde{e}_q, t_q) \in \mathcal{Q}_{e'}$. We first find its temporal neighbors from its support quadruples $\mathcal{S}_{e'} = \{(e', r_i, \tilde{e}_i, t_i)$ or $(\tilde{e}_i, r_i, e', t_i)\}_{i=1}^{K}$, and then compute e''s time-aware representation at t_q through aggregation:

$$\mathbf{h}_{(e',t_q)} = \sum_{(\tilde{e}_i, r_i, t_i) \in \mathcal{N}_{e'}} \gamma_q^i \mathbf{W}_\mathbf{g}(\mathbf{h}_{\tilde{e}_i} \| \mathbf{h}_{r_i}), \quad \gamma_q^i = \frac{\exp(1/|t_q - t_i|)}{\sum_{(\tilde{e}_j, r_j, t_j) \in \mathcal{N}_{e'}} \exp(1/|t_q - t_j|)}. \tag{4}$$

$\mathbf{W}_g$ denotes the weight matrix in our graph encoder. $\mathcal{N}_{e'}$ denotes the observed neighborhood of e' and $|\mathcal{N}_{e'}| = K$. γ_q^i is the importance of the ith temporal neighbor $\tilde{e}_i$ based on the time difference between t_q and t_i. The smaller the time difference is, the more important a temporal neighbor is during aggregation. The motivation of our time difference-based graph encoder is that we assume the temporal neighbors that are temporally closer to the query timestamp t_q tend to contribute more to predicting the links at t_q. Since we take the temporal neighbors of an entity from its incoming edges, we transform every support quadruple whose form is $(e', r_i, \tilde{e}_i, t_i)$ to $(\tilde{e}_i, r_i^{-1}, e', t_i)$, where r_i^{-1} corresponds to the inverse relation of r_i. We manage to incorporate every support quadruple into the aggregation process with this quadruple transformation. Note that if $t_q - t_i = 0$, the denominator of the exponential term will be 0. Thus, we use a constant λ to assign a value to $\exp(1/|t_q - t_i|)$ if t_q equals t_i, and λ serves as a hyperparameter that can be tuned. Figure 2 illustrates the structure of our graph encoder with an example. After aggregation, we further infuse the concept-aware information from the lower branch into the output of our graph encoder: $\mathbf{h}_{(e',t_q)} := \mathbf{h}_{(e',t_q)} + \delta_2 \sigma(\mathbf{W}_c^2 \mathbf{h}_{e'}^{\mathcal{C}_{e'}})$. We show in Section 5.4 that our simple-structured graph encoder can beat more complicated structures in the TKG OOG link prediction task.

4.3 Parameter Learning

For each meta-training task T, we have N simulated unseen entities $\mathcal{E}_T$. We use the hinge loss for learning model parameters:

$$\mathcal{L} = \sum_{e' \in \mathcal{E}_T} \sum_{q^+ \in \mathcal{Q}_{e'}} \sum_{q^- \in \mathcal{Q}_{e'}^-} \max\{\theta - score(q^+) + score(q^-), 0\}. \tag{5}$$

$\theta > 0$ is the margin. q^+ denotes a query quadruple from e''s query set. q^- is generated by negative sampling [Bordes et al., 2013]. For every $q^+ = (e', r_q, \tilde{e}_q, t_q)$ (or $q^+ = (\tilde{e}_q, r_q, e', t_q)$), we corrupt $\tilde{e}_q$ with another entity $e^- \in \{\mathcal{E}', \mathcal{E}_{\text{back}}\}$. We map our learned representations to the complex space and use ComplEx [Trouillon et al., 2016] as our scoring function, i.e., $score = \text{Re} < \mathbf{h}_s, \mathbf{h}_r, \bar{\mathbf{h}}_o >$, where $\mathbf{h}_s$, $\mathbf{h}_o$ denote the representations of the subject entity and the object entity, respectively. $\mathbf{h}_r$ denotes the relation representation. Re means taking the real part, and $\bar{\mathbf{h}}_o$ means taking the conjugate of the vector $\mathbf{h}_o$.

5. Experiments

We compare FILT with several baselines on TKG few-shot OOG link prediction. To prove the effectiveness of the model components, we conduct several ablation studies. We also do further analysis to show the robustness of our method. Besides, we visualize the learned concept representations and show that our concept modeling component helps to capture the semantics of entity concepts.

5.1 Datasets

We propose three TKG few-shot OOG link prediction datasets, i.e., ICEWS14-OOG, ICEWS18-OOG, and ICEWS0515-OOG. We first take three subsets, i.e., ICEWS14, ICEWS18, and ICEWS05-15, from the Integrated Crisis Early Warning System (ICEWS) database [Boschee et al., 2015], where they contain the timestamped political facts in 2014, in 2018, and from 2005 to 2015, respectively. Following the data construction process of [Baek et al., 2020], for each subset, we first randomly sample half of the entities whose number of associated quadruples is between a lower and a higher threshold as unseen entities. Then we split the sampled entities into three groups $\mathcal{E}'_{\text{meta-train}}$, $\mathcal{E}'_{\text{meta-valid}}$, $\mathcal{E}'_{\text{meta-test}}$ ($\mathcal{E}'_{\text{meta-train}} \cap \mathcal{E}'_{\text{meta-valid}} = \emptyset$, $\mathcal{E}'_{\text{meta-train}} \cap \mathcal{E}'_{\text{meta-test}} = \emptyset$, $\mathcal{E}'_{\text{meta-valid}} \cap \mathcal{E}'_{\text{meta-test}} = \emptyset$), where $|\mathcal{E}'_{\text{meta-train}}| : |\mathcal{E}'_{\text{meta-valid}}| : |\mathcal{E}'_{\text{meta-test}}| \approx 8:1:1$. The associated quadruples of all the entities in $\mathcal{E}'_{\text{meta-train}}$/$\mathcal{E}'_{\text{meta-valid}}$/$\mathcal{E}'_{\text{meta-test}}$ form the meta-training/meta-validation/meta-test set. The rest of the quadruples without unseen entities are used for constructing a background graph $\mathcal{G}_{\text{back}}$. The dataset statistics are presented in Table 1. We present the dataset construction process in Appendix H.

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{T}\rvert$	$\lvert\mathcal{E}'_{\text{meta-train}}\rvert$	$\lvert\mathcal{E}'_{\text{meta-valid}}\rvert$	$\lvert\mathcal{E}'_{\text{meta-test}}\rvert$	N_{back}	$N_{\text{meta-train}}$	$N_{\text{meta-valid}}$	$N_{\text{meta-test}}$
ICEWS14-OOG	7128	230	365	385	48	49	83448	5772	718	705
ICEWS18-OOG	23033	256	304	1268	160	158	444269	19291	2425	2373
ICEWS0515-OOG	10488	251	4017	647	80	82	448695	10115	1217	1228

Table 1: Dataset statistics. $|\mathcal{E}'_{\text{meta-train}}|$, $|\mathcal{E}'_{\text{meta-valid}}|$, $|\mathcal{E}'_{\text{meta-test}}|$ denote the number of unseen entities in the meta-training set, meta-validation set, meta-test set, respectively. N_{back} denotes the number of quadruples in the background graph $\mathcal{G}_{\text{back}}$. $N_{\text{meta-train}}$, $N_{\text{meta-valid}}$, $N_{\text{meta-test}}$ denote the number of quadruples concerning unseen entities in $\mathbb{T}_{\text{meta-train}}$, $\mathbb{T}_{\text{meta-valid}}$, $\mathbb{T}_{\text{meta-test}}$, respectively.

5.2 Baseline Methods

We take four types of methods as our baselines. First we consider two traditional KGC methods, i.e., ComplEx [Trouillon et al., 2016] and BiQUE [Guo and Kok, 2021a]. Then we consider several traditional TKGC methods, i.e., TNTComplEx [Lacroix et al., 2020], TeLM [Xu et al., 2021a], and TeRo [Xu et al., 2020a]. We combine all the quadruples in the background graph $\mathcal{G}_{\text{back}}$ with the quadruples of the meta-training set to construct a training set for traditional KGC as well as TKGC methods, and let them evaluate on all the query quadruples in the meta-validation/meta-test set. We also include two inductive KGC methods for OOG link prediction that do not employ meta-learning framework, i.e., MEAN [Hamaguchi et al., 2017], LAN [Wang et al., 2019b]. To achieve fair comparison, we only allow them to utilize support quadruples during inference, rather than an auxiliary set containing a large number of quadruples for each unseen entity $e' \in \{\mathcal{E}'_{\text{meta-valid}}, \mathcal{E}'_{\text{meta-test}}\}$. Apart from the first three types of methods, we further consider a meta-learning-based method GEN [Baek et al., 2020] which deals with few-shot OOG link prediction on static KGs. For the baseline methods designed for static KGs, we provide them with all the quadruples in our datasets and neglect time constraints, i.e., neglecting t in (s, r, o, t). We ensure that all the methods evaluate exactly the same quadruples.

5.3 Experimental Results

We report the TKG 1-shot and 3-shot OOG link prediction results in Table 2. We use mean reciprocal rank (MRR) and Hits@1/3/10 as the evaluation metrics (definition in Appendix B). We follow the filtered setting [Bordes et al., 2013] for fairer evaluation. We observe that traditional KGC and TKGC

methods show inferior performance in predicting the links concerning unseen entities. This is due to their nature that they have no way to transfer knowledge from seen to unseen entities. Besides, they learn representations of seen entities with a large number of associated training examples, thus causing the learned representations more prone to the data concerning seen entities and failing to embed unseen entities inductively. We also observe that inductive learning methods for static KGs show degenerated performance. MEAN, LAN, heavily rely on the auxiliary set during inference. We constrain their auxiliary set to only include the support quadruples, where only 1 associated quadruple for each unseen entity is included in the 1-shot case (3 associated quadruples in the 3-shot case). Experimental results show that these methods cannot effectively deal with newly-emerged entities that have only few observed edges, which is common in real-world scenarios. GEN employs meta-learning during training, thus having the ability to alleviate the data sparsity problem. However, it has no component to model temporal information, and it also does not incorporate any additional information, e.g., textual information and concept-aware information. To this end, GEN underperforms FILT in both 1-shot and 3-shot cases. Another crucial point worth noting is that the margin between FILT and GEN is much larger in the 3-shot case than in the 1-shot case. We attribute this to our time difference-based graph encoder. Our encoder distinguishes the importance of multiple support quadruples and aggregates the temporal neighbors more effectively.

Datasets	**ICEWS14-OOG**								**ICEWS18-OOG**								**ICEWS0515-OOG**							
	MRR		H@1		H@3		H@10		MRR		H@1		H@3		H@10		MRR		H@1		H@3		H@10	
Model	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S
ComplEx	.048	.046	.018	.014	.045	.046	.099	.089	.039	.044	.031	.026	.048	.042	.085	.093	.077	.076	.045	.048	.074	.071	.129	.120
BiQUE	.039	.035	.015	.014	.041	.030	.073	.066	.029	.032	.022	.021	.033	.037	.064	.073	.075	.083	.044	.049	.072	.077	.130	.144
TNTComplEx	.043	.044	.015	.016	.033	.042	.102	.096	.046	.048	.023	.026	.043	.044	.087	.082	.034	.037	.014	.012	.031	.036	.060	.071
TeLM	.032	.035	.012	.009	.021	.023	.063	.077	.049	.019	.029	.001	.045	.013	.084	.054	.080	.072	.041	.034	.077	.072	.138	.151
TeRo	.009	.010	.002	.002	.005	.002	.015	.020	.007	.006	.003	.001	.006	.003	.013	.006	.012	.023	.000	.010	.008	.017	.024	.040
MEAN	.035	.144	.013	.054	.032	.145	.082	.339	.016	.101	.003	.014	.012	.114	.043	.283	.019	.148	.003	.039	.017	.175	.052	.384
LAN	.168	.199	.050	.061	.199	.255	**.421**	**.500**	.077	.127	.018	.025	.067	.165	.199	.344	.171	.182	.081	.068	.180	.191	.367	.467
GEN	.231	.234	.162	.155	.250	.284	.378	.389	.171	.216	.112	.137	.189	.252	.289	.351	.268	.322	.185	.231	**.308**	.362	**.413**	.507
FILT	**.278**	**.321**	**.208**	**.240**	**.305**	**.357**	.410	.475	**.191**	**.266**	**.129**	**.187**	**.209**	**.298**	**.316**	**.417**	**.273**	**.370**	**.201**	**.299**	.303	**.391**	.405	**.516**

Table 2: TKG 1-shot and 3-shot OOG link prediction results. Evaluation metrics are filtered MRR and Hits@1/3/10 (H@1/3/10). The best results are marked in bold.

5.4 Ablation Study

To prove the effectiveness of the model components, we conduct several ablation studies on ICEWS14-OOG and ICEWS18-OOG. We devise model variants in the following way. **(A) Concept Modeling Variants:** In A1 we run our model without the concept modeling component. In A2, we delete the lower branch connecting the concept modeling component with the output of the graph encoder. In A3, we delete the upper branch connecting the concept modeling component with the input of the graph encoder. **(B) Graph Encoder Variants:** In B1, we neglect the time information and switch our graph encoder to RGCN [Schlichtkrull et al., 2018]. In B2, we use Time2Vec [Kazemi et al., 2019] to model temporal information. In B3, we employ the functional time encoder introduced in [Xu et al., 2020b] as our graph encoder. In B4, we derive a time-aware attentional network as our graph encoder: $\mathbf{h}_{(e',t_q)} = \sum_{(\tilde{e}_i,r_i,t_i)\in\mathcal{N}_{e'}} \gamma_q^i \mathbf{W}_g(\mathbf{h}_{\tilde{e}_i} \| \mathbf{h}_{r_i})$, where $\gamma_q^i = \frac{\exp\big(\sigma\big(([\mathbf{h}_{r_q}\|\Phi(t_q)]\mathbf{W}_Q)^\top([\mathbf{h}_{r_i}\|\Phi(t_i)]\mathbf{W}_K)\big)\big)}{\sum_{(\tilde{e}_j,r_j,t_j)\in\mathcal{N}_{e'}}\exp\big(\sigma\big(([\mathbf{h}_{r_q}\|\Phi(t_q)]\mathbf{W}_Q)^\top([\mathbf{h}_{r_j}\|\Phi(t_j)]\mathbf{W}_K)\big)\big)}$. Φ denotes the functional time encoder proposed in [Xu et al., 2020b] and $\mathbf{W}_Q$, $\mathbf{W}_K$ are two weight matrices.

We report the experimental results of the ablation studies in Table 3. From A1 to A3, we show that our concept modeling component helps to improve model performance, and it benefits from its double branch structure. From B1, we find that incorporating temporal information into the graph encoder is important for modeling TKGs. Besides, B2 to B4 show that in TKG few-shot OOG link prediction, it is not necessary to employ a complicated graph encoding structure. A possible reason is that we can only observe K (1 or 3) associated quadruples for every unseen entity, and this forms a tiny neighborhood. Complicated structures, e.g., our time-aware attentional network, are unable to demonstrate their superiority in this case.

Datasets	**ICEWS14-OOG**						**ICEWS18-OOG**					
	MRR		H@1		H@10		MRR		H@1		H@10	
Model	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S
A1	.267	.302	.195	.220	.407	.462	.187	.261	.128	.181	.315	.408
A2	.271	.285	.203	.217	.403	.454	.188	.265	.129	.187	.316	.411
A3	.276	.306	.206	.235	.401	.471	.189	.265	.125	.185	.316	.415
B1	.243	.256	.171	.179	.361	.402	.184	.238	.122	.162	.314	.383
B2	.258	.281	.181	.196	.393	.432	.185	.240	.119	.165	.316	.388
B3	.249	.278	.177	.179	.389	.438	.183	.242	.116	.166	.314	.395
B4	.263	.284	.192	.195	.400	.450	.181	.245	.112	.174	.307	.393
FILT	**.278**	**.321**	**.208**	**.240**	**.410**	**.475**	**.191**	**.266**	**.129**	**.187**	**.316**	**.417**

Table 3: Ablation studies of FILT on ICEWS14-OOG and ICEWS18-OOG. H@1/3/10 denote Hits@1/3/10, respectively. The best results are marked in bold.

5.5 Further Analysis

Cross shot analysis. We evaluate our trained 3-shot and 1-shot models with varying shots (1,3 or 5-shot) during meta-test. We observe in Table 4 that for both trained models, the performance increases as the test shot size rises. This is due to the effectiveness of our time-aware graph encoder. It distinguishes the importance of different support quadruples and better incorporates graph information as the shot size increases. We also observe that when the test shot size is larger than 3, FILT trained with 3 shots performs better than it trained with 1 shot. This is because during 3-shot meta-training, we simulate that for every unseen entity, 3 support examples are observable, which helps the model to generalize to the cases where their shot sizes are larger than 1 during meta-test. Besides, test with random shots does not greatly affect our model performance, thus showing FILT's robustness.

Datasets	**ICEWS14-OOG**						**ICEWS18-OOG**						**ICEWS0515-OOG**					
	(Train) 1-shot			**(Train) 3-shot**			**(Train) 1-shot**			**(Train) 3-shot**			**(Train) 1-shot**			**(Train) 3-shot**		
Test Shots	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10	MRR	H@1	H@10
1-shot	.278	.208	.410	.265	.195	.386	.191	.129	.316	.178	.117	.305	.273	.201	.405	.258	.184	.399
3-shot	.293	.212	.452	.321	.240	.475	.232	.158	.381	.266	.187	.417	.331	.254	.482	.370	.299	.516
5-shot	.297	.212	.467	.322	.231	.503	.256	.183	.400	.289	.206	.449	.351	.275	.499	.394	.317	.553
R-shot	.283	.203	.440	.299	.214	.462	.224	.154	.364	.242	.167	.390	.315	.240	.460	.337	.262	.490

Table 4: Cross shot analysis results. R-shot denotes the setting that we randomly sample 1, 3 or 5 support quadruples for every unseen entity during meta-test. H@1/3/10 denote Hits@1/3/10, respectively.

Visualization of concept representations. We plot the trained concept representations of the 3-shot model on ICEWS18-OOG with t-SNE [Van der Maaten and Hinton, 2008]. The entity concepts in the ICEWS database are hierarchical. For example, under the concept *Government*, there exist other concepts, e.g., *Foreign Ministry*. We only create labels for the first hierarchy concepts and assign other concepts belonging to them with the same label. From Figure 3, we can observe that the concepts bearing the same label tend to form a cluster, and the clusters having similar semantic meanings tend to be close to each other, e.g., the clusters of *Parties* and *Government*. This demonstrates that our concept modeling component learns the semantics of entity concepts, which helps to improve inductive learning for unseen new entities. We present three case studies in Appendix G.

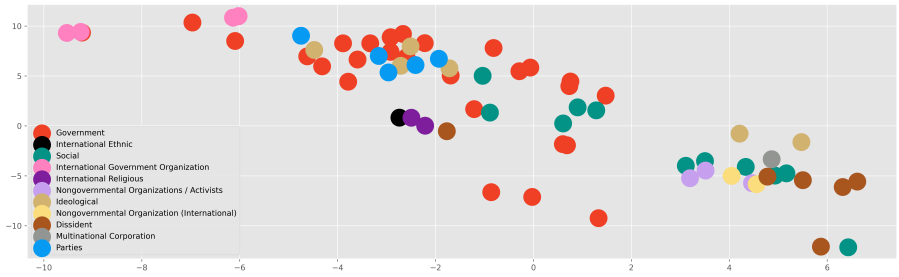


Figure 3: Visualization of learned concept representations on 3-shot ICEWS18-OOG.

6. Conclusion

We propose a new task: temporal knowledge graph (TKG) few-shot out-of-graph (OOG) link prediction, aiming to introduce the inductive entity representation learning problem into TKGs. We develop a model that focuses on the **f**ew-shot **i**nductive **l**earning on **T**KGs (FILT). Given only few edges associated to each newly-emerged entity, FILT employs a meta-learning framework that enables inductive knowledge transfer from seen entities to new unseen entities. FILT uses a time-aware graph encoder to learn the contextualized representations of unseen entities, which shows stronger performance as the shot size increases. It also utilizes the external entity concept information specified in the temporal knowledge bases. We propose three new datasets for TKG few-shot OOG link prediction and compare FILT with several baselines. Experimental results show that learning concept-aware information improves inductive learning for emerging entities. In the future, we would like to generalize rule-based knowledge graph reasoning methods to the TKG inductive learning scenario. Another direction is to combine future link prediction with our proposed TKG few-shot OOG link prediction task since our task currently does not support link forecasting.

References

Jinheon Baek, Dong Bok Lee, and Sung Ju Hwang. Learning to extrapolate knowledge: Transductive few-shot out-of-graph link prediction. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020,*

December 6-12, 2020, virtual, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/0663a4ddceacb40b095eda264a85f15c-Abstract.html.

Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795, 2013. URL https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. ICEWS Coded Event Data, 2015. URL https://doi.org/10.7910/DVN/28075.

Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. Meta relational learning for few-shot link prediction in knowledge graphs. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *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 4216–4225. Association for Computational Linguistics, 2019. doi: 10.18653/v1/D19-1431. URL https://doi.org/10.18653/v1/D19-1431.

Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 6572–6583, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/69386f6bb1dfed68692a24c8686939b9-Abstract.html.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL, 2014. doi: 10.3115/v1/d14-1179. URL https://doi.org/10.3115/v1/d14-1179.

Zifeng Ding, Yunpu Ma, Bailan He, and Volker Tresp. A simple but powerful graph encoder for temporal knowledge graph completion. *CoRR*, abs/2112.07791, 2021. URL https://arxiv.org/abs/2112.07791.

Zifeng Ding, Bailan He, Yunpu Ma, Zhen Han, and Volker Tresp. Learning meta representations of one-shot relations for temporal knowledge graph link prediction. *CoRR*, abs/2205.10621, 2022a. doi: 10.48550/arXiv.2205.10621. URL https://doi.org/10.48550/arXiv.2205.10621.

Zifeng Ding, Ruoxia Qi, Zongyue Li, Bailan He, Jingpei Wu, Yunpu Ma, Zhao Meng, Zhen Han, and Volker Tresp. Forecasting question answering over temporal knowledge graphs. *CoRR*, abs/2208.06501, 2022b. doi: 10.48550/arXiv.2208.06501. URL https://doi.org/10.48550/arXiv.2208.06501.

Jia Guo and Stanley Kok. Bique: Biquaternionic embeddings of knowledge graphs. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8338–8351. Association for Computational Linguistics, 2021a. doi: 10.18653/v1/2021.emnlp-main.657. URL https://doi.org/10.18653/v1/2021.emnlp-main.657.

Jia Guo and Stanley Kok. BiQUE: Biquaternionic embeddings of knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8338–8351, Online and Punta Cana, Dominican Republic, November 2021b. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.657. URL https://aclanthology.org/2021.emnlp-main.657.

Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. Knowledge transfer for out-of-knowledge-base entities : A graph neural network approach. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1802–1808. ijcai.org, 2017. doi: 10.24963/ijcai.2017/250. URL https://doi.org/10.24963/ijcai.2017/250.

Zhen Han, Yunpu Ma, Peng Chen, and Volker Tresp. Dyernie: Dynamic evolution of riemannian manifold embeddings for temporal knowledge graph completion. *arXiv preprint arXiv:2011.03984*, 2020a.

Zhen Han, Yunpu Ma, Yuyi Wang, Stephan Günnemann, and Volker Tresp. Graph hawkes neural network for forecasting on temporal knowledge graphs. *arXiv preprint arXiv:2003.13432*, 2020b.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021a. URL https://openreview.net/forum?id=pGIHq1m7PU.

Zhen Han, Zifeng Ding, Yunpu Ma, Yujia Gu, and Volker Tresp. Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8352–8364. Association for Computational Linguistics, 2021b. doi: 10.18653/v1/2021.emnlp-main.658. URL https://doi.org/10.18653/v1/2021.emnlp-main.658.

Zhen Han, Gengyuan Zhang, Yunpu Ma, and Volker Tresp. Time-dependent entity embedding is not all you need: A re-evaluation of temporal knowledge graph completion models under a unified framework. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8104–8118, Online and Punta Cana, Dominican Republic, November

2021c. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.639. URL https://aclanthology.org/2021.emnlp-main.639.

Yongquan He, Zhihan Wang, Peng Zhang, Zhaopeng Tu, and Zhaochun Ren. VN network: Embedding newly emerging entities with virtual neighbors. In Mathieu d'Aquin, Stefan Dietze, Claudia Hauff, Edward Curry, and Philippe Cudré-Mauroux, editors, *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, pages 505–514. ACM, 2020. doi: 10.1145/3340531.3411865. URL https://doi.org/10.1145/3340531.3411865.

Jaehun Jung, Jinhong Jung, and U Kang. Learning to walk across time for interpretable temporal knowledge graph completion. In Feida Zhu, Beng Chin Ooi, and Chunyan Miao, editors, *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 786–795. ACM, 2021. doi: 10.1145/3447548.3467292. URL https://doi.org/10.1145/3447548.3467292.

Seyed Mehran Kazemi, Rishab Goel, Sepehr Eghbali, Janahan Ramanan, Jaspreet Sahota, Sanjay Thakur, Stella Wu, Cathal Smyth, Pascal Poupart, and Marcus A. Brubaker. Time2vec: Learning a vector representation of time. *CoRR*, abs/1907.05321, 2019. URL http://arxiv.org/abs/1907.05321.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. Tensor decompositions for temporal knowledge base completion. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=rke2P1BFwS.

Julien Leblay and Melisachew Wudage Chekol. Deriving validity time in knowledge graph. In Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis, editors, *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1771–1776. ACM, 2018. doi: 10.1145/3184558.3191639. URL https://doi.org/10.1145/3184558.3191639.

Wei Li, Ruihan Bao, Keiko Harimoto, Deli Chen, Jingjing Xu, and Qi Su. Modeling the stock relation with graph network for overnight stock movement prediction. In Christian Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 4541–4547. ijcai.org, 2020. doi: 10.24963/ijcai.2020/626. URL https://doi.org/10.24963/ijcai.2020/626.

Yunpu Ma, Volker Tresp, and Erik A. Daxberger. Embedding models for episodic knowledge graphs. *J. Web Semant.*, 59, 2019.

Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. Distant supervision for relation extraction with an incomplete knowledge base. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, editors, *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 777–782. The Association for Computational Linguistics, 2013. URL https://aclanthology.org/N13-1095/.

Mehrnoosh Mirtaheri, Mohammad Rostami, Xiang Ren, Fred Morstatter, and Aram Galstyan. One-shot learning for temporal knowledge graphs. In *3rd Conference on Automated Knowledge Base Construction*, 2021.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 809–816. Omnipress, 2011. URL https://icml.cc/2011/papers/438_icmlpaper.pdf.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

Ali Sadeghian, Mohammadreza Armandpour, Anthony Colas, and Daisy Zhe Wang. Chronor: Rotation based temporal knowledge graph embedding. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6471–6479. AAAI Press, 2021. URL https://ojs.aaai.org/index.php/AAAI/article/view/16802.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4498–4507, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.412. URL https://aclanthology.org/2020.acl-main.412.

Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam, editors, *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer, 2018. doi: 10.1007/978-3-319-93417-4_38. URL https://doi.org/10.1007/978-3-319-93417-4_38.

Jiawei Sheng, Shu Guo, Zhenyu Chen, Juwei Yue, Lihong Wang, Tingwen Liu, and Hongbo Xu. Adaptive attentional network for few-shot knowledge graph completion. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1681–1691. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.131. URL https://doi.org/10.18653/v1/2020.emnlp-main.131.

Baoxu Shi and Tim Weninger. Open-world knowledge graph completion. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1957–1964. AAAI Press, 2018. URL https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16055.

Haohai Sun, Jialun Zhong, Yunpu Ma, Zhen Han, and Kun He. TimeTraveler: Reinforcement learning for temporal knowledge graph forecasting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8306–8319, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.655. URL https://aclanthology.org/2021.emnlp-main.655.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL https://openreview.net/forum?id=HkgEQnRqYQ.

Volker Tresp, Cristóbal Esteban, Yinchong Yang, Stephan Baier, and Denis Krompaß. Learning with memory embeddings. *arXiv preprint arXiv:1511.07972*, 2015.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org, 2016. URL http://proceedings.mlr.press/v48/trouillon16.html.

Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. Composition-based multi-relational graph convolutional networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL https://openreview.net/forum?id=BylA_C4tPr.

Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. Knowledge graph convolutional networks for recommender systems. In Ling Liu, Ryen W. White, Amin Mantrach, Fabrizio Silvestri, Julian J. McAuley, Ricardo Baeza-Yates, and Leila Zia, editors, *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*, pages 3307–3313. ACM, 2019a. doi: 10.1145/3308558.3313417. URL https://doi.org/10.1145/3308558.3313417.

Peifeng Wang, Jialong Han, Chenliang Li, and Rong Pan. Logic attention based neighborhood aggregation for inductive knowledge graph embedding. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February*

1, 2019, pages 7152–7159. AAAI Press, 2019b. doi: 10.1609/aaai.v33i01.33017152. URL https://doi.org/10.1609/aaai.v33i01.33017152.

Xiang Wang, Dingxian Wang, Canran Xu, Xiangnan He, Yixin Cao, and Tat-Seng Chua. Explainable reasoning over knowledge graphs for recommendation. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 5329–5336. AAAI Press, 2019c. doi: 10.1609/aaai.v33i01.33015329. URL https://doi.org/10.1609/aaai.v33i01.33015329.

Zihao Wang, Kwun Ping Lai, Piji Li, Lidong Bing, and Wai Lam. Tackling long-tailed relations and uncommon entities in knowledge graph completion. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *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 250–260. Association for Computational Linguistics, 2019d. doi: 10.18653/v1/D19-1024. URL https://doi.org/10.18653/v1/D19-1024.

Jiapeng Wu, Meng Cao, Jackie Chi Kit Cheung, and William L. Hamilton. Temp: Temporal message passing for temporal knowledge graph completion. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5730–5746. Association for Computational Linguistics, 2020. doi: 10.18653/v1/2020.emnlp-main.462. URL https://doi.org/10.18653/v1/2020.emnlp-main.462.

Ruobing Xie, Zhiyuan Liu, Jia Jia, Huanbo Luan, and Maosong Sun. Representation learning of knowledge graphs with entity descriptions. In Dale Schuurmans and Michael P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2659–2665. AAAI Press, 2016. URL http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12216.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. One-shot relational learning for knowledge graphs. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1980–1990. Association for Computational Linguistics, 2018. doi: 10.18653/v1/d18-1223. URL https://doi.org/10.18653/v1/d18-1223.

Chengjin Xu, Mojtaba Nayyeri, Fouad Alkhoury, Hamed Shariat Yazdi, and Jens Lehmann. Tero: A time-aware knowledge graph embedding via temporal rotation. In Donia Scott, Núria Bel, and Chengqing Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020*, pages 1583–1593. International Committee on Computational Linguistics, 2020a. doi: 10.18653/v1/2020.coling-main.139. URL https://doi.org/10.18653/v1/2020.coling-main.139.

Chengjin Xu, Yung-Yu Chen, Mojtaba Nayyeri, and Jens Lehmann. Temporal knowledge graph completion using a linear temporal regularizer and multivector embeddings. In Kristina Toutanova,

Anna Rumshisky, Luke Zettlemoyer, Dilek Hakkani-Tür, Iz Beltagy, Steven Bethard, Ryan Cotterell, Tanmoy Chakraborty, and Yichao Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 2569–2578. Association for Computational Linguistics, 2021a. doi: 10.18653/v1/2021.naacl-main.202. URL https://doi.org/10.18653/v1/2021.naacl-main.202.

Da Xu, Chuanwei Ruan, Evren Körpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020b. URL https://openreview.net/forum?id=rJeW1yHYwH.

Wentao Xu, Weiqing Liu, Lewen Wang, Yingce Xia, Jiang Bian, Jian Yin, and Tie-Yan Liu. HIST: A graph-based framework for stock trend forecasting via mining concept-oriented shared information. *CoRR*, abs/2110.13716, 2021b. URL https://arxiv.org/abs/2110.13716.

Donghan Yu, Yiming Yang, Ruohong Zhang, and Yuexin Wu. Knowledge embedding based graph convolutional network. In Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia, editors, *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 1619–1628. ACM / IW3C2, 2021. doi: 10.1145/3442381.3449925. URL https://doi.org/10.1145/3442381.3449925.

Yufeng Zhang, Weiqing Wang, Wei Chen, Jiajie Xu, An Liu, and Lei Zhao. Meta-learning based hyper-relation feature modeling for out-of-knowledge-base embedding. In Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong, editors, *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 2637–2646. ACM, 2021. doi: 10.1145/3459637.3482367. URL https://doi.org/10.1145/3459637.3482367.

Appendix A. Long-Tail Distribution of Entities in Temporal Knowledge Bases

Figure A illustrates the entity occurrence of ICEWS14, ICEWS18 and ICEWS05-15 databases. We find that most entities occur for only a few times.

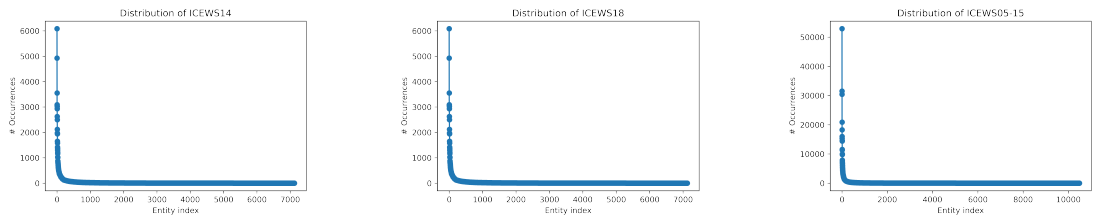


Figure 4: Entity occurrence of ICEWS14, ICEWS18 and ICEWS05-15 databases.

Appendix B. Evaluation Metrics

We use two evaluation metrics for our experiments, i.e., mean reciprocal rank (MRR) and Hits@1/3/10. For every link prediction query, we compute the rank ψ of the ground truth missing entity. MRR is defined as: $\frac{1}{\sum_{e' \in \mathcal{E}'_{\text{meta-test}}} |\mathcal{Q}_{e'}|} \sum_{e' \in \mathcal{E}'_{\text{meta-test}}} \sum_{q^+ \in \mathcal{Q}_{e'}} \frac{1}{\psi}$. Hits@1/3/10 denote the proportions of the predicted links where ground truth missing entities are ranked as top 1, top3, top10, respectively.

Appendix C. Implementation Details

We implement all the experiments with PyTorch [Paszke et al., 2019] on a single NVIDIA Tesla T4. We search hyperparameters following Table 5. For each dataset, we do 108 trials to try different hyperparameter settings. We run 15000 batches for each trail and compare their meta-validation results. We choose the setting leading to the best meta-validation result and take it as the best hyperparameter setting. We report the best hyperparameter setting in Table 6. Every result of our model is the average result of five runs. For the models leading to the results reported in Table 2, we provide their meta-validation results in Table 7. We also specify their GPU memory usage (Table 8) and number of parameters (Table 9). For different datasets, we use different numbers of unseen entities N in each meta-training task T. We set $N = 100$ for ICEWS14-OOG and ICEWS0515-OOG, $N = 200$ for ICEWS18-OOG. We sample 32 negative samples for every positive sample.

Hyperparameter	**Search Space**
Embedding Size	{50, 100, 200}
# Aggregation Step	{1, 2}
Activation Function	{Tanh, ReLU, LeakyReLU}
Dropout	{0.2, 0.3, 0.5}
λ	{0.2, 0.4}

Table 5: Hyperparameter searching strategy.

Datasets	**ICEWS14-OOG**	**ICEWS18-OOG**	**ICEWS0515-OOG**
Hyperparameter			
Embedding Size	100	100	100
# Aggregation Step	1	1	1
Activation Function	LeakyReLU	LeakyReLU	LeakyReLU
Dropout	0.3	0.3	0.3
λ	0.2	0.4	0.4

Table 6: Best hyperparameter settings.

Datasets	**ICEWS14-OOG**								**ICEWS18-OOG**								**ICEWS0515-OOG**							
	MRR		H@1		H@3		H@10		MRR		H@1		H@3		H@10		MRR		H@1		H@3		H@10	
Model	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S
FILT	.251	.354	.171	.271	.285	.389	.410	.511	.187	.242	.127	.163	.204	.264	.308	.406	.232	.316	.163	.229	.247	.350	.378	.491

Table 7: TKG 1-shot and 3-shot OOG link prediction results on the meta-validation set. Evaluation metrics are filtered MRR and Hits@1/3/10 (H@1/3/10).

Datasets	**ICEWS14-OOG**		**ICEWS18-OOG**		**ICEWS0515-OOG**	
	GPU Memory		GPU Memory		GPU Memory	
Model	1-S	3-S	1-S	3-S	1-S	3-S
FILT	1493MB	1466MB	1871MB	1841MB	1557MB	1541MB

Table 8: GPU memory usage.

Datasets	**ICEWS14-OOG**		**ICEWS18-OOG**		**ICEWS0515-OOG**	
	# Param		# Param		# Param	
Model	1-S	3-S	1-S	3-S	1-S	3-S
FILT	2966303	2966303	4567203	4567203	3310703	3310703

Table 9: Number of parameters.

For baseline methods, except MEAN, we use their official implementations, i.e., ComplEx[2], BiQUE[3], TNTComplEx[4], TeLM[5], TeRo[6], LAN[7], GEN[8]. We use the MEAN implementation provided in the LAN repository. We use default hyperparameters of TKGC methods for ICEWS datasets. For other methods, we keep their embedding size the same as FILT's. We keep other hyperparameters of them as their default settings.

Appendix D. Further Discussion of TKG Few-Shot OOG Link Prediction

Figure 5 illustrates how we formulate TKG few-shot OOG link prediction into a meta-learning problem with an example. Green edges correspond to the support quadruples and orange edges correspond to the query quadruples (timestamps and relations are omitted for brevity). The meta-training process consists of a number of meta-training tasks. During each meta-training task T, N unseen entities from $\mathcal{E}'_{\text{meta-train}}$ are randomly sampled. In Figure 5, $e'_1, e'_2 \in \mathcal{E}'_{\text{meta-train}}$ are sampled in task T. For each sampled unseen entity, K ($K = 1$ in Figure 5) quadruples from all the quadruples containing itself are sampled to form its support set. The rest form its query set. During meta-validation, all the unseen entities (e'_5, e'_6, e'_7, e'_8) from $\mathcal{E}'_{\text{meta-valid}}$ are treated as appearing simultaneously, which also applies to meta-test and the unseen entities $(e'_9, e'_{10}, e'_{11}, e'_{12})$ from $\mathcal{E}'_{\text{meta-test}}$.

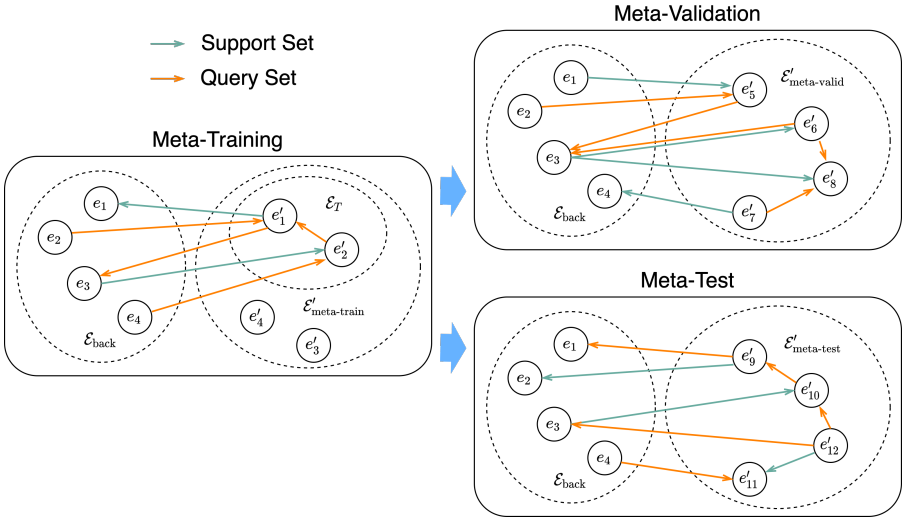


Figure 5: Illustration of the meta-learning framework formulated from the TKG few-shot OOG link prediction task.

TKG few-shot OOG link prediction vs. TKG completion. For the existing TKGC benchmark datasets, e.g., ICEWS14[9], there exist a number of entities that only appear in the test sets (or the

2. https://github.com/ttrouill/complex
3. https://github.com/guojiapub/BiQUE
4. https://github.com/facebookresearch/tkbc
5. https://github.com/soledad921/TeLM
6. https://github.com/soledad921/ATISE
7. https://github.com/wangpf3/LAN
8. https://github.com/JinheonBaek/GEN
9. https://github.com/BorealisAI/de-simple

validation sets) and are unseen in their training sets. Evaluating on the links concerning these unseen entities coincides to the evaluation setting of TKG few-shot OOG link prediction. However, in our proposed task, we focus on the unseen entities that are long-tail, and we also introduce a realistic setting that each unseen entity is coupled with K support quadruples containing itself, while in traditional TKGC benchmark datasets the unseen entities are not guaranteed to be long-tail and no associated edge is given for learning the inductive representations of them. The aim of TKG few-shot OOG link prediction is to ask the TKG reasoning models to learn strong representations of the unseen entities inductively from extracting the information from the provided K support quadruples, which corresponds to the realistic situation where every newly-emerged entity is often coupled with a small number of associated edges.

Appendix E. Ablation Study Details

We present the detailed equations of graph encoder variants (B1-B3 in Table 3, B4 already presented). In B1, RGCN computes the unseen entity e''s representation as:

$$\mathbf{h}_{(e',t_q)} = \frac{1}{|\mathcal{N}_{e'}|} \sum_{(\tilde{e}_i,r_i,t_i)\in\mathcal{N}_{e'}} \mathbf{W}_{r_i}(\mathbf{h}_{\tilde{e}_i}), \tag{6}$$

where $\mathbf{W}_{r_i}$ is a weight matrix modeling r_i. In B2, Time2Vec computes e''s representation as:

$$\mathbf{h}_{(e',t_q)} = \frac{1}{|\mathcal{N}_{e'}|} \sum_{(\tilde{e}_i,r_i,t_i)\in\mathcal{N}_{e'}} \mathbf{W}_g(\mathbf{h}_{(\tilde{e}_i,t_i)} \| \mathbf{h}_{r_i}), \tag{7}$$

where $\mathbf{h}_{(\tilde{e}_i,t_i)}$ is defined as:

$$\begin{aligned} \mathbf{h}_{(\tilde{e}_i,t_i)} &= f(\mathbf{h}_{\tilde{e}_i} \| \Phi(t_i)), \\ \Phi(t_i)[j] &= \begin{cases} \omega_j t_i + \varphi_j, & \text{if } j = 0, \\ \sin(\omega_j t_i + \varphi_j), & \text{if } 1 \le j \le d_t. \end{cases} \end{aligned} \tag{8}$$

f denotes a layer of feed forward neural network. $\Phi(t_i)[j]$ denotes the jth component of t_i's time representation $\Phi(t_i)$. d_t is the dimension size of time representations. ω_j and φ_j represent the trainable frequency and phase parameters, respectively. In B3, we use the same aggregation function 7 as in Time2vec, however, we use another form of time encoder to encode time information:

$$\begin{aligned} \mathbf{h}_{(\tilde{e}_i,t_i)} &= f(\mathbf{h}_{\tilde{e}_i} \| \Phi(t_i)), \\ \Phi(t_i) &= \sqrt{\frac{1}{d_t}} [\cos(\omega_1 t_i + \varphi_1), \ldots, \cos(\omega_{d_t} t_i + \varphi_{d_t}))], \end{aligned} \tag{9}$$

where $\omega_1 \ldots \omega_{d_t}$ and $\phi_1 \ldots \phi_{d_t}$ are trainable parameters.

Appendix F. Concept Extraction of ICEWS Database

We take the sectors of ICEWS entities as their concepts. The sector classification can be found on the ICEWS official website[10]. ICEWS sectors have hierarchies. We do not consider hierarchies and

10. https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/28118

consider each sector as individual. For example, the sector *Foreign Ministry* belongs to the sector *Government*. We learn their representations separately.

A number of entities in the ICEWS coded event data are not labeled with any sector. Some of them are regions, e.g., *North Korea*. We create a new sector named *Region* for them. For other entities, we find their affiliations and pick out their sectors. We then choose from their affiliations' sectors the most suitable ones and label these entities. For example, *European Parliament* has no associated sector in the ICEWS coded event data. We find its affiliation *European Union*. *European Union* is assigned a sector *Regional Diplomatic IGOs*. We take *Regional Diplomatic IGOs* as *European Parliament*'s sector and it is taken as a concept in our meta-learning process.

Appendix G. Case Study of Learned Concept Representations

We further find three cases to show that our learned concept representations capture the semantic meaning of concepts, which helps to embed unseen entities inductively. We resize the visualization in Figure 3 and label several concepts close to each other.

The first case is about the concepts *Foreign Ministry*, *International Government Organization* and *Regional Diplomatic IGOs*, where *IGO* stands for international government organization. From human intuition, *Foreign Ministry* is closely related to international interactions. Similarly, *international Government Organization* and *Regional Diplomatic IGOs* also possess the same semantics.

The second case is about the concepts *International Ethnic*, *International Religious* and *Muslim*. *Muslim* stands for not only a religion but also an ethnicity, therefore, it is close to both *International Religious* and *International Ethnic*.

The third case is about the concepts *Medical / Health NGOs*, *Human Rights NGOs* and *Human Rights IGOs*, where *NGO* stands for nongovernmental organizations. We can observe that *Human Rights NGOs* and *Human Rights IGOs* are extremely close to each other. Since protecting human rights is normally concerned with providing medical aid, they are also close to *Medical / Health NGOs*.

Appendix H. Dataset Construction Process

1. We take ICEWS14[11], ICEWS18[12] and ICEWS05-15[13] as the databases for dataset construction.

2. We set the upper and lower thresholds for entity frequencies. We do not want the upper threshold to be large since in real-world scenarios, newly-emerged entities normally are only coupled with very few edges. We also do not want the lower threshold to be too small since we want to include enough test examples. We set the upper and lower threshold to 10 and 25 for every dataset.

3. We pick out the entities whose frequencies are between thresholds and sample half of them as the total unseen entities $\mathcal{E}'$ (following [Baek et al., 2020]). We take the quadruples without any unseen entity as the background graph $\mathcal{G}_{\text{back}}$.

11. https://github.com/BorealisAI/de-simple
12. https://github.com/INK-USC/RE-Net
13. https://github.com/mniepert/mmkb/tree/master/TemporalKGs

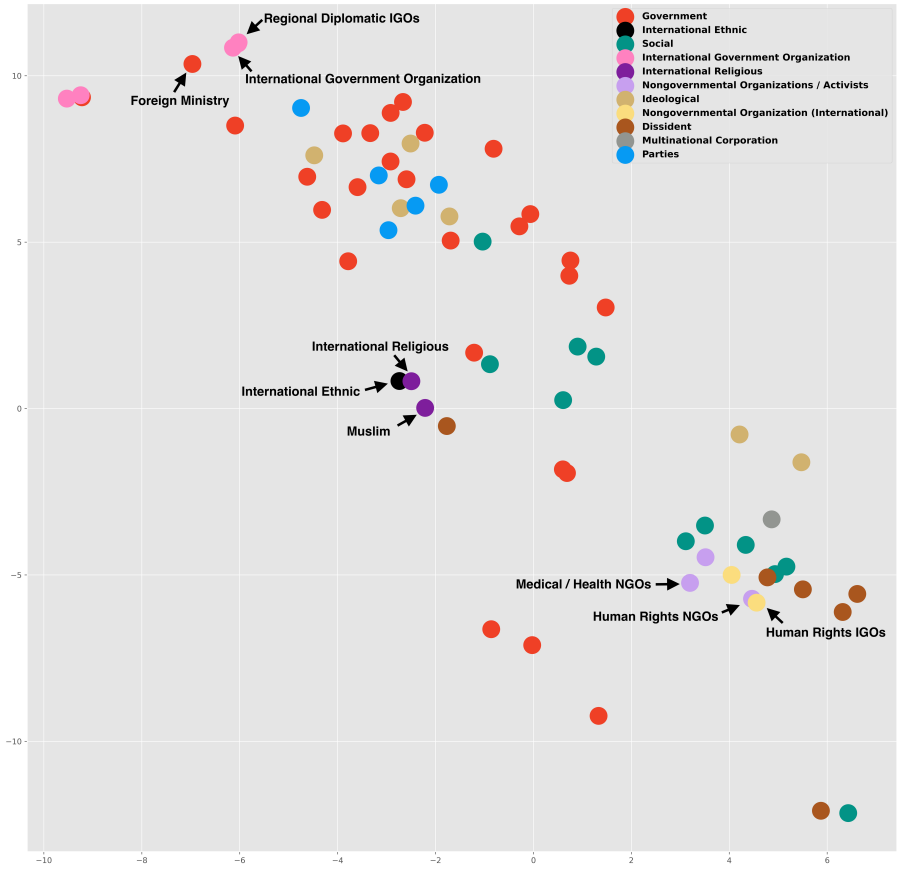


Figure 6: Resized visualization of learned concept representations on 3-shot ICEWS18-OOG.

4. We split the unseen entities as meta-training $\mathcal{E}'_{\text{meta-train}}$, meta-validation $\mathcal{E}'_{\text{meta-valid}}$ and meta-test $\mathcal{E}'_{\text{meta-test}}$ entities. $|\mathcal{E}'_{\text{meta-train}}| : |\mathcal{E}'_{\text{meta-valid}}| : |\mathcal{E}'_{\text{meta-test}}| \approx 8 : 1 : 1$. Their associated quadruples form the corresponding meta-learning sets.

Chapter 4

Improving Few-Shot Inductive Learning on Temporal Knowledge Graphs using Confidence-Augmented Reinforcement Learning

This chapter contains the publication

> **Zifeng Ding***, Jingpei Wu*, Zongyue Li, Yunpu Ma, and Volker Tresp. Improving Few-Shot Inductive Learning on Temporal Knowledge Graphs using Confidence-Augmented Reinforcement Learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2023. *Equal Contribution. DOI: 10.1007/978-3-031-43418-1_33

Improving Few-Shot Inductive Learning on Temporal Knowledge Graphs Using Confidence-Augmented Reinforcement Learning

Zifeng Ding[1,2], Jingpei Wu[1], Zongyue Li[1,3], Yunpu Ma[1,2], and Volker Tresp[1(✉)]

[1] LMU Munich, Geschwister-Scholl-Platz 1, 80539 Munich, Germany
zifeng.ding@campus.lmu.de, Volker.Tresp@lmu.de
[2] Siemens AG, Otto-Hahn-Ring 6, 81739 Munich, Germany
[3] Munich Center for Machine Learning (MCML), Munich, Germany

Abstract. Temporal knowledge graph completion (TKGC) aims to predict the missing links among the entities in a temporal knowledge graph (TKG). Most previous TKGC methods only consider predicting the missing links among the entities seen in the training set, while they are unable to achieve great performance in link prediction concerning newly-emerged unseen entities. Recently, a new task, i.e., TKG few-shot out-of-graph (OOG) link prediction, is proposed, where TKGC models are required to achieve great link prediction performance concerning newly-emerged entities that only have few-shot observed examples. In this work, we propose a TKGC method FITCARL that combines few-shot learning with reinforcement learning to solve this task. In FITCARL, an agent traverses through the whole TKG to search for the prediction answer. A policy network is designed to guide the search process based on the traversed path. To better address the data scarcity problem in the few-shot setting, we introduce a module that computes the confidence of each candidate action and integrate it into the policy for action selection. We also exploit the entity concept information with a novel concept regularizer to boost model performance. Experimental results show that FITCARL achieves stat-of-the-art performance on TKG few-shot OOG link prediction. Code and supplementary appendices are provided (https://github.com/ZifengDing/FITCARL/tree/main).

Keywords: Temporal knowledge graph · Few-shot learning

1 Introduction

Knowledge graphs (KGs) store knowledge by representing facts in the form of triples, i.e., (s, r, o), where s and o are the subject and object entities, and r denotes the relation between them. To further specify the time validity of the facts, temporal knowledge graphs (TKGs) are introduced by using a quadruple

Z. Ding and J. Wu—Equal contribution.

D. Koutra et al. (Eds.): ECML PKDD 2023, LNAI 14171, pp. 550–566, 2023.
https://doi.org/10.1007/978-3-031-43418-1_33

(s, r, o, t) to represent each fact, where t is the valid time of this fact. In this way, TKGs are able to capture the ever-evolving knowledge over time. It has already been extensively explored to use KGs and TKGs to assist downstream tasks, e.g., question answering [12,27,45] and natural language generation [2,20].

Since TKGs are known to be incomplete [19], a large number of researches focus on proposing methods to automatically complete TKGs, i.e., temporal knowledge graph completion (TKGC). In traditional TKGC, models are given a training set consisting of a TKG containing a finite set of entities during training, and they are required to predict the missing links among the entities seen in the training set. Most previous TKGC methods, e.g., [11,17,19,31], achieve great success on traditional TKGC, however, they still have drawbacks. (1) Due to the ever-evolving nature of world knowledge, new unseen entities always emerge in a TKG and traditional TKGC methods fail to handle them. (2) Besides, in real-world scenarios, newly-emerged entities are usually coupled with only a few associated edges [13]. Traditional TKGC methods require a large number of entity-related data examples to learn expressive entity representations, making them hard to optimally represent newly-emerged entities. To this end, recently, Ding et al. [13] propose the TKG few-shot out-of-graph (OOG) link prediction (LP) task based on traditional TKGC, aiming to draw attention to studying how to achieve better LP results regarding newly-emerged TKG entities.

In this work, we propose a TKGC method to improve **f**ew-shot **i**nductive learning over newly-emerged entities on **T**KGs using **c**onfidence-**a**ugmented **r**einforcement **l**earning (FITCARL). FITCARL is developed to solve TKG few-shot OOG LP [13]. It is a meta-learning based method trained with episodic training [36]. For each unseen entity, FITCARL first employs a time-aware Transformer [35] to adaptively learn its expressive representation. Then it starts from the unseen entity and sequentially takes actions by transferring to other entities according to the observed edges associated with the current entity, following a policy parameterized by a learnable policy network. FITCARL traverses the TKG for a fixed number of steps and stops at the entity that is expected to be the LP answer. To better address the data scarcity problem in the few-shot setting, we introduce a confidence learner that computes the confidence of each candidate action and integrate it into the policy for action selection. Following [13], we also take advantage of the concept information presented in the temporal knowledge bases (TKBs) and design a novel concept regularizer. We summarize our contributions as follows: (1) This is the first work using reinforcement learning-based method to reason over newly-emerged few-shot entities in TKGs and solve the TKG few-shot OOG LP task. (2) We propose a time-aware Transformer using a time-aware positional encoding method to better utilize few-shot information in learning representations of new-emerged entities. (3) We design a novel confidence learner to alleviate the negative impact of the data scarcity problem brought by the few-shot setting. (4) We propose a parameter-free concept regularizer to utilize the concept information provided by the TKBs and it demonstrates strong effectiveness. (5) FITCARL achieves state-of-the-art performance on all datasets of TKG few-shot OOG LP and provides explainability.

2 Related Work

2.1 Knowledge Graph and Temporal Knowledge Graph Completion

Knowledge graph completion (KGC) methods can be summarized into two types. The first type of methods focuses on designing KG score functions that directly compute the plausibility scores of KG triples [1,4,5,22,25,32,43]. Other KGC methods are neural-based models [28,34]. Neural-based models are built by coupling KG score functions with neural structures, e.g., graph neural network (GNN). It is shown that neural structures make great contributions to enhancing the performance of KGC methods. TKGC methods are developed by incorporating temporal reasoning techniques. A line of works aims to design time-aware KG score functions that are able to process time information [7,19,23,26,42,44]. Another line of works employs neural structures to encode temporal information, where some of them use recurrent neural structures, e.g., Transformer [35], to model the temporal dependencies in TKGs [39], and others design time-aware GNNs to achieve temporal reasoning by computing time-aware entity representations through aggregation [11,17]. Reinforcement learning (RL) has already been used to reason TKGs, e.g., [21,30]. TITer [30] and CluSTeR [21] achieve temporal path modeling with RL. However, they are traditional TKG reasoning models and are not designed to deal with few-shot unseen entities[1].

2.2 Inductive Learning on KGs and TKGs

In recent years, inductive learning on KGs and TKGs has gained increasing interest. A series of works [8,10,24,29,40] focuses on learning strong inductive representations of few-shot unseen relations using meta-learning-based approaches. These methods achieve great effectiveness, however, they are unable to deal with newly-emerged entities. Some works try to deal with unseen entities by inductively transferring knowledge from seen to unseen entities with an auxiliary set provided during inference [15,16,37]. Their performance highly depends on the size of the auxiliary set. [13] shows that with a tiny auxiliary set, these methods cannot achieve ideal performance. Besides, these methods are developed for static KGs, thus without temporal reasoning ability. On top of them, Baek et al. [3] propose a more realistic task, i.e., KG few-shot OOG LP, aiming to draw attention to better studying few-shot OOG entities. They propose a model GEN that contains two GNNs and train it with a meta-learning framework to adapt to the few-shot setting. Same as [15,16,37], GEN does not have a temporal reasoning module, and therefore, it cannot reason TKGs. Ding et al. [13] propose the TKG few-shot OOG LP task that generalizes [3] to the context of TKGs. They develop a meta-learning-based model FILT that achieves temporal reasoning with a time difference-based graph encoder and mines concept-aware

[1] TITer can model unseen entities, but it is not designed for few-shot setting and requires a substantial number of associated facts. Besides, both TITer and CluSTeR are TKG forecasting methods, where models are asked to predict future links given the past TKG information (different from TKGC, see Appendix B for discussion).

information from the entity concepts specified in TKBs. Recently, another work [38] proposes a task called few-shot TKG reasoning, aiming to ask TKG models to predict future facts for newly-emerged few-shot entities. In few-shot TKG reasoning, for each newly-emerged entity, TKG models are asked to predict the unobserved associated links happening after the observed few-shot examples. Such restriction is not imposed in TKG few-shot OOG LP, meaning that TKG models should predict the unobserved links happening at any time along the time axis. In our work, we only consider the task setting of TKG few-shot OOG LP and do not consider the setting of [38].

3 Task Formulation and Preliminaries

3.1 TKG Few-Shot Out-of-Graph Link Prediction

Definition 1 (TKG Few-Shot OOG LP). Assume we have a background TKG $\mathcal{G}_{\text{back}} = \{(s, r, o, t) | s, o \in \mathcal{E}_{\text{back}}, r \in \mathcal{R}, t \in \mathcal{T}\} \subseteq \mathcal{E}_{\text{back}} \times \mathcal{R} \times \mathcal{E}_{\text{back}} \times \mathcal{T}$, where $\mathcal{E}_{\text{back}}$, $\mathcal{R}$, $\mathcal{T}$ denote a finite set of seen entities, relations and timestamps, respectively. An unseen entity e' is an entity $e' \in \mathcal{E}'$ and $\mathcal{E}' \cap \mathcal{E}_{\text{back}} = \emptyset$. For each $e' \in \mathcal{E}'$, given K observed e' associated TKG facts $(e', r, \tilde{e}, t)$ (or $(\tilde{e}, r, e', t)$), where $\tilde{e} \in (\mathcal{E}_{\text{back}} \cup \mathcal{E}')$, $r \in \mathcal{R}$, $t \in \mathcal{T}$, TKG few-shot OOG LP asks models to predict the missing entities of LP queries $(e', r_q, ?, t_q)$ (or $(?, r_q, e', t_q)$) derived from unobserved TKG facts containing e' ($r_q \in \mathcal{R}$, $t_q \in \mathcal{T}$). K is a small number denoting shot size, e.g., 1 or 3.

Ding et al. [13] formulate TKG few-shot OOG LP into a meta-learning problem and use episodic training [36] to train the model. For a TKG $\mathcal{G} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$, they split its entities into background (seen) entities $\mathcal{E}_{\text{back}}$ and unseen entities $\mathcal{E}'$, where $\mathcal{E}' \cap \mathcal{E}_{\text{back}} = \emptyset$ and $\mathcal{E} = (\mathcal{E}_{\text{back}} \cup \mathcal{E}')$. A background TKG $\mathcal{G}_{\text{back}} \subseteq \mathcal{E}_{\text{back}} \times \mathcal{R} \times \mathcal{E}_{\text{back}} \times \mathcal{T}$ is constructed by including all the TKG facts that do not contain unseen entities. Then, unseen entities $\mathcal{E}'$ are further split into three non-overlapped groups $\mathcal{E}'_{\text{meta-train}}$, $\mathcal{E}'_{\text{meta-valid}}$ and $\mathcal{E}'_{\text{meta-test}}$. The union of all the facts associated to each group's entities forms the corresponding meta-learning set, e.g., the meta-training set $\mathbb{T}_{\text{meta-train}}$ is formulated as $\{(e', r, \tilde{e}, t) | \tilde{e} \in \mathcal{E}, r \in \mathcal{R}, e' \in \mathcal{E}'_{\text{meta-train}}, t \in \mathcal{T}\} \cup \{(\tilde{e}, r, e', t) | \tilde{e} \in \mathcal{E}, r \in \mathcal{R}, e' \in \mathcal{E}'_{\text{meta-train}}, t \in \mathcal{T}\}$. Ding et al. ensure that there exists no link between every two of the meta-learning sets. During meta-training, models are trained over a number of episodes, where a training task T is sampled in each episode. For each task T, N unseen entities $\mathcal{E}_T$ are sampled from $\mathcal{E}'_{\text{meta-train}}$. For each $e' \in \mathcal{E}_T$, K associated facts are sampled to form a support set $Sup_{e'} = \{(e', r_i, \tilde{e}_i, t_i) \text{ or } (\tilde{e}_i, r_i, e', t_i) | \tilde{e}_i \in (\mathcal{E}_{\text{back}} \cup \mathcal{E}'), r_i \in \mathcal{R}, t_i \in \mathcal{T}\}_{i=1}^{K}$, and the rest of its associated facts are taken as its query set $Que_{e'} = \{(e', r_i, \tilde{e}_i, t_i) \text{ or } (\tilde{e}_i, r_i, e', t_i) | \tilde{e}_i \in (\mathcal{E}_{\text{back}} \cup \mathcal{E}'), r_i \in \mathcal{R}, t_i \in \mathcal{T}\}_{i=K+1}^{M_{e'}}$, where $M_{e'}$ denotes the number of e''s associated facts. Models are asked to simultaneously perform LP over $Que_{e'}$ for each $e' \in \mathcal{E}_T$, given their $Sup_{e'}$ and $\mathcal{G}_{\text{back}}$. After meta-training, models are validated with a meta-validation set $\mathbb{T}_{\text{meta-valid}}$ and tested with a meta-test set $\mathbb{T}_{\text{meta-test}}$. In our work, we also train FITCARL in the same way as [13] with episodic training on the same meta-learning problem.

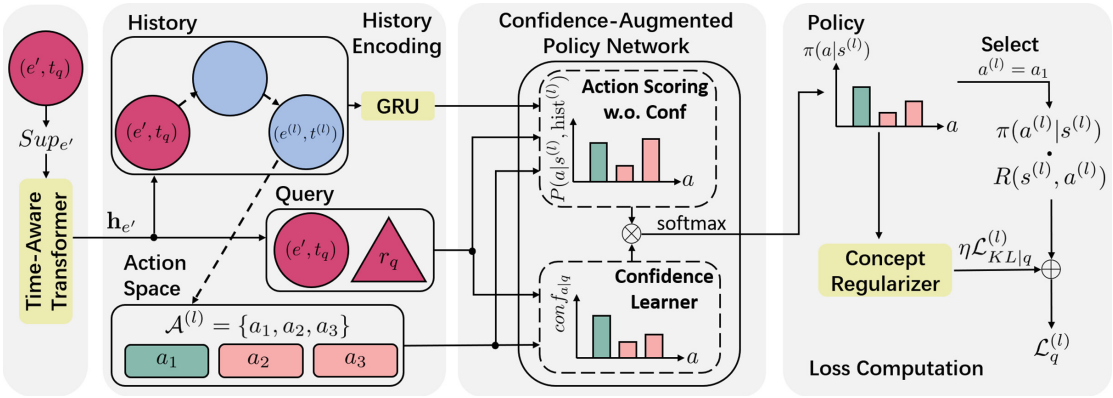


Fig. 1. Overview of FITCARL. To do prediction over the LP query $q = (e', r_q, ?, t_q)$, FITCARL first learns $\mathbf{h}_{e'}$ from a time-aware Transformer. It is then used in history encoding (with GRU) and policy network. To search for the answer, FITCARL starts from node (e', t_q). It goes to $(e^{(l)}, t^{(l)})$, state $s^{(l)}$, at step l. It computes a policy using a confidence-augmented policy network. Assume FITCARL selects action a_1 in current action space $\mathcal{A}^{(l)}$ as the current action $a^{(l)}$. We compute a loss $\mathcal{L}_q^{(l)}$ at step l, considering a_1's probability in policy and reward $R(s^{(l)}, a^{(l)})$, as well as an extra regularization loss $\eta\mathcal{L}_{KL|q}^{(l)}$ computed by a concept regularizer. Please refer to Sect. 4.1, 4.2 and 4.3 for details.

3.2 Concepts for Temporal Knowledge Graph Entities

[13] extracts the concepts of TKG entities by exploring the associated TKBs. Entity concepts describe the characteristics of entities. For example, in the Integrated Crisis Early Warning System (ICEWS) database [6], the entity *Air Force (Canada)* is described with the following concepts: *Air Force*, *Military* and *Government*. Ding et al. propose three ICEWS-based datasets for TKG few-shot OOG LP and manage to couple every entity with its unique concepts. We use $\mathcal{C}$ to denote all the concepts existing in a TKG and $\mathcal{C}_e$ to denote e's concepts.

4 The Proposed FITCARL Model

Given the support set $Sup_{e'} = \{(e', r_i, \tilde{e}_i, t_i) \text{ or } (\tilde{e}_i, r_i, e', t_i)\}_{i=1}^{K}$ of $e' \in \mathcal{E}'$, assume we want to predict the missing entity from the LP query $q = (e', r_q, ?, t_q)$ derived from a query quadruple[2] $(e', r_q, \tilde{e}_q, t_q) \in Que_{e'}$. To achieve this, FITCARL first learns a representation $\mathbf{h}_{e'} \in \mathbb{R}^d$ (d is dimension size) for e' (Sect. 4.1). Then it employs an RL agent that starts from the node (e', t_q) and sequentially takes actions by traversing to other nodes (in the form of (*entity*, *timestamp*)) following a policy (Sect. 4.2 and 4.3). After L traverse steps, the agent is expected to stop at a target node containing $\tilde{e}_q$. Figure 1 shows an overview of FITCARL during training, showing how it computes loss $\mathcal{L}_q^{(l)}$ at step l.

[2] For each query quadruple in the form of $(\tilde{e}_q, r_q, e', t_q)$, we derive its LP query as $(e', r_q^{-1}, ?, t_q)$. r_q^{-1} is r_q's inverse relation. The agent always starts from (e', t_q).

4.1 Learning Unseen Entities with Time-Aware Transformer

We follow FILT [13] and use the entity and relation representations pre-trained with ComplEx [32] for model initialization. Note that pre-training only considers all the background TKG facts, i.e., $\mathcal{G}_{\text{back}}$.

To learn $\mathbf{h}_{e'}$, we start from learning K separate meta-representations. Given $Sup_{e'}$, we transform every support quadruple whose form is $(e', r_i, \tilde{e}_i, t_i)$ to $(\tilde{e}_i, r_i^{-1}, e', t_i)$, where r_i^{-1} denotes the inverse relation[3] of r_i. Then we create a temporal neighborhood $\mathcal{N}_{e'} = \{(\tilde{e}_i, r_i, t_i) | (\tilde{e}_i, r_i, e', t_i) \in Sup_{e'}$ or $(e', r_i^{-1}, \tilde{e}_i, t_i) \in Sup_{e'}\}$ for e' based on $Sup_{e'}$, where $|\mathcal{N}_{e'}| = K$. We compute a meta-representation $\mathbf{h}^i_{e'}$ from each temporal neighbor $(\tilde{e}_i, r_i, t_i)$ as $\mathbf{h}^i_{e'} = f(\mathbf{h}_{\tilde{e}_i} \| \mathbf{h}_{r_i})$, where $\mathbf{h}_{r_i} \in \mathbb{R}^d$ is the representation of the relation r_i and $\|$ is the concatenation operation.

We collect $\{\mathbf{h}^i_{e'}\}_{i=1}^K$ and use a time-aware Transformer to compute a contextualized representation $\mathbf{h}_{e'}$. We treat each temporal neighbor $(\tilde{e}_i, r_i, t_i) \in \mathcal{N}_{e'}$ as a token and the corresponding meta-representation $\mathbf{h}^i_{e'}$ as its token representation. We concatenate the classification ([CLS]) token with the temporal neighbors in $\mathcal{N}_{e'}$ as a sequence and input it into a Transformer, where the sequence length is $K + 1$. The order of temporal neighbors is decided by the sampling order of support quadruples.

To better utilize temporal information from temporal neighbors, we propose a time-aware positional encoding method. For any two tokens u, v in the input sequence, we compute the time difference $t_u - t_v$ between their associated timestamps, and then map it into a time-difference representation $\mathbf{h}_{t_u - t_v} \in \mathbb{R}^d$,

$$\mathbf{h}_{t_u - t_v} = \sqrt{\frac{1}{d}}[cos(\omega_1(t_u - t_v) + \phi_1), ..., cos(\omega_d(t_u - t_v) + \phi_d)]. \tag{1}$$

ω_1 to ω_d and ϕ_1 to ϕ_d are trainable parameters. The timestamp for each temporal neighbor is t_i and we set the timestamp of the [CLS] token to the query timestamp t_q since we would like to use the learned $\mathbf{h}_{e'}$ to predict the LP query happening at t_q. The attention $\text{att}_{u,v}$ of any token v to token u in an attention layer of our time-aware Transformer is written as

$$\begin{aligned} \text{att}_{u,v} &= \frac{\exp(\alpha_{u,v})}{\sum_{k=1}^{K+1} \exp(\alpha_{u,k})}, \\ \alpha_{u,v} &= \frac{1}{\sqrt{d}}(\mathbf{W}_{TrQ}\mathbf{h}_u)^\top (\mathbf{W}_{TrK}\mathbf{h}_v) + \mathbf{w}_{Pos}{}^\top \mathbf{h}_{t_u - t_v}. \end{aligned} \tag{2}$$

$\mathbf{h}_u, \mathbf{h}_v \in \mathbb{R}^d$ are the input representations of token u, v into this attention layer. $\mathbf{W}_{TrQ}, \mathbf{W}_{TrK} \in \mathbb{R}^{d \times d}$ are the weight matrices following original definition in [35]. $\mathbf{w}_{Pos} \in \mathbb{R}^d$ is a parameter that maps $\mathbf{h}_{t_u - t_v}$ to a scalar representing time-aware relative position from token v to u. We use several attention layers and also employ multi-head attention to increase model expressiveness. The output representation of the [CLS] token from the last attention layer is taken as $\mathbf{h}_{e'}$. Figure 2 illustrates how the time-aware Transformer learns $\mathbf{h}_{e'}$ in the 3-shot case.

[3] Both original and inverse relations are trained in pre-training.

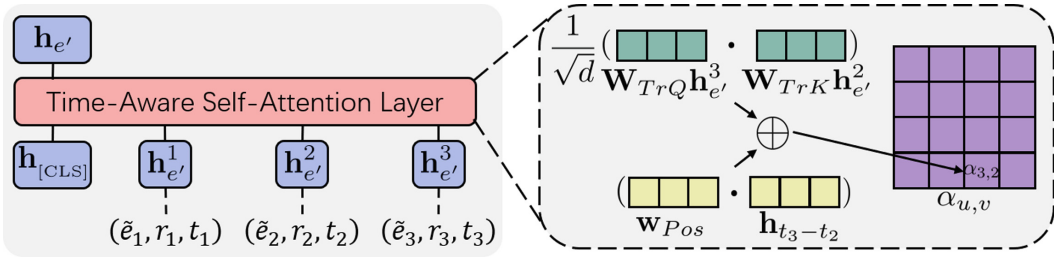


Fig. 2. Time-aware Transformer with one attention layer for learning unseen entity representation in the 3-shot case.

4.2 Reinforcement Learning Framework

We formulate the RL process as a Markov Decision Process, and we introduce its elements as follows. **(1) States:** Let $\mathcal{S}$ be a state space. A state is denoted as $s^{(l)} = (e^{(l)}, t^{(l)}, e', r_q, t_q) \in \mathcal{S}$. $(e^{(l)}, t^{(l)})$ is the node that is visited by the agent at step l and e', r_q, t_q are taken from the LP query $(e', r_q, ?, t_q)$. The agent starts from (e', t_q), and thus $s^{(0)} = (e', t_q, e', r_q, t_q)$. **(2) Actions:** Let $\mathcal{A}$ denote an action space and $\mathcal{A}^{(l)} \subset \mathcal{A}$ denotes the action space at step l. $\mathcal{A}^{(l)}$ is sampled from all the possible outgoing edges starting from $(e^{(l)}, t^{(l)})$, i.e., $\{a = (r, e, t) | (e^{(l)}, r, e, t) \in (\mathcal{G}_{\text{back}} \cup \bigcup_{e'' \in \mathcal{E}_T} Sup_{e''}), r \in \mathcal{R}, e \in (\mathcal{E}_{\text{back}} \cup \mathcal{E}_T), t \in \mathcal{T}\}$. We do sampling because if $e^{(l)} \in \mathcal{E}_{\text{back}}$, there probably exist lots of outgoing edges in $\mathcal{G}_{\text{back}}$. If we include all of them into $\mathcal{A}^{(l)}$, they will lead to an excessive consumption of memory and cause out-of-memory problem on hardware devices. We sample $\mathcal{A}^{(l)}$ in a time-adaptive manner. For each outgoing edge (r, e, t), we compute a score $\mathbf{w}_{\Delta t}{}^{\top} \mathbf{h}_{t_q - t}$, where $\mathbf{w}_{\Delta t} \in \mathbb{R}^d$ is a time modeling weight and $\mathbf{h}_{t_q - t}$ is the representation denoting the time difference $t_q - t$. $\mathbf{h}_{t_q - t}$ is computed as in Eq. 1 with shared parameters. We rank the scores of outgoing edges in descending order and take a fixed number of top-ranked edges as $\mathcal{A}^{(l)}$. We also include one self-loop action in each $\mathcal{A}^{(l)}$ that makes the agent stay at the current node. **(3) Transition:** A transition function δ is used to transfer from one state to another, i.e., $\delta(s^{(l)}, a^{(l)}) = s^{(l+1)} = (e^{(l+1)}, t^{(l+1)}, e', r_q, t_q)$, according to the selected action $a^{(l)}$. **(4) Rewards:** We give the agent a reward at each step of state transition and consider a cumulative reward for the whole searching process. The reward of doing a candidate action $a \in \mathcal{A}^{(l)}$ at step l is given as $R(s^{(l)}, a) = \text{Sigmoid}\left(\theta - \left\|\mathbf{h}_{\tilde{e}_q} - \mathbf{h}_{e_a}\right\|_2\right)$. θ is a hyperparameter adjusting the range of reward. $\mathbf{h}_{e_a}$ denotes the representation of entity e_a selected in the action $a = (r_a, e_a, t_a)$. $\|\cdot\|_2$ is the L2 norm. The closer e_a is to $\tilde{e}_q$, the greater reward the agent gets if it does action a.

4.3 Confidence-Augmented Policy Network

We design a confidence-augmented policy network that calculates the probability distribution over all the candidate actions $\mathcal{A}^{(l)}$ at the search step l, according to the current state $s^{(l)}$, the search history $\text{hist}^{(l)} = ((e', t_q), r^{(1)}, (e^{(1)}, t^{(1)}), ..., r^{(l)}, (e^{(l)}, t^{(l)}))$, and the confidence $\text{conf}_{a|q}$ of each $a \in \mathcal{A}^{(l)}$. During the search,

we represent each visited node with a time-aware representation related to the LP query q. For example, for the node $(e^{(l)}, t^{(l)})$ visited at step l, we compute its representation as $\mathbf{h}_{(e^{(l)},t^{(l)})} = \mathbf{h}_{e^{(l)}} \| \mathbf{h}_{t_q - t^{(l)}}$. $\mathbf{h}_{t_q - t^{(l)}}$ is computed as same in Eq. 1 and parameters are shared.

Encoding Search History. The search history $\text{hist}^{(l)}$ is encoded as

$$\begin{aligned} \mathbf{h}_{\text{hist}^{(l)}} &= \text{GRU}\left(\left(\mathbf{h}_{r^{(l)}} \| \mathbf{h}_{(e^{(l)},t^{(l)})}\right), \mathbf{h}_{\text{hist}^{(l-1)}}\right), \\ \mathbf{h}_{\text{hist}^{(0)}} &= \text{GRU}\left(\left(\mathbf{h}_{r_{\text{dummy}}} \| \mathbf{h}_{(e',t_q)}\right), \mathbf{0}\right). \end{aligned} \tag{3}$$

GRU is a gated recurrent unit [9]. $\mathbf{h}_{\text{hist}^{(0)}} \in \mathbb{R}^{3d}$ is the initial hidden state of GRU and $\mathbf{h}_{r_{\text{dummy}}} \in \mathbb{R}^{d}$ is the representation of a dummy relation for GRU initialization. $\mathbf{h}_{(e',t_q)}$ is the time-aware representation of the starting node (e', t_q).

Confidence-Aware Action Scoring. We design a score function for computing the probability of selecting each candidate action $a \in \mathcal{A}^{(l)}$. Assume $a = (r_a, e_a, t_a)$, where $(e^{(l)}, r_a, e_a, t_a) \in (\mathcal{G}_{\text{back}} \cup \bigcup_{e'' \in \mathcal{E}_T} Sup_{e''})$. We first compute an attentional feature $\mathbf{h}_{\text{hist}^{(l)},q|a}$ that extracts the information highly-related to action a from the visited search history $\text{hist}^{(l)}$ and the LP query q.

$$\begin{aligned} \mathbf{h}_{\text{hist}^{(l)},q|a} &= \text{att}_{\text{hist}^{(l)},a} \cdot \bar{\mathbf{h}}_{\text{hist}^{(l)}} + \text{att}_{q,a} \cdot \bar{\mathbf{h}}_q, \\ \bar{\mathbf{h}}_{\text{hist}^{(l)}} &= \mathbf{W}_1{}^{\top} \mathbf{h}_{\text{hist}^{(l)}}, \quad \bar{\mathbf{h}}_q = \mathbf{W}_2{}^{\top} \left(\mathbf{h}_{r_q} \| \mathbf{h}_{(e',t_q)}\right). \end{aligned} \tag{4}$$

$\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{2d \times 3d}$ are two weight matrices. $\mathbf{h}_{r_q}$ is the representation of the query relation r_q. $\text{att}_{\text{hist}^{(l)},a}$ and $\text{att}_{q,a}$ are two attentional weights that are defined as

$$\text{att}_{\text{hist}^{(l)},a} = \frac{\exp(\phi_{\text{hist}^{(l)},a})}{\exp(\phi_{\text{hist}^{(l)},a}) + \exp(\phi_{q,a})}, \text{att}_{q,a} = \frac{\exp(\phi_{q,a})}{\exp(\phi_{\text{hist}^{(l)},a}) + \exp(\phi_{q,a})}, \tag{5}$$

where

$$\begin{gathered} \phi_{\text{hist}^{(l)},a} = \bar{\mathbf{h}}_a^{\top} \bar{\mathbf{h}}_{\text{hist}^{(l)}} + \mathbf{w}_{\Delta t}^{\top} \mathbf{h}_{t_a - t^{(l)}}, \ \phi_{q,a} = \bar{\mathbf{h}}_a^{\top} \bar{\mathbf{h}}_q + \mathbf{w}_{\Delta t}^{\top} \mathbf{h}_{t_a - t_q}, \\ \bar{\mathbf{h}}_a = \mathbf{W}_3{}^{\top} \left(\mathbf{h}_{r_a} \| \mathbf{h}_{(e_a,t_a)}\right). \end{gathered} \tag{6}$$

$\mathbf{W}_3 \in \mathbb{R}^{2d \times 3d}$ is a weight matrix. $\mathbf{h}_{r_a}$ is the representation of r_a. $\mathbf{h}_{(e_a,t_a)}$ is the time-aware representation of node (e_a, t_a) from action a. $\mathbf{w}_{\Delta t}$ maps time differences to a scalar indicating how temporally important is the action a to the history and the query q. We take $t^{(l)}$ as search history's timestamp because it is the timestamp of the node where the search stops. Before considering confidence, we compute a probability for each candidate action $a \in \mathcal{A}^{(l)}$ at step l

$$P(a|s^{(l)}, \text{hist}^{(l)}) = \frac{\exp(\bar{\mathbf{h}}_a^{\top} \mathbf{W}_4 \mathbf{h}_{\text{hist}^{(l)},q|a})}{\sum_{a' \in \mathcal{A}^{(l)}} \exp(\bar{\mathbf{h}}_{a'}^{\top} \mathbf{W}_4 \mathbf{h}_{\text{hist}^{(l)},q|a'})}, \tag{7}$$

where $\mathbf{W}_4 \in \mathbb{R}^{2d \times 2d}$ is a weight matrix. The probability of each action a is decided by its associated node (e_a, t_a) and the attentional feature $\mathbf{h}_{\text{hist}^{(l)},q|a}$ that adaptively selects the information highly-related to a.

In TKG few-shot OOG LP, only a small number of K edges associated to each unseen entity are observed. This leads to an incomprehensive action space $\mathcal{A}^{(0)}$ at the start of search because our agent starts travelling from node (e', t_q) and $|\mathcal{A}^{(0)}| = K$ is extremely tiny. Besides, since there exist plenty of unseen entities in $\mathcal{E}_T$, it is highly probable that the agent travels to the nodes with other unseen entities during the search, causing it sequentially experience multiple tiny action spaces. As the number of the experienced incomprehensive action spaces increases, more noise will be introduced in history encoding. From Eqs. 4 to 7, we show that we heavily rely on the search history for computing candidate action probabilities. To address this problem, we design a confidence learner that learns the confidence $\text{conf}_{a|q}$ of each $a \in \mathcal{A}^{(l)}$, independent of the search history. The form of confidence learner is inspired by a KG score function TuckER [4].

$$\text{conf}_{a|q} = \frac{\exp(\psi_{a|q})}{\sum_{a' \in \mathcal{A}^{(l)}} \exp(\psi_{a'|q})}, \text{ where } \psi_{a|q} = \mathcal{W} \times_1 \mathbf{h}_{(e',t_q)} \times_2 \mathbf{h}_{r_q} \times_3 \mathbf{h}_{(e_a,t_a)}. \tag{8}$$

$\mathcal{W} \in \mathbb{R}^{2d \times d \times 2d}$ is a learnable core tensor introduced in [4]. As defined in tucker decomposition [33], $\times_1, \times_2, \times_3$ are three operators indicating the tensor product in three different modes (see [4,33] for detailed explanations). Equation 8 can be interpreted as another action scoring process that is irrelevant to the search history. If $\psi_{a|q}$ is high, then it implies that choosing action a is sensible and e_a is likely to resemble the ground truth missing entity $\tilde{e}_q$. Accordingly, the candidate action a will be assigned a great confidence. In this way, we alleviate the negative influence of cascaded noise introduced by multiple tiny action spaces in the search history. The policy $\pi(a|s^{(l)})$ at step l is defined as

$$\pi(a|s^{(l)}) = \frac{\exp(P(a|s^{(l)}, \text{hist}^{(l)}) \cdot \text{conf}_{a|q})}{\sum_{a' \in \mathcal{A}^{(l)}} \exp(P(a'|s^{(l)}, \text{hist}^{(l)}) \cdot \text{conf}_{a'|q})} \tag{9}$$

4.4 Concept Regularizer

In the background TKG $\mathcal{G}_{\text{back}}$, the object entities of each relation conform to a unique distribution. For each relation $r \in \mathcal{R}$, we track all the TKG facts containing r in $\mathcal{G}_{\text{back}}$, and pick out all their object entities $\mathcal{E}_r$ ($\mathcal{E}_r \in \mathcal{E}_{\text{back}}$) together with their concepts $\{\mathcal{C}_e | e \in \mathcal{E}_r\}$. We sum up the number of appearances n_c of each concept c and compute a probability $P(c|r)$ denoting how probable it is to see c when we perform object prediction[4] over the LP queries concerning r. For example, for r, $\mathcal{E}_r = \{e_1, e_2\}$ and $\mathcal{C}_{e_1} = \{c_1, c_2\}$, $\mathcal{C}_{e_2} = \{c_2\}$. The probability $P(c_1|r) = n_{c_1} / \sum_{c \in \mathcal{C}} n_c = 1/3$, $P(c_2|r) = n_{c_2} / \sum_{c \in \mathcal{C}} n_c = 2/3$. Assume we have an LP query $q = (e', r_q, ?, t_q)$, and at search step l, we have an action probability from policy $\pi(a|s^{(l)})$ for each candidate action $a \in \mathcal{A}^{(l)}$. We collect the concepts $\mathcal{C}_{e_a}$ of e_a in each action a and compute a concept-aware action probability

$$P(a|\mathcal{C}_{e_a}, q) = \frac{\exp(\sum_{c \in \mathcal{C}_{e_a}} P(c|r_q))}{\sum_{a' \in \mathcal{A}^{(l)}} \exp(\sum_{c' \in \mathcal{C}_{e_{a'}}} P(c'|r_q))} \tag{10}$$

[4] All LP queries are transformed into object prediction in TKG few-shot OOG LP.

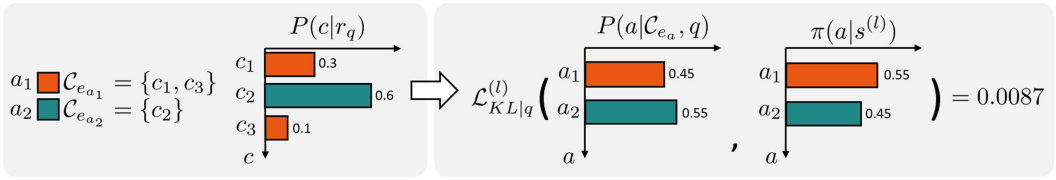


Fig. 3. Concept regularizer. $P(a_1|\mathcal{C}_{e_{a_1}}, q) = \exp(0.3+0.1)/(\exp(0.3+0.1)+\exp(0.6)) = 0.45$. $P(a_2|\mathcal{C}_{e_{a_2}}, q) = \exp(0.6)/(\exp(0.3+0.1)+\exp(0.6)) = 0.55$.

We then compute the Kullback-Leibler (KL) divergence between $P(a|\mathcal{C}_{e_a}, q)$ and $\pi(a|s^{(l)})$ and minimize it during parameter optimization.

$$\mathcal{L}^{(l)}_{\text{KL}|q} = \sum_{a \in \mathcal{A}^{(l)}} \pi(a|s^{(l)}) \log \left(\frac{\pi(a|s^{(l)})}{P(a|\mathcal{C}_{e_a}, q)} \right). \tag{11}$$

Note that $r_q \in \mathcal{R}$ is observable in $\mathcal{G}_{\text{back}}$. $\mathcal{G}_{\text{back}}$ is huge and contains a substantial number of facts of r_q. As stated in FILT [13], although we have only K associated edges for each unseen entity e', its concepts $\mathcal{C}_{e'}$ is known. Our concept regularizer enables a parameter-free approach to match the concept-aware action probability $P(a|\mathcal{C}_{e_a}, q)$ with the action probability taken from the policy $\pi(a|s^{(l)})$. It can be taken as guiding the policy to conform to the distribution of r_q's objects' concepts observed in $\mathcal{G}_{\text{back}}$. We illustrate our concept regularizer in Fig. 3.

4.5 Parameter Learning

Following [13], we train FITCARL with episodic training. In each episode, a training task T is sampled, where we sample a $Sup_{e'}$ for every unseen entity $e' \in \mathcal{E}'_{\text{meta-train}}$ ($\mathcal{E}_T = \mathcal{E}'_{\text{meta-train}}$) and calculate loss over $Que_{e'}$. For each LP query q, we aim to maximize the cumulative reward along L steps of search. We write our loss function (we minimize our loss) for each training task T as follows.

$$\mathcal{L}_T = \frac{1}{\sum_{e'} |Que_{e'}|} \sum_{e'} \sum_{q \in Que_{e'}} \sum_{l=0}^{L-1} \gamma^l \mathcal{L}^{(l)}_q, \quad \mathcal{L}^{(l)}_q = \eta \mathcal{L}^{(l)}_{\text{KL}|q} - \log(\pi(a^{(l)}|s^{(l)}))R(s^{(l)}, a^{(l)}). \tag{12}$$

$a^{(l)}$ is the selected action at search step l. γ^l is the l^{th} order of a discount factor $\gamma \in [0, 1)$. η is a hyperparameter deciding the magnitude of concept regularization. We use Algorithm 1 in Appendix E to further illustrate our meta-training process.

5 Experiments

We compare FITCARL with baselines on TKG few-shot OOG LP (Sect. 5.2). In Sect. 5.3, we first do several ablation studies to study the effectiveness of different model components. We then plot the performance over time to show FITCARL's robustness and present a case study to show FITCARL's explainability and the importance of learning confidence. We provide implementation details in Appendix A.

5.1 Experimental Setting

We do experiments on three datasets proposed in [13], i.e., ICEWS14-OOG, ICEWS18-OOG and ICEWS0515-OOG. They contain the timestamped political facts in 2014, 2018 and from 2005 to 2015, respectively. All of them are constructed by taking the facts from the ICEWS [6] TKB. Dataset statistics are shown in Table 1. We employ two evaluation metrics, i.e., mean reciprocal rank (MRR) and Hits@1/3/10. We provide detailed definitions of both metrics in Appendix D. We use the filtered setting proposed in [5] for fairer evaluation. For baselines, we consider the following methods. (1) Two traditional KGC methods, i.e., ComplEx [32] and BiQUE [14]. (2) Three traditional TKGC methods, i.e., TNTComplEx [18], TeLM [41], and TeRo [42]. (3) Three inductive KGC methods, i.e., MEAN [15], LAN [37], and GEN [3]. Among them, only GEN is trained with a meta-learning framework. (4) Two inductive TKG reasoning methods, including an inductive TKG forecasting method TITer [30], and a meta-learning-based inductive TKGC method FILT [13] (FILT is the only previous work developed to solve TKG few-shot OOG LP). We take the experimental results of all baselines (except TITer) from [13]. Following [13], we train TITer over all the TKG facts in $\mathcal{G}_{\text{back}}$ and $\mathbb{T}_{\text{meta-train}}$. We constrain TITer to only observe support quadruples of each test entity in $\mathcal{E}'_{\text{meta-test}}$ for inductive learning during inference. All methods are tested over exactly the same test examples.

Table 1. Dataset statistics.

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{T}\rvert$	$\lvert\mathcal{E}'_{\text{meta-train}}\rvert$	$\lvert\mathcal{E}'_{\text{meta-valid}}\rvert$	$\lvert\mathcal{E}'_{\text{meta-test}}\rvert$	$\lvert\mathcal{G}_{\text{back}}\rvert$	$\lvert\mathbb{T}_{\text{meta-train}}\rvert$	$\lvert\mathbb{T}_{\text{meta-valid}}\rvert$	$\lvert\mathbb{T}_{\text{meta-test}}\rvert$
ICEWS14-OOG	7128	230	365	385	48	49	83448	5772	718	705
ICEWS18-OOG	23033	256	304	1268	160	158	444269	19291	2425	2373
ICEWS0515-OOG	10488	251	4017	647	80	82	448695	10115	1217	1228

5.2 Main Results

Table 2 shows the experimental results of TKG 1-shot/3-shot OOG LP. We observe that traditional KGC and TKGC methods are beaten by inductive learning methods. It is because traditional methods cannot handle unseen entities. Besides, we also find that meta-learning-based methods, i.e., GEN, FILT and FITCARL, show better performance than other inductive learning methods. This is because meta-learning is more suitable for dealing with few-shot learning problems. FITCARL shows superior performance over all metrics on all datasets. It outperforms the previous stat-of-the-art FILT with a huge margin. We attribute it to several reasons. (1) Unlike FILT that uses KG score function over all the entities for prediction, FITCARL is an RL-based method that directly searches the predicted answer through their multi-hop temporal neighborhood, making it better capture highly-related graph information through time. (2) FITCARL takes advantage of its confidence learner. It helps to alleviate the negative impact from the few-shot setting. (3) Concept regularizer serves as a strong tool for exploiting concept-aware information in TKBs and adaptively

guides FITCARL to learn a policy that conforms to the concept distribution shown in $\mathcal{G}_{\text{back}}$.

Table 2. Experimental results of TKG 1-shot and 3-shot OOG LP. Evaluation metrics are MRR and Hits@1/3/10 (H@1/3/10). Best results are marked bold.

Datasets	ICEWS14-OOG								ICEWS18-OOG								ICEWS0515-OOG							
	MRR		H@1		H@3		H@10		MRR		H@1		H@3		H@10		MRR		H@1		H@3		H@10	
Model	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S
ComplEx	.048	.046	.018	.014	.045	.046	.099	.089	.039	.044	.031	.026	.048	.042	.085	.093	.077	.076	.045	.048	.074	.071	.129	.120
BiQUE	.039	.035	.015	.014	.041	.030	.073	.066	.029	.032	.022	.021	.033	.037	.064	.073	.075	.083	.044	.049	.072	.077	.130	.144
TNTComplEx	.043	.044	.015	.016	.033	.042	.102	.096	.046	.048	.023	.026	.043	.044	.087	.082	.034	.037	.014	.012	.031	.036	.060	.071
TeLM	.032	.035	.012	.009	.021	.023	.063	.077	.049	.019	.029	.001	.045	.013	.084	.054	.080	.072	.041	.034	.077	.072	.138	.151
TeRo	.009	.010	.002	.002	.005	.002	.015	.020	.007	.006	.003	.001	.006	.003	.013	.006	.012	.023	.000	.010	.008	.017	.024	.040
MEAN	.035	.144	.013	.054	.032	.145	.082	.339	.016	.101	.003	.014	.012	.114	.043	.283	.019	.148	.003	.039	.017	.175	.052	.384
LAN	.168	.199	.050	.061	.199	.255	.421	.500	.077	.127	.018	.025	.067	.165	.199	.344	.171	.182	.081	.068	.180	.191	.367	.467
GEN	.231	.234	.162	.155	.250	.284	.378	.389	.171	.216	.112	.137	.189	.252	.289	.351	.268	.322	.185	.231	.308	.362	.413	.507
TITer	.144	.200	.105	.148	.163	.226	.228	.314	.064	.115	.038	.076	.075	.131	.011	.186	.115	.228	.080	.168	.130	.262	.173	.331
FILT	.278	.321	.208	.240	.305	.357	.410	.475	.191	.266	.129	.187	.209	.298	.316	.417	.273	.370	.201	.299	.303	.391	.405	.516
FITCARL	**.418**	**.481**	**.284**	**.329**	**.522**	**.646**	**.681**	**.696**	**.297**	**.370**	**.156**	**.193**	**.386**	**.559**	**.584**	**.627**	**.345**	**.513**	**.202**	**.386**	**.482**	**.618**	**.732**	**.700**

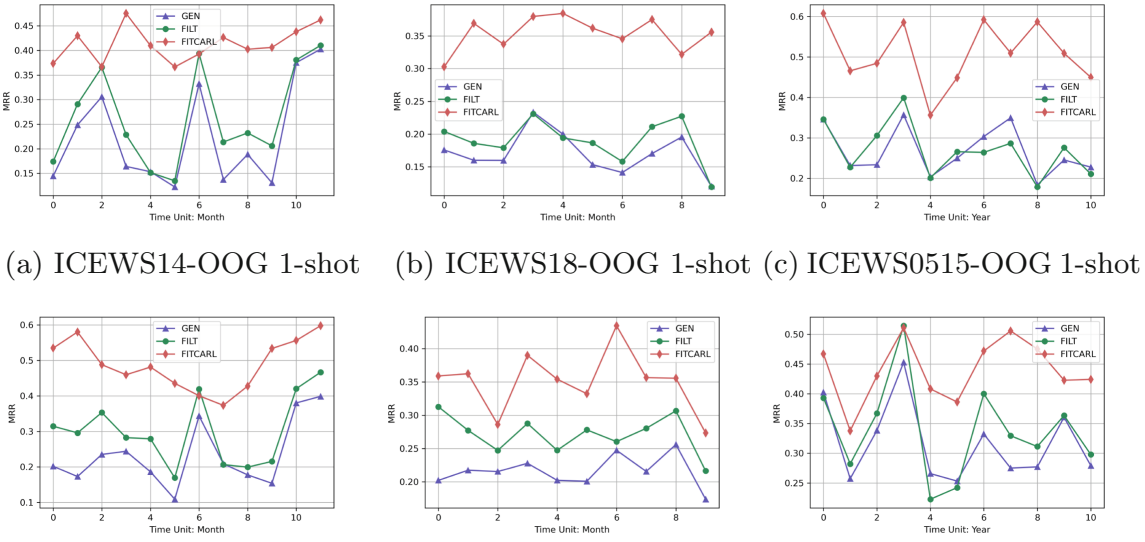
5.3 Further Analysis

Ablation Study. We conduct several ablation studies to study the effectiveness of different model components. **(A) Action Space Sampling Variants:** To prevent oversized action space $\mathcal{A}^{(l)}$, we use a time-adaptive sampling method (see Sect. 4.2). We show its effectiveness by switching it to random sample (ablation A1) and time-proximity sample (ablation A2). In time-proximity sample, we take a fixed number of outgoing edges temporally closest to the current node at $t^{(l)}$ as $\mathcal{A}^{(l)}$. We keep $|\mathcal{A}^{(l)}|$ unchanged. **(B) Removing Confidence Learner:** In ablation B, we remove the confidence learner. **(C) Removing Concept Regularizer:** In ablation C, we remove concept regularizer. **(D) Time-Aware Transformer Variants:** We remove the time-aware positional encoding method by deleting the second term of Eq. 2. **(E) Removing Temporal Reasoning Modules:** In ablation E, we study the importance of temporal reasoning. We first combine ablation A1 and D, and then delete every term related to time difference representations computed with Eq. 1. We create a model variant without using any temporal information (see Appendix C for detailed setting). We present the experimental results of ablation studies in Table 3. From ablation A1 and A2, we observe that time-adaptive sample is effective. We also see a great performance drop in ablation B and C, indicating the strong importance of our confidence learner and concept regularizer. We only do ablation D for 3-shot model because in 1-shot case our model does not need to distinguish the importance of multiple support quadruples. We find that our time-aware positional encoding makes great contribution. Finally, we observe that ablation E shows poor performance (worse than A1 and D in most cases), implying that incorporating temporal information is essential for FITCARL to solve TKG few-shot OOG LP.

Table 3. Ablation study results. Best results are marked bold.

Datasets	ICEWS14-OOG								ICEWS18-OOG								ICEWS0515-OOG							
	MRR		H@1		H@3		H@10		MRR		H@1		H@3		H@10		MRR		H@1		H@3		H@10	
Model	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S	1-S	3-S
A1	.404	.418	.283	.287	.477	.494	.647	.667	.218	.260	.153	.167	.220	.296	.404	.471	.190	.401	.108	.289	.196	.467	.429	.624
A2	.264	.407	.241	.277	.287	.513	.288	.639	.242	.265	.126	.168	.337	.291	.444	.499	.261	.414	.200	.267	.298	.545	.387	.640
B	.373	.379	.255	.284	.454	.425	.655	.564	.156	.258	.106	.191	.162	.271	.273	.398	.285	.411	.198	.336	.328	.442	.447	.567
C	.379	.410	.265	.236	.489	.570	.667	.691	.275	.339	.153	.190	.346	.437	.531	.556	.223	.411	.130	.243	.318	.544	.397	.670
D	–	.438	–	.262	–	.626	–	.676	–	.257	–	.160	–	.280	–	.500	–	.438	–	.262	–	.610	–	.672
E	.270	.346	.042	.178	.480	.466	.644	.662	.155	.201	.012	.117	.197	.214	.543	.429	.176	.378	.047	.239	.194	.501	.506	.584
FITCARL	**.418**	**.481**	**.284**	**.329**	**.522**	**.646**	**.681**	**.696**	**.297**	**.370**	**.156**	**.193**	**.386**	**.559**	**.584**	**.627**	**.345**	**.513**	**.202**	**.386**	**.482**	**.618**	**.732**	**.700**

Performance Over Time. To demonstrate the robustness of FITCARL, we plot its MRR performance over prediction time (query time t_q). We compare FITCARL with two meta-learning-based strong baselines GEN and FILT. From Figs. 4a to 4f, we find that our model can constantly outperform baselines. This indicates that FITCARL improves LP performance for examples existing at almost all timestamps, proving its robustness. GEN is not designed for TKG reasoning, and thus it cannot show optimal performance. Although FILT is designed for TKG few-shot OOG LP, we show that our RL-based model is much stronger.

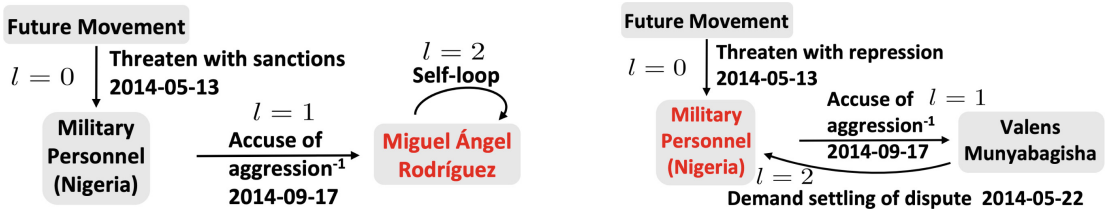


(a) ICEWS14-OOG 1-shot (b) ICEWS18-OOG 1-shot (c) ICEWS0515-OOG 1-shot

(d) ICEWS14-OOG 3-shot (e) ICEWS18-OOG 3-shot (f) ICEWS0515-OOG 3-shot

Fig. 4. Performance comparison among FITCARL, FILT and GEN over different query time t_q. Horizontal axis of each subfigure denotes how temporally faraway from the first timestamp. We aggregate the performance of each month to one point in ICEWS14-OOG and ICEWS18-OOG. A point for ICEWS0515-OOG denotes the aggregated performance in each year.

Case Study. We do a case study to show how FITCARL provides explainability and how the confidence learner helps in reasoning. We ask 3-shot FITCARL and its variant without the confidence learner (both trained on ICEWS14-OOG) to predict the missing entity of the LP query (*Future Movement*, *Express intent to cooperate on intelligence*, ?, 2014-11-12), where *Future Movement* is a newly-emerged entity that is unseen during training and the answer to this LP query is *Miguel Ángel Rodríguez*. We visualize a specific reasoning path of each model and present them in Fig. 5. The relation *Express intent to cooperate on intelligence* indicates a positive relationship between subject and object entities. FITCARL performs a search with length $L = 3$, where it finds an entity *Military Personnel (Nigeria)* that is in a negative relationship with both *Future Movement* and *Miguel Ángel Rodríguez*. FITCARL provides explanation by finding a reasoning path representing the proverb: The enemy of the enemy is my friend. For FITCARL without confidence learner, we find that it can also provide similar explanation by finding another entity that is also an enemy of *Military Personnel (Nigeria)*. However, it fails to find the ground truth answer because it neglects the confidence of each action. The confidence learner assigns high probability to the ground truth entity, leading to a correct prediction.



(a) FITCARL (b) FITCARL w.o. Confidence

Fig. 5. Case study reasoning path visualization. The entity marked in red are the answer predicted by the model. w.o. means without. (Color figure online)

6 Conclusion

We present an RL-based TKGC method FITCARL to solve TKG few-shot OOG LP, where models are asked to predict the links concerning newly-emerged entities that have only a few observed associated facts. FITCARL is a meta-learning-based model trained with episodic training. It learns representations of newly-emerged entities by using a time-aware Transformer. To further alleviate the negative impact of the few-shot setting, a confidence learner is proposed to be coupled with the policy network for making better decisions. A parameter-free concept regularizer is also developed to better exploit concept-aware information in TKBs. Experimental results show that FITCARL achieves a new state-of-the-art and provides explainability.

References

1. Abboud, R., Ceylan, İ.İ., Lukasiewicz, T., Salvatori, T.: Boxe: a box embedding model for knowledge base completion. In: NeurIPS (2020)
2. Ammanabrolu, P., Hausknecht, M.J.: Graph constrained reinforcement learning for natural language action spaces. In: ICLR. OpenReview.net (2020)
3. Baek, J., Lee, D.B., Hwang, S.J.: Learning to extrapolate knowledge: transductive few-shot out-of-graph link prediction. In: NeurIPS (2020)
4. Balazevic, I., Allen, C., Hospedales, T.M.: Tucker: tensor factorization for knowledge graph completion. In: EMNLP/IJCNLP (1), pp. 5184–5193. Association for Computational Linguistics (2019)
5. Bordes, A., Usunier, N., García-Durán, A., Weston, J., Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In: NIPS, pp. 2787–2795 (2013)
6. Boschee, E., Lautenschlager, J., O'Brien, S., Shellman, S., Starz, J., Ward, M.: ICEWS Coded Event Data (2015)
7. Chen, K., Wang, Y., Li, Y., Li, A.: Rotateqvs: representing temporal information as rotations in quaternion vector space for temporal knowledge graph completion. In: ACL (1), pp. 5843–5857. Association for Computational Linguistics (2022)
8. Chen, M., Zhang, W., Zhang, W., Chen, Q., Chen, H.: Meta relational learning for few-shot link prediction in knowledge graphs. In: EMNLP/IJCNLP (1), pp. 4216–4225. Association for Computational Linguistics (2019)
9. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: EMNLP, pp. 1724–1734. ACL (2014)
10. Ding, Z., He, B., Ma, Y., Han, Z., Tresp, V.: Learning meta representations of one-shot relations for temporal knowledge graph link prediction. CoRR abs/2205.10621 (2022)
11. Ding, Z., Ma, Y., He, B., Han, Z., Tresp, V.: A simple but powerful graph encoder for temporal knowledge graph completion. In: NeurIPS 2022 Temporal Graph Learning Workshop (2022)
12. Ding, Z., et al.: Forecasting question answering over temporal knowledge graphs. CoRR abs/2208.06501 (2022)
13. Ding, Z., Wu, J., He, B., Ma, Y., Han, Z., Tresp, V.: Few-shot inductive learning on temporal knowledge graphs using concept-aware information. In: 4th Conference on Automated Knowledge Base Construction (2022)
14. Guo, J., Kok, S.: Bique: biquaternionic embeddings of knowledge graphs. In: EMNLP (1), pp. 8338–8351. Association for Computational Linguistics (2021)
15. Hamaguchi, T., Oiwa, H., Shimbo, M., Matsumoto, Y.: Knowledge transfer for out-of-knowledge-base entities: a graph neural network approach. In: IJCAI, pp. 1802–1808. ijcai.org (2017)
16. He, Y., Wang, Z., Zhang, P., Tu, Z., Ren, Z.: VN network: embedding newly emerging entities with virtual neighbors. In: CIKM, pp. 505–514. ACM (2020)
17. Jung, J., Jung, J., Kang, U.: Learning to walk across time for interpretable temporal knowledge graph completion. In: KDD, pp. 786–795. ACM (2021)
18. Lacroix, T., Obozinski, G., Usunier, N.: Tensor decompositions for temporal knowledge base completion. In: ICLR. OpenReview.net (2020)
19. Leblay, J., Chekol, M.W.: Deriving validity time in knowledge graph. In: WWW (Companion Volume), pp. 1771–1776. ACM (2018)
20. Li, J., Tang, T., Zhao, W.X., Wei, Z., Yuan, N.J., Wen, J.: Few-shot knowledge graph-to-text generation with pretrained language models. In: ACL/IJCNLP (Findings). Findings of ACL, vol. ACL/IJCNLP 2021, pp. 1558–1568. Association for Computational Linguistics (2021)

21. Li, Z., et al.: Search from history and reason for future: two-stage reasoning on temporal knowledge graphs. In: ACL/IJCNLP (1), pp. 4732–4743. Association for Computational Linguistics (2021)
22. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI, pp. 2181–2187. AAAI Press (2015)
23. Messner, J., Abboud, R., Ceylan, İ.İ.: Temporal knowledge graph completion using box embeddings. In: AAAI, pp. 7779–7787. AAAI Press (2022)
24. Mirtaheri, M., Rostami, M., Ren, X., Morstatter, F., Galstyan, A.: One-shot learning for temporal knowledge graphs. In: 3rd Conference on Automated Knowledge Base Construction (2021)
25. Nickel, M., Tresp, V., Kriegel, H.: A three-way model for collective learning on multi-relational data. In: ICML, pp. 809–816. Omnipress (2011)
26. Sadeghian, A., Armandpour, M., Colas, A., Wang, D.Z.: Chronor: rotation based temporal knowledge graph embedding. In: AAAI, pp. 6471–6479. AAAI Press (2021)
27. Saxena, A., Tripathi, A., Talukdar, P.P.: Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In: ACL, pp. 4498–4507. Association for Computational Linguistics (2020)
28. Schlichtkrull, M., Kipf, T.N., Bloem, P., van den Berg, R., Titov, I., Welling, M.: Modeling relational data with graph convolutional networks. In: Gangemi, A., et al. (eds.) ESWC 2018. LNCS, vol. 10843, pp. 593–607. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-93417-4_38
29. Sheng, J., et al.: Adaptive attentional network for few-shot knowledge graph completion. In: EMNLP (1), pp. 1681–1691. Association for Computational Linguistics (2020)
30. Sun, H., Zhong, J., Ma, Y., Han, Z., He, K.: Timetraveler: reinforcement learning for temporal knowledge graph forecasting. In: EMNLP (1), pp. 8306–8319. Association for Computational Linguistics (2021)
31. Tresp, V., Esteban, C., Yang, Y., Baier, S., Krompaß, D.: Learning with memory embeddings. arXiv preprint arXiv:1511.07972 (2015)
32. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: ICML, JMLR Workshop and Conference Proceedings, vol. 48, pp. 2071–2080. JMLR.org (2016)
33. Tucker, L.R.: The extension of factor analysis to three-dimensional matrices. In: Gulliksen, H., Frederiksen, N. (eds.) Contributions to Mathematical Psychology, pp. 110–127. Holt, Rinehart and Winston, New York (1964)
34. Vashishth, S., Sanyal, S., Nitin, V., Talukdar, P.P.: Composition-based multi-relational graph convolutional networks. In: ICLR. OpenReview.net (2020)
35. Vaswani, A., et al.: Attention is all you need. In: NIPS, pp. 5998–6008 (2017)
36. Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K., Wierstra, D.: Matching networks for one shot learning. In: NIPS, pp. 3630–3638 (2016)
37. Wang, P., Han, J., Li, C., Pan, R.: Logic attention based neighborhood aggregation for inductive knowledge graph embedding. In: AAAI, pp. 7152–7159. AAAI Press (2019)
38. Wang, R., et al.: Learning to sample and aggregate: few-shot reasoning over temporal knowledge graphs. In: NeurIPS (2022)
39. Wu, J., Cao, M., Cheung, J.C.K., Hamilton, W.L.: Temp: temporal message passing for temporal knowledge graph completion. In: EMNLP (1), pp. 5730–5746. Association for Computational Linguistics (2020)

40. Xiong, W., Yu, M., Chang, S., Guo, X., Wang, W.Y.: One-shot relational learning for knowledge graphs. In: EMNLP, pp. 1980–1990. Association for Computational Linguistics (2018)
41. Xu, C., Chen, Y., Nayyeri, M., Lehmann, J.: Temporal knowledge graph completion using a linear temporal regularizer and multivector embeddings. In: NAACL-HLT, pp. 2569–2578. Association for Computational Linguistics (2021)
42. Xu, C., Nayyeri, M., Alkhoury, F., Yazdi, H.S., Lehmann, J.: Tero: a time-aware knowledge graph embedding via temporal rotation. In: COLING, pp. 1583–1593. International Committee on Computational Linguistics (2020)
43. Yang, B., Yih, W., He, X., Gao, J., Deng, L.: Embedding entities and relations for learning and inference in knowledge bases. In: ICLR (Poster) (2015)
44. Zhang, F., Zhang, Z., Ao, X., Zhuang, F., Xu, Y., He, Q.: Along the time: timeline-traced embedding for temporal knowledge graph completion. In: CIKM, pp. 2529–2538. ACM (2022)
45. Zhang, Y., Dai, H., Kozareva, Z., Smola, A.J., Song, L.: Variational reasoning for question answering with knowledge graph. In: AAAI, pp. 6069–6076. AAAI Press (2018)

A Implementation Details

All experiments are implemented with PyTorch on a single NVIDIA A40 with 48GB memory. We search hyperparameters following Table 1. For each dataset, we do 108 trials to try different hyperparameter settings. We run 1000 episodes for each trail and compare their meta-validation results. We choose the setting leading to the best meta-validation result and take it as the best hyperparameter setting. The best hyperparameter setting is reported in Table 2. Our time-aware Transformer uses two heads and two attention layers for all experiments. The results of FITCARL is the average of five runs. The GPU memory usage, training time and the number of parameters are presented in Table 3, 4 and 5, respectively. For all datasets, we use all meta-training entities $\mathcal{E}'_{\text{meta-train}}$ as the considered unseen entities in each meta-training task T. This also applies during meta-validation and meta-test, where all the entities in $\mathcal{E}'_{\text{meta-valid}}/\mathcal{E}'_{\text{meta-test}}$ are considered appearing simultaneously in one evaluation task. All the datasets are taken from FILT's official repository[1]. We also take the pre-trained representations from it for our experiments. During evaluation, we follow previous RL-based TKG reasoning models TITer and CluSTeR and use beam search for answer searching. The beam size is 100 for all experiments.

We implement TITer with its official code[2]. We give it the whole background graph $\mathcal{G}_{\text{back}}$ as well as all meta-training quadruples $\mathbb{T}_{\text{meta-train}}$ for training. During meta-validation and meta-test, it is further given support quadruples for predicting the query quadruples.

Table 1: Hyperparameter searching strategy.

Hyperparameter	**Search Space**
Embedding Size d	{100, 200}
Sampled Action Space Size	{25, 50, 100}
Search Step L	{3, 4}
Regularizer Coefficient η	{1e-11, 1e-9, 1e-7}
Margin of Reward θ	{1, 5, 10}

B Difference between TKGC and TKG forecasting

Assume we have a TKG $\mathcal{G} = \{(s, r, o, t)|s, o \in \mathcal{E}, r \in \mathcal{R}, t \in \mathcal{T}\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$, where $\mathcal{E}$, $\mathcal{R}$, $\mathcal{T}$ denote a finite set of entities, relations and timestamps, respectively. We define the TKG forecasting task (also known as TKG extrapolation) as follows. Assume we have an LP query $(s_q, r_q, ?, t_q)$ (or $(?, r_q, o_q, t_q)$)

[1] https://github.com/Jasper-Wu/FILT
[2] https://github.com/JHL-HUST/TITer

Table 2: Best hyperparameter settings.

Datasets	**ICEWS14-OOG**	**ICEWS18-OOG**	**ICEWS0515-OOG**
Hyperparameter			
Embedding Size d	100	100	100
Sampled Action Space Size	50	50	50
Search Step L	3	3	3
Regularizer Coefficient η	1e-9	1e-9	1e-9
Margin of Reward θ	5	5	5

Table 3: GPU memory usage (MB).

Datasets	**ICEWS14-OOG**		**ICEWS18-OOG**		**ICEWS0515-OOG**	
	GPU Memory		GPU Memory		GPU Memory	
Model	1-S	3-S	1-S	3-S	1-S	3-S
FITCARL	10729	11153	14761	15419	14765	15475

Table 4: Training time (min).

Datasets	**ICEWS14-OOG**		**ICEWS18-OOG**		**ICEWS0515-OOG**	
	Time		Time		Time	
Model	1-S	3-S	1-S	3-S	1-S	3-S
FITCARL	225	85	305	764	1059	297

Table 5: Number of parameters.

Datasets	**ICEWS14-OOG**		**ICEWS18-OOG**		**ICEWS0515-OOG**	
	# Param		# Param		# Param	
Model	1-S	3-S	1-S	3-S	1-S	3-S
FITCARL	8271206	8271410	14633206	10006710	9615206	9615410

derived from a query quadruple (s_q, r_q, o_q, t_q). TKG forecasting aims to predict the missing entity in the LP query, given the observed **past** TKG facts $\mathcal{O} = \{(s_i, r_i, o_i, t_i) | t_i < t_q\}$. Such temporal restriction is not imposed in TKGC (also known as TKG interpolation), where the observed TKG facts from any timestamp, including t_q and the timestamps after t_q, can be used for prediction.

TITer is designed for TKG forecasting, therefore it only performs its RL search process in the direction pointing at the past. This leads to a great loss of information along the whole time axis. TITer also does not use a meta-learning framework for adapting to the few-shot setting, which is also a reason for its weak performance on TKG few-shot OOG LP. Please refer to the papers studying TKG forecasting for more details.

C Ablation E Details

We describe here how we change equations in ablation E to build a model variant without using any temporal information. First, we change the action space sampling method to random sample, which corresponds to ablation A1. This means we do not use temporal information to compute time-adaptive sampling probabilities. Next, we neglect the last term in Equation 2 of the main paper. It thus becomes

$$\begin{aligned} \mathrm{att}_{u,v} &= \frac{\exp(\alpha_{u,v})}{\sum_{k=1}^{K+1} \exp(\alpha_{u,k})}, \\ \alpha_{u,v} &= \frac{1}{\sqrt{d}} (\mathbf{W}_{TrQ} \mathbf{h}_u)^\top (\mathbf{W}_{TrK} \mathbf{h}_v), \end{aligned} \tag{1}$$

which corresponds to ablation D. Finally, we remove every term in all equations containing time-difference representations. For a node (e, t), its representation becomes $\mathbf{h}_e$. Thus, Equation 3 of the main paper becomes

$$\begin{aligned} \mathbf{h}_{\mathrm{hist}^{(l)}} &= \mathrm{GRU}\left((\mathbf{h}_{r^{(l)}} \| \mathbf{h}_{e^{(l)}}), \mathbf{h}_{\mathrm{hist}^{(l-1)}} \right), \\ \mathbf{h}_{\mathrm{hist}^{(0)}} &= \mathrm{GRU}\left(\left(\mathbf{h}_{r_{\mathrm{dummy}}} \| \mathbf{h}_{e'} \right), \mathbf{0} \right). \end{aligned} \tag{2}$$

Equation 4 of the main paper becomes

$$\begin{aligned} \mathbf{h}_{\mathrm{hist}^{(l)},q|a} &= \mathrm{att}_{\mathrm{hist}^{(l)},a} \cdot \bar{\mathbf{h}}_{\mathrm{hist}^{(l)}} + \mathrm{att}_{q,a} \cdot \bar{\mathbf{h}}_q, \\ \bar{\mathbf{h}}_{\mathrm{hist}^{(l)}} &= {\mathbf{W}_1}^\top \mathbf{h}_{\mathrm{hist}^{(l)}}, \quad \bar{\mathbf{h}}_q = {\mathbf{W}_2}^\top \left(\mathbf{h}_{r_q} \| \mathbf{h}_{e'} \right). \end{aligned} \tag{3}$$

Equation 5 and 6 of the main paper become

$$\mathrm{att}_{\mathrm{hist}^{(l)},a} = \frac{\exp(\phi_{\mathrm{hist}^{(l)},a})}{\exp(\phi_{\mathrm{hist}^{(l)},a}) + \exp(\phi_{q,a})}, \mathrm{att}_{q,a} = \frac{\exp(\phi_{q,a})}{\exp(\phi_{\mathrm{hist}^{(l)},a}) + \exp(\phi_{q,a})}, \tag{4}$$

where

$$\begin{aligned} \phi_{\mathrm{hist}^{(l)},a} = \bar{\mathbf{h}}_a^\top \bar{\mathbf{h}}_{\mathrm{hist}^{(l)}}, \ \phi_{q,a} = \bar{\mathbf{h}}_a^\top \bar{\mathbf{h}}_q, \\ \bar{\mathbf{h}}_a = {\mathbf{W}_3}^\top \left(\mathbf{h}_{r_a} \| \mathbf{h}_{e_a} \right). \end{aligned} \tag{5}$$

Algorithm 1: FITCARL Meta-Training

Input: Meta-training entities $\mathcal{E}'_{\text{meta-train}}$, background TKG $\mathcal{G}_{\text{back}}$, shot size K

for episode = *1: M* **do**
  for $e' \in \mathcal{E}'_{meta\text{-}train}$ **do**
    Sample a support set $Sup_{e'}$ and a query set $Que_{e'}$
    Learn meta-representations $\{\mathbf{h}^i_{e'}\}_{i=1}^K$
  for $e' \in \mathcal{E}'_{meta\text{-}train}$ **do**
    for *query* $\in Que_{e'}$ **do**
      Derive LP query q from *query*
      Compute $\mathbf{h}_{e'}$ using time-aware Transformer `// Section 4.1`
      Initialize $s^{(0)} \leftarrow (e', t_q, e', r_q, t_q)$
      $\{R(s^{(l)}, a^{(l)})\}_{l=0}^{L-1}, \{\mathcal{L}^{(l)}_{\text{KL}|q}\}_{l=0}^{L-1}, \{\pi(a^{(l)}|s^{(l)})\}_{l=0}^{L-1} \leftarrow \text{Search}(L, s^{(0)})$
  Compute loss $\mathcal{L}_T$ `// Equation 12`
  Update model parameters using gradient of $\bigtriangledown\mathcal{L}_T$

Procedure Search($L, s^{(0)}$)
  for l = *0:L-1* **do**
    Sample action space $\mathcal{A}^{(l)}$ from all observed outgoing edges of node $(e^{(l)}, t^{(l)})$
    Compute $P(a|s^{(l)}, \text{hist}^{(l)})$ and $\text{conf}_{a|q}$ for $a \in \mathcal{A}^{(l)}$ `// Equation 7, 8`
    Compute $\pi(a|s^{(l)})$ for each $a \in \mathcal{A}^{(l)}$ `// Equation 9`
    Compute $\mathcal{L}^{(l)}_{\text{KL}|q}$ `// Equation 11`
    Sample $a^{(l)} = (e_{a^{(l)}}, r_{a^{(l)}}, t_{a^{(l)}})$ according to policy π
    Compute reward $R(s^{(l)}, a^{(l)})$
    Execute $a^{(l)}$, agent transfers to state $s^{(l+1)} = (e^{(l+1)}, r^{(l+1)}, e', r_q, t_q)$
  return $\{R(s^{(l)}, a^{(l)})\}_{l=0}^{L-1}, \{\mathcal{L}^{(l)}_{\text{KL}|q}\}_{l=0}^{L-1}, \{\pi(a^{(l)}|s^{(l)})\}_{l=0}^{L-1}$

Equation 8 of the main paper becomes

$$\text{conf}_{a|q} = \frac{\exp(\psi_{a|q})}{\sum_{a' \in \mathcal{A}^{(l)}} \exp(\psi_{a'|q})}, \text{ where } \psi_{a|q} = \mathcal{W} \times_1 \mathbf{h}_{e'} \times_2 \mathbf{h}_{r_q} \times_3 \mathbf{h}_{e_a}. \tag{6}$$

The other equations remain unchanged. To this end, we create a model variant that uses no temporal information.

D Evaluation Metrics

We use two evaluation metrics, i.e., mean reciprocal rank (MRR) and Hits@1/3/10. For every LP query q, we compute the rank $rank_q$ of the ground truth missing entity. We define MRR as: $\frac{1}{\sum_{e' \in \mathcal{E}'_{\text{meta-test}}} |Que_{e'}|} \sum_{e' \in \mathcal{E}'_{\text{meta-test}}} \sum_{q \in Que_{e'}} \frac{1}{rank_q}$. Hits@1/3/10 denote the proportions of the predicted links where ground truth missing entities are ranked as top 1, top3, top10, respectively. We also use the filtered setting proposed in previous works for fairer evaluation.

E Meta-Training Algorithm of FITCARL

We train FITCARL with episodic training. We present our meta-training process in Algorithm 1.

Chapter 5

Zero-Shot Relational Learning on Temporal Knowledge Graphs with Large Language Models

This chapter contains the publication

> **Zifeng Ding***, Heling Cai*, Jingpei Wu, Yunpu Ma, Ruotong Liao, Bo Xiong, Volker Tresp. zrLLM: Zero-Shot Relational Learning on Temporal Knowledge Graphs with Large Language Models. In *2024 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, 2024. *Equal Contribution. https://aclanthology.org/2024.naacl-long.104.pdf

zrLLM: Zero-Shot Relational Learning on Temporal Knowledge Graphs with Large Language Models

Zifeng Ding[*1,2], **Heling Cai**[*1], **Jingpei Wu**[1], **Yunpu Ma**[1,3],
Ruotong Liao[1,3], **Bo Xiong**[†4], **Volker Tresp**[†1]
[1]LMU Munich [2]Siemens AG
[3]Munich Center for Machine Learning (MCML) [4]University of Stuttgart
{zifeng.ding, heling.cai}@campus.lmu.de
{jingpei.wu,liao,tresp}@dbs.ifi.lmu.de,
cognitive.yunpu@gmail.com, bo.xiong@ki.uni-stuttgart.de

Abstract

Modeling evolving knowledge over temporal knowledge graphs (TKGs) has become a heated topic. Various methods have been proposed to forecast links on TKGs. Most of them are embedding-based, where hidden representations are learned to represent knowledge graph (KG) entities and relations based on the observed graph contexts. Although these methods show strong performance on traditional TKG forecasting (TKGF) benchmarks, they face a strong challenge in modeling the unseen zero-shot relations that have no prior graph context. In this paper, we try to mitigate this problem as follows. We first input the text descriptions of KG relations into large language models (LLMs) for generating relation representations, and then introduce them into embedding-based TKGF methods. LLM-empowered representations can capture the semantic information in the relation descriptions. This makes the relations, whether seen or unseen, with similar semantic meanings stay close in the embedding space, enabling TKGF models to recognize zero-shot relations even without any observed graph context. Experimental results show that our approach helps TKGF models to achieve much better performance in forecasting the facts with previously unseen relations, while still maintaining their ability in link forecasting regarding seen relations.

1 Introduction

Knowledge graphs (KGs) represent world knowledge with a collection of facts in the form of (s, r, o) triples, where in each fact, s, o are the subject and object entities and r is the relation between them. Temporal knowledge graphs (TKGs) are introduced by further specifying the time validity. Each TKG fact is denoted as a quadruple (s, r, o, t), where t (a timestamp or a time period) provides temporal constraints. Since world knowledge is ever-evolving, TKGs are more expressive in representing dynamic factual information and have drawn increasing interest in a wide range of downstream tasks, e.g., natural language question answering over TKGs (Saxena et al., 2021; Ding et al., 2023b).

In recent years, there has been an increasing number of works paying attention to forecasting future facts in TKGs, i.e., TKG forecasting (TKGF) or TKG extrapolated link prediction (LP). Most of them are embedding-based, where entity and relation representations are learned with the help of the observed graph contexts. Although traditional embedding-based TKGF methods show impressive performance on current benchmarks, they share a common limitation. In these works, models are trained on the TKG facts regarding a set of relations $\mathcal{R}$, and they are only expected to be evaluated on the facts containing the relations in $\mathcal{R}$. They cannot handle any zero-shot unseen relation $r \notin \mathcal{R}$ because no graph context regarding unseen relations exists in the training data and thus no reasonable relation representations can be learned. In the forecasting scenario, as time flows, new knowledge is constantly introduced into a TKG, making it expand in size. This increases the chance of encountering newly-emerged relations, and therefore, it is meaningful to improve embedding-based TKGF methods to be more adaptive to zero-shot relations.

With the increasing scale of pre-trained language models (LMs), LMs become large LMs (LLMs). Recent studies find that LLMs have shown emerging abilities in various aspects (Wei et al., 2022) and can be taken as strong semantic knowledge bases (KBs) (Petroni et al., 2019). Inspired by this, we try to enhance the performance of embedding-based TKGF models over zero-shot relations with an approach consisting of the following three steps: (1) Based on the relation text descriptions provided

[*]Equal contribution.
[†]Corresponding author.

Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), pages 1877–1895

in TKG datasets, we first use an LLM to produce an enriched relation description (ERD) with more details for each KG relation (Sec. 3.1). (2) We then generate the relation representations by leveraging another LLM, i.e., T5-11B (Raffel et al., 2020). We input ERDs into T5's encoder and transform its output into relation representations of TKGF models (Sec. 3.1). (3) We design a relation history learner (RHL) to capture historical relation patterns, where we leverage LLM-empowered relation representations to better reason over zero-shot relations (Sec. 3.2). With these steps, we align the natural language space provided by LLMs to the embedding space of TKGF models, rather than letting models learn relation representations solely from observed graph contexts. Even without any observed associated facts, zero-shot relations can be represented with LLM-empowered representations that contain semantic information. We term our approach as zrLLM since it is used to enhance **z**ero-shot **r**elational learning on TKGF models by using **LLM**s.

We experiment zrLLM on seven recent embedding-based TKGF models and evaluate them on three new datasets constructed specifically for studying TKGF regarding zero-shot relations. Our contribution is three-folded: (1) To the best of our knowledge, this is the first work trying to study zero-shot relational learning in TKGF. (2) We design an LLM-empowered approach zrLLM and manage to enhance various recent embedding-based TKGF models in reasoning over zero-shot relations. (3) Experimental results show that zrLLM helps to substantially improve all considered TKGF models' abilities in forecasting the facts containing unseen zero-shot relations, while still maintaining their ability in link forecasting regarding seen relations.

2 Preliminaries

2.1 Related Work

Traditional TKG Forecasting Methods. Traditional TKGF methods are trained to forecast the facts containing the KG relations (and entities) seen in the training data, regardless of the case where zero-shot relations (or entities) appear as new knowledge arrives. These methods can be categorized into two types: embedding-based and rule-based. Embedding-based methods learn hidden representations of KG relations and entities, and perform link forecasting based on them. Most existing embedding-based methods, e.g., (Jin et al., 2020; Han et al., 2021b; Li et al., 2021b, 2022; Liu et al., 2023), learn evolutional entity and relation representations from the historical TKG information by jointly employing graph neural networks (Kipf and Welling, 2017) and recurrent neural structures, e.g., GRU (Cho et al., 2014). Some other approaches (Han et al., 2021a; Sun et al., 2021; Li et al., 2021a) start from each LP query[1] and traverse the temporal history in a TKG to search for the prediction answer. There also exist some methods, e.g., (Zhu et al., 2021; Xu et al., 2023b), that achieve forecasting based on the appearance of historical facts. Compared with embedding-based TKGF approaches, rule-based TKGF has still not been extensively explored. One popular rule-based TKGF method is TLogic (Liu et al., 2022). It extracts temporal logical rules from TKGs and uses a symbolic reasoning module for LP. Based on it, ALRE-IR (Mei et al., 2022) proposes an adaptive logical rule embedding model to encode temporal logical rules into rule representations. This makes ALRE-IR both a rule-based and an embedding-based method. Rule-based TKGF methods have strong ability in reasoning over zero-shot unseen entities connected by the seen relations, however, they are not able to handle unseen relations since the learned rules are strongly bounded by the observed relations.

Inductive Learning on TKGs. Inductive learning on TKGs refers to developing models that can handle the relations and entities unseen in the training data. Most of TKG inductive learning methods are based on few-shot learning, e.g., (Ding et al., 2022; Zhang et al., 2019; Ding et al., 2023c; Mirtaheri et al., 2021; Ding et al., 2023a,a; Ma et al., 2023). They first compute inductive representations of newly-emerged entities or relations based on K-associated facts (K is a small number, e.g., 1 or 3), and then use them to predict other facts regarding few-shot elements. One limitation of these works is that the inductive representations cannot be learned without the K-shot examples, making them hard to solve the zero-shot problems. Different from few-shot learning methods, SST-BERT (Chen et al., 2023a) pre-trains a time-enhanced BERT (Devlin et al., 2019) and proves its inductive power over unseen entities but has not shown its ability in reasoning zero-shot relations. Another

[1] A TKG LP query is denoted as $(s, r, ?, t)$ (object prediction query) or $(?, r, o, t)$ (subject prediction query).

recent work MTKGE (Chen et al., 2023b) is able to concurrently deal with both unseen entities and relations. However, it requires a support graph containing a substantial number of data examples related to the unseen entities and relations, which is far from the zero-shot setting.

TKG Reasoning with Language Models. Recently, more and more works have introduced LMs into TKG reasoning. SST-BERT pre-trains an LM on a corpus of training TKGs for fact reasoning. ECOLA (Han et al., 2023) aligns facts with additional fact-related texts and enhances TKG reasoning with BERT-encoded language representations. PPT (Xu et al., 2023a) converts TKGF into the pre-trained LM masked token prediction task and finetunes a BERT for TKGF. Apart from them, one recent work (Lee et al., 2023) explores in-context learning (ICL) (Brown et al., 2020) with LLMs to predict future facts without finetuning. Another recent work GenTKG (Liao et al., 2023) finetunes Llama2-7B (Touvron et al., 2023), and let it directly generate the LP answer in TKGF.

Although previous works have shown success of LMs in TKG reasoning, they have limitations: (1) None of them has studied whether LMs, in particular LLMs, can be used to better reason zero-shot relations. (2) By only using ICL, LLMs are beaten by traditional TKGF methods in performance (Lee et al., 2023). The performance can be greatly improved by finetuning LLMs (Liao et al., 2023), but finetuning LLMs requires huge computational resources. (3) Since LMs are pre-trained with a huge corpus originating from diverse information sources, it is inevitable that they have already seen the world knowledge before they are used to solve TKG reasoning tasks. Most popular TKGF benchmarks are constructed with the facts before 2020 (ICEWS14/18/05-15 (Jin et al., 2020)). The facts inside are based on the world knowledge before 2019, which means LMs might have encountered them in their training corpus, posing a threat of information leak to the LM-driven TKG reasoning models. To this end, we (1) draw attention to studying the impact of LLMs on zero-shot relational learning in TKGs; (2) make a compromise between performance and computational efficiency by not finetuning LMs or LLMs but adapting the LLM-provided semantic information to non-LM-based TKGF methods; (3) construct new benchmarks whose facts are all happening from 2021 to 2023, which avoids the threat of information leak when we utilize T5-11B that was released in 2020.

2.2 Definitions and Task Formulation

Definition 1 (TKG). Let $\mathcal{E}$, $\mathcal{R}$, $\mathcal{T}$ denote a set of entities, relations and timestamps, respectively. A TKG $\mathcal{G} = \{(s, r, o, t)\} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$ is a set of temporal facts where each fact is represented with a fact quadruple (s, r, o, t).

Definition 2 (TKG Forecasting). Assume we have a ground truth TKG $\mathcal{G}_{gt}$ that contains all the true facts. Given an LP query $(s_q, r_q, ?, t_q)$ (or $(o_q, r_q, ?, t_q)$), TKGF requires the models to predict the missing object o_q (or subject s_q) based on the facts observed before the query timestamp t_q, i.e., $\mathcal{O} = \{(s, r, o, t_i) \in \mathcal{G}_{gt} | t_i < t_q\}$.

Definition 3 (Zero-Shot TKG Forecasting). Assume we have a ground truth TKG $\mathcal{G}_{gt} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E} \times \mathcal{T}$, where $\mathcal{R}$ can be split into seen $\mathcal{R}_{se}$ and unseen $\mathcal{R}_{un}$ relations ($\mathcal{R} = \mathcal{R}_{se} \cup \mathcal{R}_{un}, \mathcal{R}_{se} \cap \mathcal{R}_{un} = \emptyset$). Given an LP query $(s_q, r_q, ?, t_q)$ (or $(o_q, r_q, ?, t_q)$) whose query relation $r_q \in \mathcal{R}_{un}$, models are asked to predict the missing object o_q (or subject s_q) based on the facts $\mathcal{O} = \{(s, r_i, o, t_i) \in \mathcal{G}_{gt} | t_i < t_q, r_i \in \mathcal{R}_{se}\}$ containing seen relations and happening before t_q.

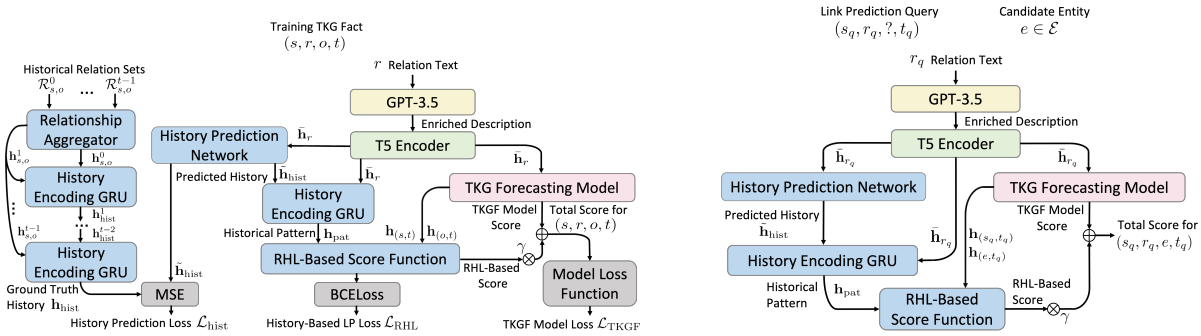
3 zrLLM

zrLLM is coupled with TKGF models to enhance zero-shot ability. It uses GPT-3.5 to generate enriched relation descriptions (ERDs) based on the relation texts provided by TKG datasets. It then inputs the ERDs into the encoder of T5-11B and aligns its output to TKG embedding space. zrLLM also employs a relation history learner (RHL) to capture the temporal relation patterns based on the LLM-based relation representations, which further promotes embedding space alignment. See Fig. 1 for illustration of zrLLM-enhanced TKGF models.

3.1 Represent KG Relations with LLMs

Generate Text Representations with ERDs. We generate text representations with T5-11B based on the textual descriptions of KG relations. Since the relation texts provided by TKG datasets are short and concise, we use GPT-3.5[2] to enrich them for more comprehensive semantics. Our prompt for description enrichment is depicted in Fig. 2. For each relation, we treat the combination of its relation text and LLM-generated explanation as its ERD. See Table 1 for two enrichment examples.

[2]https://platform.openai.com/docs/model-index-for-researchers



(a) Training pipeline of zrLLM-enhanced model.

(b) Evaluation pipeline of zrLLM-enhanced model.

Figure 1: Illustration of zrLLM-enhanced TKGF models. RHL-related components are marked in blue. RHL works differently in training and evaluation. During training, since we know both entities (s, o in 1a) in the training fact, we can find the ground truth historical relations between them over time. We train a history prediction network (HPN) that aims to generate the relation history between two entities given their current relation (r). During evaluation, we directly use the trained HPN to infer the relation history. See Sec. 3 for details.

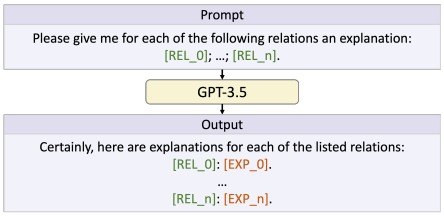


Figure 2: Prompting GPT-3.5 for ERDs. [REL_0], ..., [REL_n] are the dataset provided relation texts for a batch of n KG relations. [EXP_0], ..., [EXP_n] are the LLM-generated explanations. [REL:_0]: [EXP_0], ..., [REL:_n]: [EXP_n] are taken as ERDs. See Appendix A for an expanded version of this figure.

KG Relation Text	Enriched Relation Description
Engage in negotiation	Engage in negotiation: This indicates a willingness to participate in discussions or dialogues with the aim of reaching agreements or settlements on various issues.
Praise or endorse	Praise or endorse: This signifies a positive evaluation or approval of another entity's actions, policies, or behavior. It is a form of expressing support or admiration.

Table 1: Relation description enrichment examples.

We then input the ERDs into T5-11B. T5 is with an encoder-decoder architecture, where its encoder can be taken as a module that helps to understand the text input and the decoder is solely used for text generation. We take the output of T5-11B's encoder, i.e., the hidden representations, for our downstream task. Note that although ERDs are produced by GPT-3.5 who is trained with the corpus until the end of 2021, the representations used for TKGF are generated only with T5-11B, preventing information leak. Also, through our prompt, GPT-3.5 does not know our underlying task of TKGF. We manually check the ERDs generated by GPT-3.5 and make sure that GPT-3.5 generates relation explanations solely from the semantic perspective and no world knowledge is contained in its output.

Align Text Representations to TKG Embedding Space. For each KG relation r, the T5-generated text representation is a parameter matrix $\bar{\mathbf{H}}_r \in \mathbb{R}^{L \times d_w}$. L is the length of the Transformers (Vaswani et al., 2017) in T5 and d_w is the embedding size of each word output from T5 encoder. The l^{th} row in $\bar{\mathbf{H}}_r$ is the T5 encoded hidden representation $\mathbf{w}_l \in \mathbb{R}^{d_w}$ of the l^{th} word in the enriched description. To align $\bar{\mathbf{H}}_r$ to an embedding-based TKGF model, we first use a multi-layer perceptron (MLP) to map each $\mathbf{w}_l$ to the dimension of the TKGF model's relation representation.

$$\mathbf{w}'_l = \text{MLP}(\mathbf{w}_l), \text{where } \mathbf{w}'_l \in \mathbb{R}^d. \tag{1}$$

Then we learn a representation of r's ERD $\bar{\mathbf{h}}_r$ using a GRU.

$$\begin{aligned} &\bar{\mathbf{h}}_r^{(l)} = \text{GRU}(\mathbf{w}'_l, \bar{\mathbf{h}}_r^{(l-1)});\ \bar{\mathbf{h}}_r^{(0)} = \mathbf{w}'_0, \\ &\bar{\mathbf{h}}_r = \bar{\mathbf{h}}_r^{(L-1)}. \end{aligned} \tag{2}$$

$l \in [1, L-1]$. $\bar{\mathbf{h}}_r$ contains semantic information from ERD, and therefore, we can view it as an LM-based relation representation. We substitute the relation representations of TKGF models with LM-based representations for semantics integration. Note that we fix the values of every $\bar{\mathbf{H}}_r$ to keep the LLM-provided semantic information intact. This is because we do not want the relation representations to lay excessive emphasis on the training data where zero-shot relations never appear. We want the models to maximally benefit from the semantic information for better generalization power. The textual descriptions of the relations with close meanings will show similar semantics. Since for each relation r, $\bar{\mathbf{H}}_r$ is generated based on

r's ERD, the relations with close meanings will naturally lead to highly correlated text representations, building connections on top of the natural language space regardless of the observed TKG data.

3.2 Improving Text-to-Graph Alignment with Relation History Learner

As the relationship between two entities evolves through time, it follows certain temporal patterns. For example, the fact (*China*, *Sign formal agreement*, *Nicaragua*, 2022-01-10) happens after (*China*, *Grant diplomatic recognition*, *Nicaragua*, 2022-01-04), implying that an agreement will be signed after showing diplomatic recognition. These temporal patterns are entity-agnostic and can reflect the dynamic relationship between any two entities over time. To this end, we develop RHL, aiming to capture such patterns. RHL leverages the LLM-based relation representations for pattern modeling, which further promotes the alignment between the text and TKG embedding spaces.

Assume we have a training fact (s, r, o, t), we search for the historical facts $\mathcal{G}_{s,o}^{<t}$ containing s and o before t, and group these facts according to their timestamps, i.e., $\mathcal{G}_{s,o}^{<t} = \{\mathcal{G}_{s,o}^{0}, ..., \mathcal{G}_{s,o}^{t-1}\}$. The searched facts with the same timestamp are put into the same group. For each group, we pick out the relations of all its facts and form a relation set, e.g., $\mathcal{R}_{s,o}^{0}$ is derived from $\mathcal{G}_{s,o}^{0}$. s and o's relationship at t_i ($t_i \in [0, t-1]$) is computed with an aggregator

$$\mathbf{h}_{s,o}^{t_i} = \textstyle\sum_m a_m \bar{\mathbf{h}}_{r_m};\ a_m = \text{softmax}(\bar{\mathbf{h}}_{r_m}^{\top} \text{MLP}_{\text{agg}}(\bar{\mathbf{h}}_r)). \quad (3)$$

$r_m \in \mathcal{R}_{s,o}^{t_i}$ denotes a relation bridging s and o at t_i. If $\mathcal{R}_{s,o}^{t_i} = \emptyset$, we set $\mathbf{h}_{s,o}^{t_i}$ to a dummy embedding $\mathbf{h}_{\text{dum}}$. To capture the historical relation dynamics, we use another GRU, i.e., GRU_{RHL}.

$$\begin{aligned} &\mathbf{h}_{\text{hist}}^{t_i} = \text{GRU}_{\text{RHL}}(\mathbf{h}_{s,o}^{t_i}, \mathbf{h}_{\text{hist}}^{t_i-1});\ \mathbf{h}_{\text{hist}}^{0} = \mathbf{h}_{s,o}^{0}, \\ &\mathbf{h}_{\text{hist}} = \mathbf{h}_{\text{hist}}^{t-1}. \end{aligned} \quad (4)$$

$\mathbf{h}_{\text{hist}}$ is taken as the encoded relation history until $t-1$. Note that during evaluation, TKGF asks models to predict the missing object of each LP query $(s_q, r_q, ?, t_q)$, which means we do not know which two entities should be used for historical fact searching[3]. To solve this problem, during training, we train another history prediction network (HPN) that aims to directly infer the relation history given the training fact relation r.

$$\tilde{\mathbf{h}}_{\text{hist}} = \alpha \text{MLP}_{\text{hist}}(\bar{\mathbf{h}}_r) + \bar{\mathbf{h}}_r. \quad (5)$$

Here, α is a hyperparameter scalar and MLP_{hist} is an MLP. $\tilde{\mathbf{h}}_{\text{hist}}$ is the predicted relation history given r. Since we want $\tilde{\mathbf{h}}_{\text{hist}}$ to represent the ground truth relation history, we use a mean square error (MSE) loss to constrain it to be close to $\mathbf{h}_{\text{hist}}$.

$$\mathcal{L}_{\text{hist}} = \text{MSE}(\tilde{\mathbf{h}}_{\text{hist}}, \mathbf{h}_{\text{hist}}). \quad (6)$$

In this way, during evaluation, we can directly use Eq. 5 to generate a meaningful $\tilde{\mathbf{h}}_{\text{hist}}$ for further computation. Given $\tilde{\mathbf{h}}_{\text{hist}}$, we do one more step in GRU_{RHL} to capture the r-related relation pattern.

$$\mathbf{h}_{\text{pat}} = \text{GRU}_{\text{RHL}}(\bar{\mathbf{h}}_r, \tilde{\mathbf{h}}_{\text{hist}}). \quad (7)$$

$\mathbf{h}_{\text{pat}}$ can be viewed as a hidden representation containing comprehensive information of temporal relation patterns. Inspired by TuckER (Balazevic et al., 2019), we compute an RHL-based score for the training target (s, r, o, t) as

$$\phi((s, r, o, t)) = \mathcal{W} \times_1 \mathbf{h}_{(s,t)} \times_2 \mathbf{h}_{\text{pat}} \times_3 \mathbf{h}_{(o,t)}, \quad (8)$$

where $\mathcal{W} \in \mathbb{R}^{d \times d \times d}$ is a learnable core tensor and $\times_1, \times_2, \times_3$ are three operators indicating the tensor product in three different modes (details in (Balazevic et al., 2019)). $\mathbf{h}_{(s,t)}$ and $\mathbf{h}_{(o,t)}$ are the time-aware entity representations of s and o computed by TKGF model, respectively. RHL-based score can be viewed as measuring how much two entities match the relation pattern generated by the relation history. We couple this score with the score computed by the original TKGF model $\phi'((s, r, o, t))$ and use the total score for LP.

$$\phi_{\text{total}}((s, r, o, t)) = \phi'((s, r, o, t)) + \gamma \phi((s, r, o, t)). \quad (9)$$

γ is a hyperparameter. RHL enables models to make decisions by additionally considering the temporal relation patterns. Note that patterns are captured with LLM-empowered relation representations that contain rich semantic information. This guarantees RHL to generalize well to zero-shot relations. See App. I for explanations.

3.3 Parameter Learning and Evaluation

We let zrLLM be co-trained with TKGF model. Assume f is a TKGF model's loss function, e.g., cross-entropy, where f takes a fact quadruple's

[3] We can indeed couple s_q with every candidate entity $e \in \mathcal{E}$ and search for their historical facts. But it requires huge computational resources and greatly harms model's scalability.

score computed by model's score function ϕ' and returns a loss for this fact. We input the quadruple score computed with Eq. 9 into f to let TKGF models better learn the parameters in RHL.

$$\mathcal{L}_{\text{TKGF}} = \frac{1}{|\mathcal{G}_{\text{train}}|} \sum_{\lambda \in \mathcal{G}_{\text{train}}} f(\phi_{\text{total}}(\lambda)), \tag{10}$$

where λ denotes a fact quadruple $(s, r, o, t) \in \mathcal{G}_{\text{train}}$ in the training set $\mathcal{G}_{\text{train}}$. Besides, we also employ an additional binary cross-entropy loss $\mathcal{L}_{\text{RHL}}$ directly on the RHL-based score

$$\begin{aligned} \mathcal{L}_{\text{RHL}} &= \frac{1}{N} \sum_{\lambda} \sum_{e \in \mathcal{E}} \mathcal{L}_{\text{RHL}}^{\lambda,e}; \\ \mathcal{L}_{\text{RHL}}^{\lambda,e} &= -y_{\lambda'} \log(\phi(\lambda')) - (1 - y_{\lambda'}) \log(1 - \phi(\lambda')). \end{aligned} \tag{11}$$

$N = |\mathcal{G}_{\text{train}}| \times |\mathcal{E}|$. λ' is a perturbed fact by switching the object of λ to any $e \in \mathcal{E}$ and $y_{\lambda'}$ is its label. If $\lambda' \in \mathcal{G}_{\text{train}}$, then $y_{\lambda'} = 1$, otherwise $y_{\lambda'} = 0$. Finally, we define the total loss $\mathcal{L}_{\text{total}}$ as

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{TKGF}} + \mathcal{L}_{\text{hist}} + \eta \mathcal{L}_{\text{RHL}}. \tag{12}$$

η is a hyperparameter deciding $\mathcal{L}_{\text{RHL}}$'s magnitude. Given our loss, we can also view RHL as a module that does a subtask during training. The subtask is to leverage the relation patterns encoded solely with LLM-based relation representations to perform TKG forecasting, which is parallel to the pipeline of the original TKGF model. This subtask training process helps to improve the embedding space alignment between text and graph representations. During evaluation, for each LP query $(s_q, r_q, ?, t_q)$, we compute scores $\{\phi_{\text{total}}((s_q, r_q, e, t_q))\} | e \in \mathcal{E}\}$ and take the entity with maximum score as the predicted answer. We provide algorithms of training and evaluation in App. D.

4 Experiments

We give details of our new zero-shot TKGF datasets in Sec. 4.1. In Sec. 4.3, we (1) do a comparative study to show how zrLLM improves TKGF models, (2) do ablation studies, (3) compare zrLLM with recent LM-enhanced TKGF models, and (4) do a case study to prove RHL's effectiveness. The implementation code and our proposed zero-shot datasets are in the following page: https://github.com/ZifengDing/zrLLM

4.1 Datasets for Zero-Shot TKGF

As discussed in Sec. 2.1, LM-enhanced TKGF models experience the risk of information leak.

Dataset	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{T}_{\text{train}}\rvert$	$\lvert\mathcal{T}_{\text{eval}}\rvert$	$\lvert\mathcal{R}_{se}\rvert$	$\lvert\mathcal{R}_{un}\rvert$	$\lvert\mathcal{G}_{\text{train}}\rvert$	$\lvert\mathcal{G}_{\text{valid}}\rvert$	$\lvert\mathcal{G}_{\text{test}}\rvert$
ACLED-zero	621	23	20	11	9	14	2,118	931	146
ICEWS21-zero	18,205	253	181	62	130	123	247,764	77,195	1,395
ICEWS22-zero	999	248	181	62	93	155	171,013	47,784	1,956

Table 2: Dataset statistics. Dataset timestamps consist of both training and evaluation timestamps, i.e., $\mathcal{T} = \mathcal{T}_{\text{train}} \cup \mathcal{T}_{\text{eval}}$, $\mathcal{T}_{\text{train}} \cap \mathcal{T}_{\text{eval}} = \emptyset$, $\max(\mathcal{T}_{\text{train}}) < \min(\mathcal{T}_{\text{eval}})$.

To exclude this concern, we construct new benchmark datasets on top of the facts happening after the publication date of T5-11B. We first construct two datasets ICEWS21-zero and ICEWS22-zero based on the Integrated Crisis Early Warning System (ICEWS) (Boschee et al., 2015) KB. ICEWS21-zero contains the facts happening from 2021-01-01 to 2021-08-31, while all the facts in ICEWS22-zero happen from 2022-01-01 to 2022-08-31. Besides, we also construct another dataset ACLED-zero based on another KB: The Armed Conflict Location & Event Data Project (ACLED) (Raleigh et al., 2010). Facts in ACLED-zero take place from 2023-08-01 to 2023-08-31. All the facts in all three datasets are based on social-political events described in English.

Inspired by (Mirtaheri et al., 2021), our dataset construction process consists of the following steps. (1) For each dataset, we first collect all the facts within the time period of interest from the associated KB and then sort them in the temporal order. (2) Then we split the collected facts into two splits, where the first split contains the facts for model training and the second one has all the facts for evaluation. Any fact from the evaluation split happens later than the maximum timestamp of all the facts from the training split. Since we are studying zero-shot relations, we exclude the facts in the evaluation split whose entities do not appear in the training split, to avoid the potential impact of unseen entities. (3) We compute the frequencies of all relations in the evaluation split, and set a frequency threshold (40 for ACLED-zero and ICEWS21-zero, 60 for ICEWS22-zero). (4) We take each relation whose frequency is lower than the threshold as a zero-shot relation, and treat every fact containing it in the evaluation split as zero-shot evaluation data $\mathcal{G}_{\text{test}}$. We exclude the facts associated with zero-shot relations from the training split to ensure that models cannot see these relations during training, and take the rest as the training set $\mathcal{G}_{\text{train}}$. The rest of facts in the evaluation split are taken as the regular evaluation data $\mathcal{G}_{\text{valid}}$. We do validation over $\mathcal{G}_{\text{valid}}$ and test over $\mathcal{G}_{\text{test}}$ because we want to study how models perform over zero-shot relations when

Datasets	ACLED-zero							ICEWS21-zero							ICEWS22-zero						
	Zero-Shot Relations			Seen Relations			Overall	Zero-Shot Relations			Seen Relations			Overall	Zero-Shot Relations			Seen Relations			Overall
Model	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
CyGNet	0.487	0.349	**0.791**	**0.751**	0.663	0.903	0.717	0.120	0.046	0.270	0.254	**0.165**	0.432	0.252	0.211	0.098	0.459	**0.315**	0.198	0.540	0.311
CyGNet+	**0.533**	**0.418**	0.753	**0.751**	**0.664**	**0.906**	**0.723**	**0.201**	**0.103**	**0.415**	**0.258**	0.162	**0.447**	**0.257**	**0.286**	**0.167**	**0.542**	**0.315**	**0.200**	**0.545**	**0.314**
TANGO-T	0.052	0.021	0.101	0.774	0.701	0.900	0.681	0.067	0.031	0.132	**0.283**	**0.190**	**0.470**	**0.279**	0.092	0.042	0.187	**0.363**	0.250	0.579	0.352
TANGO-T+	**0.525**	**0.393**	**0.764**	**0.775**	**0.702**	**0.901**	**0.743**	**0.216**	**0.125**	**0.395**	0.280	0.186	0.466	**0.279**	**0.326**	**0.198**	**0.578**	**0.363**	**0.251**	**0.585**	**0.362**
TANGO-D	0.021	0.003	0.049	**0.777**	**0.701**	**0.907**	0.679	0.012	0.005	0.023	0.266	**0.178**	0.439	0.261	0.011	0.002	0.018	**0.350**	0.227	0.569	0.337
TANGO-D+	**0.491**	**0.348**	**0.791**	0.760	0.678	0.901	**0.725**	**0.212**	**0.122**	**0.400**	**0.268**	0.175	**0.453**	**0.267**	**0.311**	**0.186**	**0.574**	**0.350**	**0.239**	**0.570**	**0.348**
RE-GCN	0.441	0.332	0.718	0.730	**0.653**	0.865	0.693	0.200	0.104	0.379	0.277	0.185	**0.456**	0.276	0.280	0.162	**0.616**	0.354	0.243	0.567	0.351
RE-GCN+	**0.529**	**0.393**	**0.784**	**0.731**	0.650	**0.876**	**0.705**	**0.214**	**0.117**	**0.406**	**0.280**	**0.188**	**0.456**	**0.279**	**0.324**	**0.194**	0.595	**0.357**	**0.244**	**0.573**	**0.356**
TiRGN	0.478	0.330	0.745	**0.754**	0.678	**0.886**	0.718	0.189	0.101	0.368	0.275	0.182	0.457	0.273	0.299	0.169	0.570	0.352	0.239	0.575	0.350
TiRGN+	**0.548**	**0.436**	**0.750**	**0.754**	0.679	0.885	**0.727**	**0.221**	**0.130**	**0.410**	**0.279**	**0.185**	**0.464**	**0.278**	**0.333**	**0.203**	**0.602**	**0.353**	**0.240**	**0.577**	**0.352**
RETIA	0.499	0.360	0.795	0.782	0.701	0.924	0.745	» 120 Hours Timeout							0.302	0.166	0.566	0.356	0.245	0.577	0.354
RETIA+	**0.557**	**0.408**	**0.814**	**0.783**	**0.703**	**0.925**	**0.754**								**0.331**	**0.201**	**0.597**	**0.358**	**0.247**	**0.578**	**0.357**
CENET	0.419	0.297	0.593	0.753	0.682	0.869	0.710	0.205	0.101	0.411	0.288	0.196	0.468	0.287	0.270	0.134	0.544	0.379	0.268	0.599	0.375
CENET+	**0.591**	**0.451**	**0.844**	**0.779**	**0.692**	**0.912**	**0.755**	**0.335**	**0.162**	**0.659**	**0.396**	**0.239**	**0.688**	**0.395**	**0.564**	**0.432**	**0.801**	**0.571**	**0.451**	**0.773**	**0.570**

Table 3: LP results. The best results between each baseline and its zrLLM-enhanced version (model name with "+") are marked in bold. TANGO-T and TANGO-D denote TANGO with TuckER (Balazevic et al., 2019) and Distmult (Yang et al., 2015), respectively. RETIA cannot be trained before 120 hours timeout on ICEWS21-zero. Complete results with Hits@3 are presented in App. F.

they reach the best performance over seen relations. See Table 2 and App. B for dataset statistics.

4.2 Experimental Setup

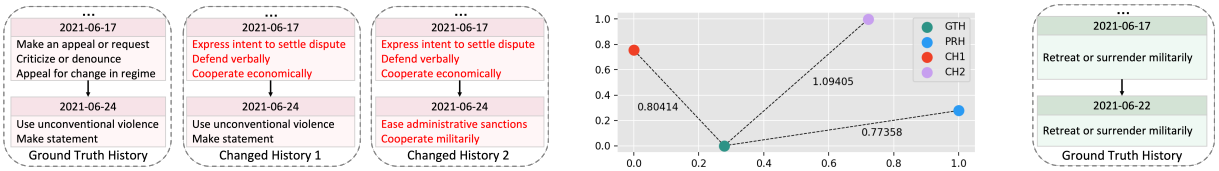
Training and Evaluation for Zero-Shot TKGF. All TKGF models are trained on $\mathcal{G}_{\text{train}}$. We take the model checkpoint achieving the best validation result on $\mathcal{G}_{\text{valid}}$ as the best model checkpoint, and report their test result on $\mathcal{G}_{\text{test}}$ to study the zero-shot inference ability. To keep zero-shot relations "always unseen" during the whole test process, we constrain all models to do LP only based on the training set as several popular TKGF methods, e.g., RE-GCN (Zhu et al., 2021). Some TKGF models, e.g., TiRGN (Li et al., 2022), allow using the ground truth TKG data until the LP query timestamp, including the facts in evaluation sets. This will violate the zero-shot setting because every unseen relation will occur multiple times in the evaluation data and is no longer zero-shot after models observe any fact of it. We prevent them from observing evaluation data to maintain the zero-shot setting. See App. C.5 for explanation. Note that $\mathcal{G}_{\text{valid}}$ and $\mathcal{G}_{\text{test}}$ share the same time period. This is because **we want to make sure that zrLLM can enhance zero-shot reasoning and simultaneously maintain TKGF models' performance on the facts with seen relations. Improving zero-shot inference ability at the cost of sacrificing too much performance over seen relations is undesired.**

Baselines and Evaluation Metrics. We consider seven recent embedding-based TKGF methods as baselines, i.e., CyGNet (Zhu et al., 2021), TANGO-TuckER/Distmult (Han et al., 2021b), RE-GCN (Li et al., 2021b), TiRGN (Li et al., 2022), CENET (Xu et al., 2023b) and RETIA (Liu et al., 2023). We couple them with zrLLM and show their improvement in zero-shot relational learning on TKGs (implementation details in App. C). We employ two evaluation metrics, i.e., mean reciprocal rank (MRR) and Hits@1/3/10. See App. E for detailed definitions. As suggested in (Gastinger et al., 2023), we use the time-aware filtering setting (Han et al., 2021a) for fairer evaluation.

4.3 Comparative Study and Further Analysis

Comparative Study. We report the LP results of all baselines and their zrLLM-enhanced versions in Table 3. We have two findings: (1) zrLLM greatly helps TKGF models in forecasting the facts with unseen zero-shot relations. (2) In most cases, zrLLM even improves models in predicting the facts with seen relations. The zrLLM-enhanced models whose performance drops over seen relations still achieve better overall performance. These findings prove that embedding-based TKGF models benefit from the semantic information extracted from LLMs, especially when they are dealing with zero-shot relations.

Ablation Study. We conduct ablation studies from three aspects. (1) First, we directly input the dataset-provided relation texts into T5-11B encoder, ignoring the relation explanations generated by GPT-3.5. From Table 4 (-ERD), we observe that in most cases, models' performance drops on the facts with both seen and zero-shot relations, which proves the usefulness of ERDs. (2) Next, we remove the RHL from all zrLLM-enhanced models. From Table 4 (-RHL), we find that all the considered TKGF models can benefit from RHL, especially CENET. (3) We switch T5-11B to T5-



(a) (b) (c)

Figure 3: (a) Ground truth and changed relation histories between *United States* and *African Union*. Changed relations are marked in red. Only the histories nearest to 2021-07-03 are shown. (b) t-SNE of encoded GTH, CH1, CH2 (computed with Eq. 4), and predicted history PRH. Numbers beside dashed lines denote point distances (L2 norm). (c) Ground truth relation histories between *United States* and *Afghanistan*.

3B to see the impact of LM size on zrLLM. We observe from Table 4 that decreasing the size of T5 harms model performance. This proves that using larger scale LMs can provide semantic information of higher quality, and can be more beneficial to downstream TKGF (whether zero-shot or not).

Datasets	**ACLED-zero** MRR			**ICEWS21-zero** MRR			**ICEWS22-zero** MRR		
Model	Zero	Seen	Overall	Zero	Seen	Overall	Zero	Seen	Overall
CyGNet+	**0.533**	0.751	**0.723**	**0.201**	**0.258**	**0.257**	**0.286**	**0.315**	**0.314**
- ERD	0.502	0.748	0.716	0.198	0.252	0.251	0.250	0.314	0.311
- RHL	0.503	**0.752**	0.720	0.199	0.256	0.255	0.268	0.297	0.296
T5-3B	0.511	**0.752**	0.721	0.117	0.204	0.202	0.257	**0.315**	0.313
TANGO-T+	0.525	**0.775**	**0.743**	**0.216**	**0.280**	**0.279**	**0.326**	**0.363**	**0.362**
- ERD	0.533	0.772	0.741	0.214	**0.280**	**0.279**	0.320	0.362	0.360
- RHL	0.506	0.755	0.740	0.213	0.277	0.276	0.309	**0.363**	0.361
T5-3B	**0.544**	0.771	0.742	0.206	0.274	0.273	0.323	0.359	0.358
TANGO-D+	**0.491**	**0.760**	**0.725**	**0.212**	**0.268**	**0.267**	**0.311**	**0.350**	**0.348**
- ERD	**0.491**	0.702	0.675	0.205	0.267	0.266	0.285	0.328	0.326
- RHL	0.490	0.725	0.695	0.197	0.224	0.224	0.296	0.324	0.323
T5-3B	0.490	0.701	0.674	0.204	0.223	0.222	0.308	0.284	0.285
RE-GCN+	**0.529**	**0.731**	**0.705**	**0.214**	**0.280**	**0.279**	**0.324**	**0.357**	**0.356**
- ERD	0.489	0.730	0.699	0.211	0.277	0.276	0.294	0.354	0.352
- RHL	0.519	0.726	0.699	0.213	0.277	0.276	0.317	0.350	0.349
T5-3B	0.504	0.721	0.693	0.211	0.259	0.258	0.301	0.354	0.352
TiRGN+	**0.548**	**0.754**	**0.727**	**0.221**	**0.279**	**0.278**	**0.333**	**0.353**	**0.352**
- ERD	0.480	0.747	0.713	0.211	0.275	0.274	0.282	**0.353**	0.350
- RHL	0.515	0.752	0.721	0.215	0.277	0.276	0.320	0.350	0.349
T5-3B	0.498	0.749	0.717	0.208	0.271	0.270	0.325	0.345	0.344
RETIA+	**0.557**	**0.783**	**0.754**	» 120 Hours Timeout			**0.331**	**0.358**	**0.357**
- ERD	0.519	0.777	0.744				0.292	0.354	0.352
- RHL	0.529	0.782	0.749				0.318	0.357	0.355
T5-3B	0.512	0.776	0.742				0.330	0.353	0.352
CENET+	**0.591**	**0.779**	**0.755**	**0.335**	**0.396**	**0.395**	**0.564**	**0.571**	**0.570**
- ERD	0.526	0.737	0.710	0.321	0.374	0.373	0.542	0.570	0.568
- RHL	0.445	0.754	0.714	0.232	0.290	0.289	0.295	0.370	0.367
T5-3B	0.568	0.736	0.714	0.303	0.330	0.329	0.550	0.555	0.554

Table 4: Ablation study (complete results in App. G).

Compare with Previous LM-Enhanced Model. We benchmark two recent LM-enhanced TKGF models PPT (Xu et al., 2023a) and ICL + GPT-NeoX-20B (Lee et al., 2023; Black et al., 2022) (Table 5). PPT finetunes BERT for TKGF. We find that although PPT achieves strong zero-shot results, it is beaten by several zrLLM-enhanced models. This proves that aligning language space to TKGF is helpful for zero-shot relational learning and LMs with larger size can be more contributive. ICL shows inferior results. This proves that without finetuning or alignment, LLMs are unable to optimally solve TKGF. zrLLM not only benefits from a large LM but also enables efficient alignment from language to TKG embedding space, which leads to superior performance.

Datasets	**ACLED-zero** MRR			**ICEWS21-zero** MRR			**ICEWS22-zero** MRR		
Model	Zero	Seen	Overall	Zero	Seen	Overall	Zero	Seen	Overall
PPT	0.532	0.782	0.748	0.212	0.269	0.268	0.323	0.332	0.331
ICL	0.537	0.736	0.709	0.156	0.178	0.177	0.255	0.229	0.230

Table 5: PPT and ICL performance. Implementation details and complete results in App. C.3 and H.

Case Study of RHL We do a case study to show: (1) RHL's HPN is able to capture ground truth relation history (GTH). (2) By capturing temporal relation patterns, RHL helps for better zero-shot TKGF. We ask zrLLM-enhanced CENET to predict the missing object of the test query $q = (s_q, r_q, ?, t_q) = (\textit{United States}, \textit{Reduce or stop military assistance}, ?, \text{2021-07-03})$ (answer is $o_q = \textit{African Union}$) taken from ICEWS21-zero. The GTH of s_q and o_q (Fig. 3a, left) shows a pattern indicating their recent worsening relationship. It can serve as a clue in LP over q because it can be viewed as a "cause" to the query relation r_q which also implies a negative relationship. In other words, the entities with a worsening historical relationship are more likely to be connected with a relation showing their bad relationship currently. Since RHL uses HPN to infer GTH during test, we wish to study whether HPN can achieve reasonable inference to support LP. Based on GTH, we first change all three relations on 2021-06-17 to randomly sampled positive relations seen in the training data and form a changed history 1 (CH1, Fig. 3a, middle). Then we further modify the relations on 2021-06-24 in the same way and form a changed history 2 (CH2, Fig. 3a, right). We use Eq. 4 to encode GTH, CH1, CH2, and visualize them together with the predicted history (PRH) computed with HPN

by using t-SNE (van der Maaten and Hinton, 2008) in Fig. 3b. We find that PRH is the closest to GTH and CH1 is closer than CH2 to GTH. The reason why CH2 is much farther from GTH is that CH2 changes more negative relations to positive, greatly changing the semantic meaning stored in GTH. CH1 only introduces changes on 2021-06-17, making it less deviated from GTH. HPN takes the r_q and can keep PRH close to GTH, making zrLLM able to maximally capture the temporal patterns indicated by GTH, while preventing the scalability problem incurred by searching relation histories of all candidate entities. By using RHL, the zrLLM-enhanced CENET can correctly predict o_q, while the model without RHL takes $o' = \textit{Afghanistan}$ as the predicted answer. We present the nearest GTH between s_q and o' in Fig. 3c and find that it indicates a positive relationship which is unlikely to cause r_q right after. During training, RHL learns patterns and matches entity pairs with them (Eq. 8). This enables RHL to exclude the entities that do not fit into the learned patterns from the answer set and make more accurate predictions.

5 Conclusion

We study zero-shot relational learning in TKGF and design an LLM-empowered approach, i.e., zrLLM. zrLLM extracts the semantic information of KG relations from LLMs and introduces it into TKG representation learning. It also uses an RHL module to capture the temporal relation patterns for better reasoning, and meanwhile promote the embedding space alignment between text and TKGs. We couple zrLLM with several embedding-based TKGF models and find that zrLLM provides huge help in forecasting the facts with zero-shot relations, and moreover, it maintains models' performance over seen relations.

6 Limitations

Our limitations can be summarized as follows. First, zrLLM is developed only for enhancing embedding-based TKG forecasting methods. It is not directly applicable to the rule-based methods, e.g., TLogic. Besides, relation history learner inevitably increases model's training and evaluation time since relation patterns are learned with GRUs where recurrent computations are performed along the time axis. More GPU memory is also required for storing relation histories. This hinders the efficiency of zrLLM-enhanced models compared with the original baselines. In the future, we will explore how to generalize our proposed method to rule-based models and try to improve model efficiency. We will also try to experiment zrLLM on more TKG forecasting methods and study whether we can benefit more of them.

Acknowledgments

This work has been partially funded by the Munich Center for Machine Learning and supported by the Federal Ministry of Education and Research and the State of Bavaria. This work has also been supported by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) as part of the project CoyPu under grant number 01MK21007K. The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Bo Xiong. Bo Xiong has also been partially funded by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy - EXC 2075 - 390740016, the Stuttgart Center for Simulation Science (SimTech), and the Bundesministerium für Wirtschaft und Energie (BMWi), grant aggrement No. 01MK20008F.

References

Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. 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 5184–5193. Association for Computational Linguistics.

Sid Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, Michael Pieler, USVSN Sai Prashanth, Shivanshu Purohit, Laria Reynolds, Jonathan Tow, Ben Wang, and Samuel Weinbach. 2022. Gpt-neox-20b: An open-source autoregressive language model. *CoRR*, abs/2204.06745.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. ICEWS Coded Event Data.

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*.

Zhongwu Chen, Chengjin Xu, Fenglong Su, Zhen Huang, and Yong Dou. 2023a. Incorporating structured sentences with time-enhanced BERT for fully-inductive temporal relation prediction. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023*, pages 889–899. ACM.

Zhongwu Chen, Chengjin Xu, Fenglong Su, Zhen Huang, and Yong Dou. 2023b. Meta-learning based knowledge extrapolation for temporal knowledge graph. In *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 2433–2443. ACM.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL.

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, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Zifeng Ding, Bailan He, Jingpei Wu, Yunpu Ma, Zhen Han, and Volker Tresp. 2023a. Learning meta-representations of one-shot relations for temporal knowledge graph link prediction. In *International Joint Conference on Neural Networks, IJCNN 2023, Gold Coast, Australia, June 18-23, 2023*, pages 1–10. IEEE.

Zifeng Ding, Zongyue Li, Ruoxia Qi, Jingpei Wu, Bailan He, Yunpu Ma, Zhao Meng, Shuo Chen, Ruotong Liao, Zhen Han, and Volker Tresp. 2023b. Forecasttkgquestions: A benchmark for temporal question answering and forecasting over temporal knowledge graphs. In *ISWC*, volume 14265 of *Lecture Notes in Computer Science*, pages 541–560. Springer.

Zifeng Ding, Jingpei Wu, Bailan He, Yunpu Ma, Zhen Han, and Volker Tresp. 2022. Few-shot inductive learning on temporal knowledge graphs using concept-aware information. In *4th Conference on Automated Knowledge Base Construction*.

Zifeng Ding, Jingpei Wu, Zongyue Li, Yunpu Ma, and Volker Tresp. 2023c. Improving few-shot inductive learning on temporal knowledge graphs using confidence-augmented reinforcement learning. In *Machine Learning and Knowledge Discovery in Databases: Research Track - European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part III*, volume 14171 of *Lecture Notes in Computer Science*, pages 550–566. Springer.

Julia Gastinger, Timo Sztyler, Lokesh Sharma, Anett Schuelke, and Heiner Stuckenschmidt. 2023. Comparing apples and oranges? on the evaluation of methods for temporal knowledge graph forecasting. In *Machine Learning and Knowledge Discovery in Databases: Research Track - European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part III*, volume 14171 of *Lecture Notes in Computer Science*, pages 533–549. Springer.

Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. 2021a. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Zhen Han, Zifeng Ding, Yunpu Ma, Yujia Gu, and Volker Tresp. 2021b. Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8352–8364. Association for Computational Linguistics.

Zhen Han, Ruotong Liao, Jindong Gu, Yao Zhang, Zifeng Ding, Yujia Gu, Heinz Koeppl, Hinrich Schütze, and Volker Tresp. 2023. ECOLA: Enhancing temporal knowledge embeddings with contextualized language representations. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 5433–5447, Toronto, Canada. Association for Computational Linguistics.

Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. 2020. Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6669–6683. Association for Computational Linguistics.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Julien Leblay and Melisachew Wudage Chekol. 2018. Deriving validity time in knowledge graph. In *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1771–1776. ACM.

Dong-Ho Lee, Kian Ahrabian, Woojeong Jin, Fred Morstatter, and Jay Pujara. 2023. Temporal knowledge graph forecasting without knowledge using in-context learning. *CoRR*, abs/2305.10613.

Yujia Li, Shiliang Sun, and Jing Zhao. 2022. Tirgn: Time-guided recurrent graph network with local-global historical patterns for temporal knowledge graph reasoning. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 2152–2158. ijcai.org.

Zixuan Li, Xiaolong Jin, Saiping Guan, Wei Li, Jiafeng Guo, Yuanzhuo Wang, and Xueqi Cheng. 2021a. Search from history and reason for future: Two-stage reasoning on temporal knowledge graphs. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4732–4743. Association for Computational Linguistics.

Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. 2021b. Temporal knowledge graph reasoning based on evolutional representation learning. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 408–417. ACM.

Ruotong Liao, Xu Jia, Yunpu Ma, and Volker Tresp. 2023. GenTKG: Generative forecasting on temporal knowledge graph. In *Temporal Graph Learning Workshop @ NeurIPS 2023*.

Kangzheng Liu, Feng Zhao, Guandong Xu, Xianzhi Wang, and Hai Jin. 2023. RETIA: relation-entity twin-interact aggregation for temporal knowledge graph extrapolation. In *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, CA, USA, April 3-7, 2023*, pages 1761–1774. IEEE.

Yushan Liu, Yunpu Ma, Marcel Hildebrandt, Mitchell Joblin, and Volker Tresp. 2022. Tlogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 4120–4127. AAAI Press.

Ruixin Ma, Biao Mei, Yunlong Ma, Hongyan Zhang, Meihong Liu, and Liang Zhao. 2023. One-shot relational learning for extrapolation reasoning on temporal knowledge graphs. *Data Min. Knowl. Discov.*, 37(4):1591–1608.

Xin Mei, Libin Yang, Xiaoyan Cai, and Zuowei Jiang. 2022. An adaptive logical rule embedding model for inductive reasoning over temporal knowledge graphs. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 7304–7316. Association for Computational Linguistics.

Mehrnoosh Mirtaheri, Mohammad Rostami, Xiang Ren, Fred Morstatter, and Aram Galstyan. 2021. One-shot learning for temporal knowledge graphs. In *3rd Conference on Automated Knowledge Base Construction, AKBC 2021, Virtual, October 4-8, 2021*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035.

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.

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.

Clionadh Raleigh, rew Linke, Håvard Hegre, and Joakim Karlsen. 2010. Introducing acled: An armed conflict location and event dataset. *Journal of Peace Research*, 47(5):651–660.

Apoorv Saxena, Soumen Chakrabarti, and Partha P. Talukdar. 2021. Question answering over temporal knowledge graphs. In *ACL/IJCNLP (1)*, pages 6663–6676. Association for Computational Linguistics.

Haohai Sun, Jialun Zhong, Yunpu Ma, Zhen Han, and Kun He. 2021. TimeTraveler: Reinforcement learning for temporal knowledge graph forecasting. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages

8306–8319, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971.

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605.

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 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Ruijie Wang, Zheng Li, Dachun Sun, Shengzhong Liu, Jinning Li, Bing Yin, and Tarek F. Abdelzaher. 2022. Learning to sample and aggregate: Few-shot reasoning over temporal knowledge graphs. In *NeurIPS*.

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. 2022. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022.

Wenjie Xu, Ben Liu, Miao Peng, Xu Jia, and Min Peng. 2023a. Pre-trained language model with prompts for temporal knowledge graph completion. In *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 7790–7803. Association for Computational Linguistics.

Yi Xu, Junjie Ou, Hui Xu, and Luoyi Fu. 2023b. Temporal knowledge graph reasoning with historical contrastive learning. In *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 4765–4773. AAAI Press.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. 2019. Quaternion knowledge graph embeddings. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 2731–2741.

Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhang. 2021. Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 4732–4740. AAAI Press.

A Detailed Illustration of Prompt for GPT-3.5

We give a detailed illustration of our prompt for producing ERDs with GPT-3.5 in Fig. 4. For every batch of n relations, we incorporate their dataset-provided texts into our prompt to generate their enriched descriptions.

B Further Details of Zero-Shot Datasets

For each dataset, we provide the distribution of all zero-shot relations' frequncies in Fig. 5. We take the relations with lowest frequencies as zero-shot relations when we construct datasets, following previous few-shot relational TKG learning frameworks, e.g., OAT (Mirtaheri et al., 2021) and MOST (Ding et al., 2023a). The proportion of zero-shot relations for each dataset is high. 14 out of 23; 123 out of 253; 155 out of 248 relations in ACLED-zero; ICEWS21-zero; ICEWS22-zero are zero-shot relations. This ensures the diversity of relation types in test sets.

C Implementation Details

All experiments are implemented with PyTorch (Paszke et al., 2019) on a server equipped with an AMD EPYC 7513 32-Core Processor and a single NVIDIA A40 with 48GB memory. All the experimental results are the average of three runs with different random seeds.

C.1 Baseline Implementation Details

Our baselines are all based on neural networks rather than pure score function-based (e.g., TTransE (Leblay and Chekol, 2018)). This is because the most popular and recent TKGF methods all leverage neural networks to gain the forecasting ability and it is hard for pure score function-based methods to achieve that solely with geometric embeddings. The implementation details of each TKGF baseline is as follows.

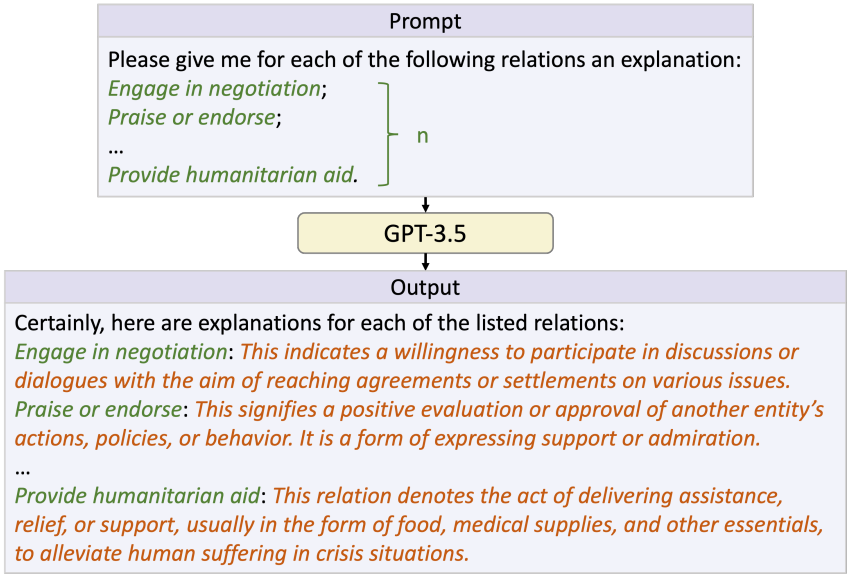
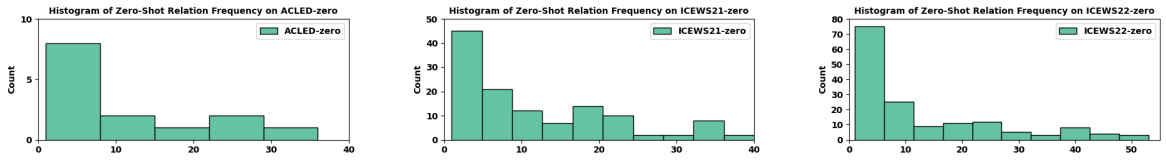


Figure 4: Prompting GPT-3.5 for ERDs. The green texts are the short relation texts provided in the original datasets. The orange texts are the generated relation explanations from GPT-3.5.



(a) ACLED-zero. (b) ICEWS21-zero. (c) ICEWS22-zero.

Figure 5: Zero-shot Relation frequency on all zero-shot TKGF datasets. Horizontal axis denotes the appearance times, i.e., frequency. Vertical axis denotes the number of relations.

- **CyGNet.** We use the official code of CyGNet[4]. We search hyperparameters of baseline CyGNet following Table 6. The best hyperparameters are marked as bold. For each dataset, we do 4 trials to try different hyperparameter settings. We run 5 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

Dataset	**ACLED-zero**	**ICEWS21-zero**	**ICEWS22-zero**
Hyperparameter	CyGNet	CyGNet	CyGNet
Embedding Size	{100, **200**}	{100, **200**}	{100, **200**}
Alpha (Eq. 9 in (Zhu et al., 2021))	{**0.2**, 0.5}	{**0.2**, 0.5}	{**0.2**, 0.5}

Table 6: CyGNet hyperparameter searching strategy.

- **TANGO-TuckER/Distmult.** We use the official code of TANGO[5]. We search hyperparameters of baseline TANGO-TuckER/Distmult following Table 7. The best hyperparameters are marked as bold. For each dataset, we do 6 (TANGO-TuckER) and 9 (TANGO-Distmult) trials to try different hyperparameter settings. We run 10 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

Dataset	**ACLED-zero**		**ICEWS21-zero**		**ICEWS22-zero**	
Hyperparameter	TuckER	Distmult	TuckER	Distmult	TuckER	Distmult
Embedding Size	{100, **200**}	{100, 200, **300**}	{**100**, 200}	{**100**, 200, 300}	{100, **200**}	{100, **200**, 300}
History Length	{**4**, 6, 10}	{**4**, 6, 10}	{**4**, 6, 10}	{**4**, 6, 10}	{**4**, 6, 10}	{**4**, 6, 10}

Table 7: TANGO hyperparameter searching strategy.

- **RE-GCN.** We use the official code of RE-GCN[6]. We search hyperparameters of baseline RE-GCN following Table 8. The best hyperparameters are marked as bold. For each dataset, we do 4 trials to try different hyperparameter settings. We run 10 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.
- **TiRGN.** We use the official code of TiRGN[7]. We search hyperparameters of baseline

[4]https://github.com/CunchaoZ/CyGNet
[5]https://github.com/TemporalKGTeam/TANGO
[6]https://github.com/Lee-zix/RE-GCN
[7]https://github.com/Liyyy2122/TiRGN

Dataset Hyperparameter	ACLED-zero RE-GCN	ICEWS21-zero RE-GCN	ICEWS22-zero RE-GCN
Embedding Size	{100, **200**}	{**100**, 200}	{100, **200**}
History Length	{**3**, 9}	{3, **9**}	{3, **9**}

Table 8: RE-GCN hyperparameter searching strategy.

TiRGN following Table 9. The best hyperparameters are marked as bold. For each dataset, we do 12 trials to try different hyperparameter settings. We run 10 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

Dataset Hyperparameter	ACLED-zero TiRGN	ICEWS21-zero TiRGN	ICEWS22-zero TiRGN
Embedding Size	{100, **200**}	{**100**, 200}	{100, **200**}
History Length	{**3**, 9}	{3, **9**}	{3, **9**}
Alpha (Eq. 11 in (Li et al., 2022))	{**0.3**, 0.5, 0.7}	{**0.3**, 0.5, 0.7}	{**0.3**, 0.5, 0.7}

Table 9: TiRGN hyperparameter searching strategy.

- **RETIA.** We use the official code of RETIA[8]. We search hyperparameters of baseline RETIA following Table 10. The best hyperparameters are marked as bold. For each dataset, we do 4 trials to try different hyperparameter settings. We run 10 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

Dataset Hyperparameter	ACLED-zero RETIA	ICEWS21-zero RETIA	ICEWS22-zero RETIA
Embedding Size	{100, **200**}	{**100**, 200}	{100, **200**}
History Length	{**3**, 9}	{3, **9**}	{3, **9**}

Table 10: RETIA hyperparameter searching strategy.

- **CENET.** We use the official code of CENET[9]. We search hyperparameters of baseline CENET following Table 11. The best hyperparameters are marked as bold. For each dataset, we do 4 trials to try different hyperparameter settings. We run 5 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

The hyperparameters not discussed above follow the settings reported in the original papers.

C.2 zrLLM Implementation Details

We fix the hyperparameters searched from the baselines and additionally search zrLLM-specific hyperparameters for zrLLM-enhanced models. The hyperparameter searching strategy and the best hyperparameter settings regarding the zrLLM-enhanced

Dataset Hyperparameter	ACLED-zero CENET	ICEWS21-zero CENET	ICEWS22-zero CENET
Embedding Size	{**100**, 200}	{100, **200**}	{100, **200**}
Mask Strategy	{**soft**, hard}	{**soft**, hard}	{**soft**, hard}

Table 11: CENET hyperparameter searching strategy.

baselines are reported in Table 12. Note that γ can be either a learnable parameter or a fixed scalar. When γ is not fixed, γ Value means the initialized parameter value during training. For each zrLLM-enhanced model, in each dataset, we do 24 trials to try different hyperparameter settings. We run 7 epochs for each trail and take the one with the best validation result as the best hyperparameter setting.

C.3 Implementation Details of PPT and ICL

We use the official code of PPT[10] and ICL[11]. For PPT, we use the default hyperparameter setting used for ICEWS14 when we implement it on all our new datasets. Since PPT only explores object entity prediction in its original implementation, we add the subject entity prediction part and report the overall result. We achieve subject prediction by first deriving the inverse relation texts for each relation in each TKG dataset, e.g., use *Inversed Reduce or stop military assistance* to represent the inverse relation of the relation *Reduce or stop military assistance*, and then turning each subject prediction query $(?, r_q, o_q, t_q)$ to an object prediction query $(o_q, r_q^{-1}, ?, t_q)$, where r_q^{-1} stands for the inverse relation of r_q. For ICL, we use the lexical-based prompt because we are dealing with zero-shot relations where text information is important. We also employ the unidirectional entity-focused history, which achieves best results on ICEWS14 as reported in ICL's original paper. We use the default history length of 20 for all datasets.

C.4 Computational Resource Usage

We report the computational resources for all zrLLM-enhanced models and PPT in Table 13. Training time denotes the period of time a model requires to reach its best validation performance. PPT requires extremely long time for sampling and thus has high time consumption. Note that zrLLM loads T5 to generate LM-based relation representations. This process takes a substantial amount of GPU memory. However, in our work, we store the output of T5's encoder as saved parameters and use them in downstream zero-shot TKGF with any

[8] https://github.com/CGCL-codes/RETIA
[9] https://github.com/xyjigsaw/CENET
[10] https://github.com/JaySaligia/PPT
[11] https://github.com/usc-isi-i2/isi-tkg-icl

Dataset	ACLED-zero				ICEWS21-zero				ICEWS22-zero			
Model	α	γ Type	γ Value	η	α	γ Type	γ Value	η	α	γ Type	γ Value	η
CyGNet+	{1, **0.1**}	{**Fixed**, Unfixed}	{**1**, 0.01, 0.001}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, 0.01, **0.001**}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, 0.01, **0.001**}	{1.2, **1**}
TANGO-T+	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{**1.2**, 1}	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{**1.2**, 1}	{1, **0.1**}	{**Fixed**, Unfixed}	{**1**, 0.01, 0.001}	{**1.2**, 1}
TANGO-D+	{1, **0.1**}	{**Fixed**, Unfixed}	{**1**, 0.01, 0.001}	{**1.2**, 1}	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{**1.2**, 1}	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{**1.2**, 1}
RE-GCN+	{1, **0.1**}	{**Fixed**, Unfixed}	{1, 0.01, **0.001**}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, **0.01**, 0.001}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, **0.01**, 0.001}	{1.2, **1**}
TiRGN+	{1, **0.1**}	{Fixed, **Unfixed**}	{1, 0.01, **0.001**}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, **0.01**, 0.001}	{1.2, **1**}	{1, **0.1**}	{**Fixed**, Unfixed}	{1, **0.01**, 0.001}	{1.2, **1**}
RETIA+	{1, **0.1**}	{Fixed, **Unfixed**}	{1, **0.01**, 0.001}	{**2**, 1}	-	-	-	-	{1, **0.1**}	{**Fixed**, Unfixed}	{1, **0.01**, 0.001}	{**2**, 1}
CENET+	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{1.2, **1**}	{1, **0.1**}	{Fixed, **Unfixed**}	{**1**, 0.01, 0.001}	{1.2, **1**}	{1, **0.1**}	{Fixed, **Unfixed**}	{1, **0.01**, 0.001}	{1.2, **1**}

Table 12: zrLLM hyperparameter searching strategy. The best settings are marked as bold.

Dataset	ACLED-zero		ICEWS21-zero		ICEWS22-zero	
Model	Training Time (h)	GPU Memory (MB)	Training Time (h)	GPU Memory (MB)	Training Time (h)	GPU Memory (MB)
CyGNet+	0.03	2,216	17.87	7,470	4.80	9,574
TANGO-T+	0.05	2,716	8.64	34,186	2.82	20,120
TANGO-D+	0.11	3,064	10.88	34,034	0.70	19,250
RE-GCN+	0.06	1,587	14.70	26,420	3.85	19,168
TiRGN+	0.10	2,654	11.67	36,780	2.40	15,976
RETIA+	0.13	4,274	-	-	9.33	26,328
CENET+	0.03	1,429	48.94	6,750	12.54	5,639
PPT	0.47	7,654	84.68	9,078	59.35	7,678

Table 13: Computational resources required by zrLLM-enhanced models and PPT.

zrLLM-enhanced model. This prevents from high memory demand during model training and evaluation. We use Fig. 6 to illustrate the direct comparison among zrLLM-enhanced models and PPT regarding their required computational resources during training.

ICL loads GPT-NeoX-20B that requires huge memory consumption. We use two NVIDIA A40 for all its experiments. Since ICL does not require training, we only report its validation and test time here. For ACLED-zero, GPU memory usage is 90,846 MB. Validation time is 0.63 h and test time is 0.12 h. For ICEWS21-zero, GPU memory usage is 90,868 MB. Validation time is 35.48 h and test time is 0.82 h. For ICEWS22-zero, GPU memory usage is 91,458 MB. Validation time is 22.98 h and test time is 1.15 h.

C.5 Zero-Shot Evaluation Setting Explanation

To keep zero-shot relations "always unseen" during the whole evaluation process, we constrain all models to do LP only based on the training set. Among all TKGF models, TANGO, RE-GCN, TiRGN and RETIA use recurrent neural structures to model historical TKG information from a short sequence of timestamps prior to the prediction timestamp. We constrain them to only use the latest training data, i.e., from $t_{\text{train\_max}} - k$ to $t_{\text{train\_max}}$, to encode historical information during evaluation. k is the considered history length and $t_{\text{train\_max}} = \max(\mathcal{T}_{\text{train}})$ is the maximum timestamp in the training data. For CyGNet and CENET, they have originally met our restriction of not observing any ground truth evaluation data during evaluation, and thus can be directly implemented in our zero-shot setting. Another point worth noting is that RHL requires ground truth relation history. We restrict zrLLM to only capture the relation history across the whole training time period to prevent from exposing zero-shot relations during evaluation.

D Algorithm

We provide algorithms to show the whole process of using zrLLM to enhance TKGF models. First, zrLLM generates LLM-based relation representations by using GPT-3.5 and T5-11B (Algorithm 1). Then we train zrLLM jointly with TKGF baseline models (Algorithm 2). The trained models are then used for evaluation (Algorithm 3).

Algorithm 1: Generate LLM-based Relation Representations

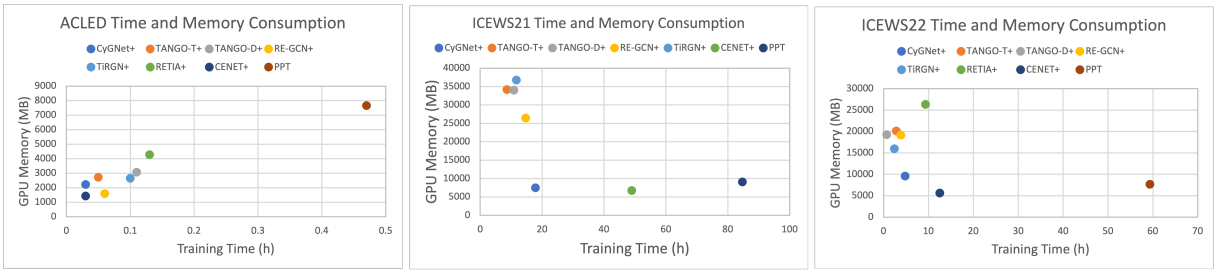
Input: Relations $\mathcal{R}$, relation text of all relations provided by the TKG dataset $\text{TEXT}_{\mathcal{R}}$

for batch = *1: B* **do**
- Take a batch of n relations from $\mathcal{R}$
- Pick out their relation texts from $\text{TEXT}_{\mathcal{R}}$
- Write prompt with the relation texts `// Fig. 2`
- Input the prompt into GPT-3.5
- Extract the ERDs from the output of GPT-3.5
- Input the ERDs into T5-11B's encoder
- Store the output of T5-11B's encoder

return T5-encoded text representation $\bar{\mathbf{H}}_r$ for every $r \in \mathcal{R}$

E Evaluation Metrics Details

We employ two evaluation metrics, i.e., mean reciprocal rank (MRR) and Hits@1/3/10. For every LP query q, we compute the rank θ_q of the ground truth missing entity. We define MRR as: $\frac{1}{|\mathcal{G}_{\text{test}}|} \sum_q \frac{1}{\theta_q}$



(a) ACLED-zero. (b) ICEWS21-zero. (c) ICEWS22-zero.

Figure 6: Computational resources required during training of zrLLM-enhanced models and PPT.

Algorithm 2: Model Training with zrLLM

Input: Entities $\mathcal{E}$, relations $\mathcal{R}$, timestamps $\mathcal{T}$, T5-encoded text representations $\{\bar{\mathbf{H}}_r\}$ for $\mathcal{R}$, training set $\mathcal{G}_{\text{train}}$

Align $\{\bar{\mathbf{H}}_r\}$ to TKG embedding space and get $\{\bar{\mathbf{h}}_r\}$ // Eq. 1, 2
for epoch = *1: V* **do**
  for batch = *1: B* **do**
    Take a batch of training facts $\{(s, r, o, t)\} \in \mathcal{G}_{\text{train}}$
    Find the relation history of s and o before t for each (s, r, o, t)
    Encode relation history until $t-1$ // Eq. 4
    Compute the predicted history with HPN // Eq. 5
    Compute history-related MSE loss $\mathcal{L}_{\text{hist}}$ // Eq. 6
    Compute the representation of the r-related temporal relation pattern // Eq. 7
    Compute the RHL-based score // Eq. 8
    Input $\{\bar{\mathbf{h}}_r\}$ into TKGF baseline and compute LP score
    Compute total score for the training batch // Eq. 9
    Compute TKGF model loss $\mathcal{L}_{\text{TKGF}}$ // Eq. 10
    Compute RHL-based loss $\mathcal{L}_{\text{RHL}}$ // Eq. 11
    Compute total loss $\mathcal{L}_{\text{total}}$ // Eq. 12
    Update model parameters using gradient of $\bigtriangledown\mathcal{L}_{\text{total}}$
return trained zrLLM-enhanced TKGF model

Algorithm 3: Model Evaluation with zrLLM

Input: Entities $\mathcal{E}$, relations $\mathcal{R}$, timestamps $\mathcal{T}$, LLM-based relation representations $\{\bar{\mathbf{h}}_r\}$ for $\mathcal{R}$, training set $\mathcal{G}_{\text{train}}$, validation set $\mathcal{G}_{\text{valid}}$, test set $\mathcal{G}_{\text{test}}$

if *evaluation set is* $\mathcal{G}_{valid}$ **then**
  $\mathcal{G}_{\text{eval}} = \mathcal{G}_{\text{valid}}$
else
  $\mathcal{G}_{\text{eval}} = \mathcal{G}_{\text{test}}$
for batch = *1: B* **do**
  Take a batch of evaluation facts $\{(s_q, r_q, o_q, t_q)\} \in \mathcal{G}_{\text{eval}}$
  Derive LP queries $\{(s_q, r_q, ?, t_q)\}$
  Input $\{r_q\}$ into HPN and compute the predicted history // Eq. 5
  Compute the representation of the r_q-related temporal relation pattern for each LP query // Eq. 7
  Compute the RHL-based score of each candidate entity $e \in \mathcal{E}$ for each LP query // Eq. 8
  Input $\{\bar{\mathbf{h}}_r\}$ into TKGF baseline and compute LP score of each candidate entity $e \in \mathcal{E}$ for each LP query
  Compute total score of each candidate entity $e \in \mathcal{E}$ for each LP query in the batch // Eq. 9
  Rank candidate entities $\mathcal{E}$ with their total scores in the descending order
  Compute and record the rank of the ground truth missing entity o_q for each LP query
Compute MRR and Hits@1/3/10
return MRR and Hits@1/3/10

(the definition is similar for $\mathcal{G}_{\text{valid}}$). Hits@1/3/10 denote the proportions of the predicted links where ground truth missing entities are ranked as top 1, top3, top10, respectively. As explored and suggested in (Gastinger et al., 2023), we also use the time-aware filtering setting proposed in (Han et al., 2021a) for fairer evaluation.

F Complete Comparative Study Results

We report the complete results of comparative study in Table 14 and 15.

G Complete Ablation Study Results

We report the complete ablation study results in Table 16.

H Complete Results of Previous LM-Enhanced TKGF Model

We report the complete results of previous LM-enhanced TKGF models in Table 14 and 15.

I Further Discussion about RHL

In RHL, temporal relation patterns are captured by only using LLM-based relation representations. Since for all relations (whether zero-shot or not), their LLM-based representations contain semantic information extracted from the same LLM, the learned HPN can do reasonable relation history prediction even with an input of unseen zero-shot relation. If we learn hidden representations for each relation based on graph contexts (as most TKGF models do), zero-shot relations cannot be easily processed by HPN anymore. In this case, zero-shot relations will not have a meaningful representation without any observed associated fact, and therefore, HPN cannot detect its meaning and will fail to find reasonable relation history.

Datasets	ICEWS21-zero									ICEWS22-zero								
	Zero-Shot Relations				Seen Relations				Overall	Zero-Shot Relations				Seen Relations				Overall
Model	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR
CyGNet	0.120	0.046	0.130	0.270	0.254	**0.165**	0.293	0.432	0.252	0.211	0.098	0.240	0.459	**0.315**	0.198	**0.373**	0.540	0.311
CyGNet+	**0.201**	**0.103**	**0.226**	**0.415**	**0.258**	0.162	**0.294**	**0.447**	**0.257**	**0.286**	**0.167**	**0.324**	**0.542**	**0.315**	**0.200**	0.364	**0.545**	**0.314**
TANGO-T	0.067	0.031	0.069	0.132	**0.283**	**0.190**	**0.319**	**0.470**	**0.279**	0.092	0.042	0.100	0.187	**0.363**	0.250	0.407	0.579	0.352
TANGO-T+	**0.216**	**0.125**	**0.245**	**0.395**	0.280	0.186	0.313	0.466	**0.279**	**0.326**	**0.198**	**0.388**	**0.578**	**0.363**	**0.251**	**0.409**	**0.585**	**0.362**
TANGO-D	0.012	0.005	0.011	0.023	0.266	**0.178**	0.298	0.439	0.261	0.011	0.002	0.007	0.018	**0.350**	0.227	**0.394**	0.569	0.337
TANGO-D+	**0.212**	**0.122**	**0.237**	**0.400**	**0.268**	0.175	**0.303**	**0.453**	**0.267**	**0.311**	**0.186**	**0.374**	**0.574**	**0.350**	**0.239**	0.393	**0.570**	**0.348**
RE-GCN	0.200	0.104	0.231	0.379	0.277	0.185	0.309	**0.456**	0.276	0.280	0.162	0.321	**0.616**	0.354	0.243	**0.398**	0.567	0.351
RE-GCN+	**0.214**	**0.117**	**0.246**	**0.406**	**0.280**	**0.188**	**0.314**	**0.456**	**0.279**	**0.324**	**0.194**	**0.376**	0.595	**0.357**	**0.244**	**0.398**	**0.573**	**0.356**
TiRGN	0.189	0.101	0.209	0.368	0.275	0.182	0.308	0.457	0.273	0.299	0.169	0.358	0.570	0.352	0.239	0.399	0.575	0.350
TiRGN+	**0.221**	**0.130**	**0.246**	**0.410**	**0.279**	**0.185**	**0.323**	**0.464**	**0.278**	**0.333**	**0.203**	**0.383**	**0.602**	**0.353**	**0.240**	**0.400**	**0.577**	**0.352**
RETIA	» 120 Hours Timeout									0.302	0.166	0.349	0.566	0.356	0.245	0.401	0.577	0.354
RETIA+										**0.331**	**0.201**	**0.384**	**0.597**	**0.358**	**0.247**	**0.402**	**0.578**	**0.357**
CENET	0.205	0.101	0.232	0.411	0.288	0.196	0.318	0.468	0.287	0.270	0.134	0.318	0.544	0.379	0.268	0.423	0.599	0.375
CENET+	**0.335**	**0.162**	**0.455**	**0.659**	**0.396**	**0.239**	**0.502**	**0.688**	**0.395**	**0.564**	**0.432**	**0.649**	**0.801**	**0.571**	**0.451**	**0.651**	**0.773**	**0.571**
PPT	0.212	0.120	0.240	0.403	0.269	0.172	0.304	0.462	0.268	0.323	0.191	0.376	0.598	0.332	0.219	0.377	0.556	0.331
ICL	0.156	0.096	0.180	0.300	0.178	0.120	0.206	0.308	0.177	0.255	0.162	0.303	0.460	0.229	0.158	0.264	0.393	0.230

Table 14: Complete LP results on ICEWS21-zero and ICEWS22-zero. We also report PPT and ICL's performance.

Datasets	ACLED-zero								
	Zero-Shot Relations				Seen Relations				Overall
Model	MRR	Hits@1	Hits@3	Hits@10	MRR	Hits@1	Hits@3	Hits@10	MRR
CyGNet	0.487	0.349	0.565	**0.791**	**0.751**	0.663	**0.827**	0.903	0.717
CyGNet+	**0.533**	**0.418**	**0.592**	0.753	**0.751**	**0.664**	0.821	**0.906**	**0.723**
TANGO-T	0.052	0.021	0.049	0.101	0.774	0.701	0.826	0.900	0.681
TANGO-T+	**0.525**	**0.393**	**0.606**	**0.746**	**0.775**	**0.702**	**0.827**	**0.901**	**0.743**
TANGO-D	0.021	0.003	0.017	0.049	**0.777**	**0.701**	**0.833**	**0.907**	0.679
TANGO-D+	**0.491**	**0.348**	**0.560**	**0.791**	0.760	0.678	0.818	0.901	**0.725**
RE-GCN	0.441	0.332	0.466	0.718	0.730	**0.653**	0.783	0.865	0.693
RE-GCN+	**0.529**	**0.393**	**0.612**	**0.784**	**0.731**	0.650	**0.789**	**0.876**	**0.705**
TiRGN	0.478	0.330	0.572	0.745	**0.754**	0.678	0.806	**0.886**	0.718
TiRGN+	**0.548**	**0.436**	**0.607**	**0.750**	**0.754**	**0.679**	**0.807**	0.885	**0.727**
RETIA	0.499	0.360	0.586	0.795	0.782	0.701	**0.844**	0.924	0.745
RETIA+	**0.557**	**0.408**	**0.676**	**0.814**	**0.783**	**0.703**	0.842	**0.925**	**0.754**
CENET	0.419	0.297	0.522	0.593	0.753	0.682	0.808	0.869	0.710
CENET+	**0.591**	**0.451**	**0.687**	**0.844**	**0.779**	**0.692**	**0.849**	**0.912**	**0.755**
PPT	0.532	0.388	0.651	0.787	0.782	0.693	0.842	0.942	0.748
ICL	0.537	0.452	0.620	0.661	0.736	0.668	0.794	0.853	0.709

Table 15: Complete LP results on ACLED-zero. We also report PPT and ICL's performance.

J Failure Case Discussion

From Table 4, we observe several failure cases when the complete zrLLM is implemented, e.g., (1) TANGO-T+ without ERDs show a slightly better zero-shot result on ACLED-zero compared with the complete TANGO-T+; (2) TANGO-T+ does not witness an improvement over the seen relations on ICEWS21-zero compared with TANGO-T+ without RHL. We attribute such failure cases to the characteristics of the considered TKGF models. As highlighted in Sec. 4.2, our goal is to use zrLLM to enhance TKGF model performance over zero-shot relations while maintaining strong performance over seen relations. By carefully comparing the overall performance of zrLLM-enhanced models with their ablated variants, e.g., -ERD, we find that the complete version of zrLLM with ERDs, RHL and T5-11B can always achieve the best overall performance, which aligns to our motivation. The small number of failure cases caused by several baseline TKGF methods cannot overturn the merit brought by the modules of zrLLM.

K Related Work Details

Traditional TKG Forecasting Methods. As discussed in Sec. 1, traditional TKGF methods are trained to forecast the facts containing the KG relations (and entities) seen in the training data, regardless of the case where zero-shot relations (or entities) appear as new knowledge arrives[12]. These methods can be categorized into two types: embedding-based and rule-based. Embedding-based methods learn hidden representations of KG relations and entities (some also learn time representations), and perform link forecasting by inputting learned representations into a score function for computing scores of fact quadruples. Most existing embedding-based methods, e.g., (Jin et al., 2020; Han et al., 2021b; Li et al., 2021b, 2022; Liu et al., 2023), learn evolutional entity and relation representations by jointly employing graph neural networks (Kipf and Welling, 2017) and recurrent neural structures, e.g., GRU (Cho et al., 2014). Historical TKG information are recurrently encoded by the models to produce the temporal sequence-aware evolutional representations for future prediction. Some other approaches (Han et al., 2021a; Sun et al., 2021; Li et al., 2021a) start from each LP query and traverse the temporal history in a TKG to search for the prediction answer. Apart from them, CyGNet (Zhu et al., 2021) achieves forecasting purely based on the appearance of historical facts.

[12]Some works of traditional TKGF methods, e.g., TANGO (Han et al., 2021b), have discussions about models' ability to reason over the facts regarding unseen entities. Note that this is not their main focus but an additional demonstration to show their models' inductive power, i.e., these models are not designed for inductive learning on TKGs.

Datasets	ACLED-zero							ICEWS21-zero							ICEWS22-zero						
	Zero-Shot Relations			Seen Relations			Overall	Zero-Shot Relations			Seen Relations			Overall	Zero-Shot Relations			Seen Relations			Overall
Model	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR	MRR	Hits@1	Hits@10	MRR	Hits@1	Hits@10	MRR
CyGNet+	**0.533**	**0.418**	**0.753**	0.751	**0.664**	**0.906**	**0.723**	**0.201**	**0.103**	**0.415**	**0.258**	**0.162**	**0.447**	**0.257**	**0.286**	**0.167**	**0.542**	**0.315**	0.200	0.545	**0.314**
- ERD	0.502	0.386	0.743	0.748	0.660	0.902	0.716	0.198	0.102	0.379	0.252	0.161	0.429	0.251	0.250	0.136	0.503	0.314	0.198	**0.546**	0.311
- RHL	0.503	0.356	0.751	**0.752**	0.663	0.901	0.720	0.199	0.100	0.398	0.256	0.159	0.445	0.255	0.268	0.144	0.536	0.297	0.181	0.531	0.296
T5-3B	0.511	0.414	0.684	**0.752**	0.663	0.905	0.721	0.117	0.068	0.186	0.204	0.127	0.348	0.202	0.257	0.135	0.521	**0.315**	**0.201**	0.540	0.313
TANGO-T+	0.525	0.393	0.764	**0.775**	0.702	**0.901**	**0.743**	**0.216**	**0.125**	0.395	**0.280**	0.186	0.466	**0.279**	**0.326**	**0.198**	**0.578**	**0.363**	**0.251**	**0.585**	**0.362**
- ERD	0.533	0.408	**0.770**	0.772	0.692	0.898	0.741	0.214	0.122	0.389	**0.280**	**0.187**	0.465	**0.279**	0.320	0.193	0.576	0.362	0.250	0.584	0.360
- RHL	0.506	0.374	0.749	0.755	**0.704**	**0.901**	0.740	0.213	0.118	**0.407**	0.277	0.181	**0.469**	0.276	0.309	0.190	0.574	**0.363**	0.250	0.584	0.361
T5-3B	**0.544**	**0.425**	0.769	0.771	0.697	0.896	0.742	0.206	0.119	0.375	0.274	0.182	0.454	0.273	0.323	0.193	0.576	0.359	0.246	0.579	0.358
TANGO-D+	**0.491**	0.348	**0.791**	**0.760**	**0.678**	**0.901**	**0.725**	**0.212**	**0.122**	**0.400**	**0.268**	**0.175**	**0.453**	**0.267**	**0.311**	**0.186**	0.574	**0.350**	**0.239**	**0.570**	**0.348**
- ERD	**0.491**	**0.350**	0.771	0.702	0.578	0.898	0.675	0.205	0.111	0.398	0.267	0.174	0.449	0.266	0.285	0.159	0.541	0.328	0.213	0.550	0.326
- RHL	0.490	0.344	0.772	0.725	0.628	0.890	0.695	0.197	0.107	0.390	0.224	0.132	0.412	0.224	0.296	0.175	0.552	0.324	0.212	0.547	0.323
T5-3B	0.490	0.341	0.786	0.701	0.576	0.897	0.674	0.204	0.109	0.393	0.223	0.131	0.408	0.222	0.308	0.177	**0.582**	0.284	0.173	0.510	0.285
RE-GCN+	**0.529**	0.393	**0.784**	**0.731**	**0.650**	**0.876**	**0.705**	**0.214**	0.117	**0.406**	**0.280**	**0.188**	**0.456**	**0.279**	**0.324**	**0.194**	**0.595**	**0.357**	**0.244**	**0.573**	**0.356**
- ERD	0.489	0.375	0.724	0.730	**0.650**	0.865	0.699	0.211	0.119	0.397	0.277	0.185	0.454	0.276	0.294	0.168	0.560	0.354	0.242	0.571	0.352
- RHL	0.519	**0.396**	0.757	0.726	0.646	0.836	0.699	0.213	0.119	0.405	0.277	0.185	0.455	0.276	0.317	0.184	0.589	0.350	0.241	0.562	0.349
T5-3B	0.504	0.361	0.767	0.721	0.638	0.864	0.693	0.211	**0.121**	0.384	0.259	0.171	0.427	0.258	0.301	0.174	0.577	0.354	0.243	0.570	0.352
TiRGN+	**0.548**	**0.436**	0.750	**0.754**	**0.679**	0.885	**0.727**	**0.221**	**0.130**	**0.410**	**0.279**	**0.185**	**0.463**	**0.278**	**0.333**	**0.203**	**0.602**	**0.353**	**0.240**	**0.577**	**0.352**
- ERD	0.480	0.387	0.673	0.747	0.669	0.882	0.713	0.211	0.120	0.387	0.275	0.181	0.460	0.274	0.282	0.157	0.544	**0.353**	**0.240**	0.576	0.350
- RHL	0.515	0.400	**0.753**	0.752	0.675	**0.887**	0.721	0.215	0.124	0.391	0.277	0.183	0.461	0.276	0.320	0.190	0.593	0.350	0.239	0.569	0.349
T5-3B	0.498	0.389	0.722	0.749	0.675	0.879	0.717	0.208	0.118	0.392	0.271	0.180	0.448	0.270	0.325	0.189	0.594	0.345	0.233	0.565	0.344
RETIA+	**0.557**	**0.408**	**0.814**	**0.783**	**0.703**	**0.925**	**0.754**								**0.331**	**0.201**	**0.597**	**0.358**	**0.247**	0.578	**0.357**
- ERD	0.519	0.391	0.765	0.777	0.692	0.917	0.744								0.292	0.163	0.562	0.354	0.242	0.576	0.352
- RHL	0.529	0.368	0.796	0.782	0.701	0.923	0.749			» 120 Hours Timeout					0.318	0.191	0.583	0.357	0.244	**0.580**	0.355
T5-3B	0.512	0.385	0.766	0.776	0.690	0.917	0.742								0.330	0.200	0.595	0.353	0.242	0.573	0.352
CENET+	**0.591**	**0.451**	**0.844**	**0.779**	**0.692**	**0.912**	**0.755**	**0.335**	**0.162**	0.659	**0.396**	**0.239**	**0.688**	**0.395**	**0.564**	**0.432**	**0.801**	**0.571**	**0.451**	0.773	**0.570**
- ERD	0.526	0.373	0.785	0.737	0.653	0.870	0.710	0.321	0.156	**0.665**	0.374	0.216	0.683	0.373	0.542	0.388	0.799	0.570	0.448	**0.774**	0.568
- RHL	0.445	0.367	0.565	0.754	0.685	0.862	0.714	0.232	0.128	0.446	0.290	0.202	0.469	0.289	0.295	0.168	0.560	0.370	0.262	0.588	0.367
T5-3B	0.568	0.426	0.819	0.736	0.646	0.900	0.714	0.303	0.158	0.568	0.330	0.203	0.712	0.329	0.550	0.413	0.798	0.555	0.431	0.765	0.554

Table 16: Complete results of ablation studies.

Another recent work CENET (Xu et al., 2023b) trains contrastive representations of LP queries to identify highly correlated entities in either historical or non-historical facts. Compared with the rapid advancement in developing embedding-based TKGF methods, rule-based TKGF has still not been extensively explored. One popular rule-based TKGF method is TLogic (Liu et al., 2022). It extracts temporal logic rules from TKGs and uses a symbolic reasoning module for LP. Based on it, ALRE-IR (Mei et al., 2022) proposes an adaptive logical rule embedding model to encode temporal logical rules into rule representations. This makes ALRE-IR both a rule-based and an embedding-based method. Experiments in TLogic and ALRE-IR have proven that rule-based TKGF methods have strong ability in reasoning over zero-shot unseen entities connected by the seen relations, however, they are not able to handle unseen relations since the learned rules are strongly bounded by the observed relations. In our work, we implement zrLLM on embedding-based TKGF models because (1) embedding-based methods are much more popular; (2) zrLLM utilizes LLM to generate relation representations, which is more compatible with embedding-based methods.

Inductive Learning on TKGs. Inductive learning on TKGs has gained increasing interest. It refers to developing models that can handle the relations and entities unseen in the training data. TKG inductive learning methods can be categorized into two types. The first type of works focuses on reasoning over unseen entities (Ding et al., 2022; Wang et al., 2022; Ding et al., 2023c; Chen et al., 2023a), while the second type of methods aims to deal with the unseen relations (Mirtaheri et al., 2021; Ding et al., 2023a; Ma et al., 2023). Most of inductive learning methods are based on few-shot learning (e.g., FILT (Ding et al., 2022), MetaTKGR (Zhang et al., 2019), FITCARL (Ding et al., 2023c), OAT (Mirtaheri et al., 2021), MOST (Ding et al., 2023a) and OSLT (Ma et al., 2023)). They first compute inductive representations of newly-emerged entities or relations based on K-associated facts (K is a small number, e.g., 1 or 3) observed during inference, and then use them to predict the facts regarding few-shot elements. One limitation of these works is that the inductive representations cannot be learned without the K-shot examples, making them hard to solve the zero-shot problems. Different from few-shot learning methods, SST-BERT (Chen et al., 2023a) pre-trains a time-enhanced BERT (Devlin et al., 2019) for TKG reasoning. It achieves inductive learning over unseen entities but has not shown its ability in reasoning zero-shot relations. Another recent work MTKGE (Chen et al., 2023b) is able to concurrently deal with both unseen entities and relations. However, it requires a support graph containing a substantial number of data examples related to the unseen entities and relations, which is far from the zero-shot problem that we focus on.

TKG Reasoning with Language Models. Recently, more and more works have introduced LMs into TKG reasoning. SST-BERT (Chen et al., 2023a) generates a small-scale pre-training corpus

based on the training TKGs and pre-trains an LM for encoding TKG facts. The encoded facts are then fed into a scoring module for LP. ECOLA (Han et al., 2023) aligns facts with additional fact-related texts and proposes a joint training framework that enhances TKG reasoning with BERT-encoded language representations. PPT (Xu et al., 2023a) converts TKGF into the pre-trained LM masked token prediction task and finetunes a BERT for TKGF. It directly input TKG facts into the LM for answer prediction. Apart from them, one recent work (Lee et al., 2023) explores the possibility of using in-context learning (ICL) (Brown et al., 2020) with LLMs to make predictions about future facts without fintuning. Another recent work GenTKG (Liao et al., 2023) finetunes an LLM, i.e., Llama2-7B (Touvron et al., 2023), and let the LLM directly generate the LP answer in TKGF. It mines temporal logical rules and uses them to retrieve historical facts for prompt generation.

Although the above-mentioned works have shown success of LMs in TKG reasoning, they have limitations: (1) None of these works has studied whether LMs can be used to better reason the zero-shot relations. (2) By only using ICL, LLMs are beaten by traditional TKG reasoning methods in performance (Lee et al., 2023). The performance can be greatly improved by finetuning LLMs (as in GenTKG (Liao et al., 2023)), but finetuning LLMs requires huge computational resources. (3) Since LMs, e.g., BERT and Llama2, are pre-trained with a huge corpus originating from diverse information sources, it is inevitable that they have already seen the world knowledge before they are used to solve TKG reasoning tasks. Most popular TKGF benchmarks are extracted from the TKGs constructed before 2020, e.g., ICEWS14, ICEWS18 and ICEWS05-15 (Jin et al., 2020). The facts inside are based on the world knowledge before 2019, which means LMs might have encountered them in their training corpus, posing a threat of information leak to the LM-driven TKG reasoning models. To this end, we (1) draw attention to studying the impact of LMs on zero-shot relational learning in TKGs; (2) make a compromise between performance and computational efficiency by not fintuning LMs or LLMs but adapting the LLM-provided semantic information to non-LM-based TKGF methods; (3) construct new benchmarks where the facts are all happening from 2021 to 2023, which avoids the possibility of information leak when we utilize T5-11B that was released in 2020.

Chapter 6

ForecastTKGQuestions: A Benchmark for Temporal Question Answering and Forecasting over Temporal Knowledge Graphs

This chapter contains the publication

> **Zifeng Ding***, Zongyue Li*, Ruoxia Qi*, Jingpei Wu, Bailan He, Yunpu Ma, Zhao Meng, Shuo Chen, Ruotong Liao, Zhen Han, Volker Tresp. ForecastTKGQuestions: A Benchmark for Temporal Question Answering and Forecasting over Temporal Knowledge Graphs. In *International Semantic Web Conference*, 2023. *Equal Contribution. DOI: 10.1007/978-3-031-47240-4_29

ForecastTKGQuestions: A Benchmark for Temporal Question Answering and Forecasting over Temporal Knowledge Graphs

Zifeng Ding[1,2], Zongyue Li[1,3], Ruoxia Qi[1], Jingpei Wu[1], Bailan He[1,2], Yunpu Ma[1,2], Zhao Meng[4], Shuo Chen[1,2], Ruotong Liao[1,3], Zhen Han[1(✉)], and Volker Tresp[1(✉)]

[1] LMU Munich, Geschwister-Scholl-Platz 1, 80539 Munich, Germany
{zifeng.ding,ruoxia.qi,bailan.he,shuo.chen}@campus.lmu.de, {zongyue.li,jingpei.wu}@outlook.com, cognitive.yunpu@gmail.com, liao@dbs.ifi.lmu.de, hanzhen02111@hotmail.com, Volker.Tresp@lmu.de
[2] Siemens AG, Otto-Hahn-Ring 6, 81739 Munich, Germany
[3] Munich Center for Machine Learning (MCML), Munich, Germany
[4] ETH Zürich, Rämistrasse 101, 8092 Zürich, Switzerland
zhmeng@ethz.ch

Abstract. Question answering over temporal knowledge graphs (TKGQA) has recently found increasing interest. Previous related works aim to develop QA systems that answer temporal questions based on the facts from a fixed time period, where a temporal knowledge graph (TKG) spanning this period can be fully used for inference. In real-world scenarios, however, it is common that given knowledge until the current instance, we wish the TKGQA systems to answer the questions asking about future. As humans constantly plan the future, building forecasting TKGQA systems is important. In this paper, we propose a novel task: forecasting TKGQA, and propose a coupled large-scale TKGQA benchmark dataset, i.e., ForecastTKGQuestions. It includes three types of forecasting questions, i.e., entity prediction, yes-unknown, and fact reasoning questions. For every question, a timestamp is annotated and QA models only have access to TKG information prior to it for answer inference. We find that previous TKGQA methods perform poorly on forecasting questions, and they are unable to answer yes-unknown and fact reasoning questions. To this end, we propose ForecastTKGQA, a TKGQA model that employs a TKG forecasting module for future inference. Experiments show that it performs well in forecasting TKGQA.

1 Introduction

Knowledge graphs (KGs) model factual information by representing every fact with a triple, i.e., (s, r, o), where s, o, r, are the subject entity, the object entity,

Z. Ding, Z. Li and R. Qi— Equal contribution.

T. R. Payne et al. (Eds.): ISWC 2023, LNCS 14265, pp. 541–560, 2023.
https://doi.org/10.1007/978-3-031-47240-4_29

and the relation between s and o, respectively. To adapt to the ever-evolving knowledge, temporal knowledge graphs (TKGs) are introduced, where they additionally specify the time validity of every fact with a time constraint t (e.g., a timestamp), and represent each fact with a quadruple (s, r, o, t). Recently, TKG reasoning has drawn increasing attention. While a lot of methods focus on temporal knowledge graph completion (TKGC) where they predict missing facts at the observed timestamps, various recent methods pay more attention to forecasting the facts at unobserved future timestamps in TKGs.

Knowledge graph question answering (KGQA) is a task aiming to answer natural language questions using a KG as the knowledge base (KB). KGQA requires QA models to extract answers from KGs, rather than retrieving or summarizing answers from text contexts. [21] first introduces question answering over temporal knowledge graphs (TKGQA). It proposes a non-forecasting TKGQA dataset CronQuestions that takes a TKG as its underlying KB. Temporal reasoning techniques are required to answer these questions. Though [21] manages to combine TKG reasoning with KGQA, it has limitations. Previous KGQA datasets, including CronQuestions, do not include yes-no and multiple-choice questions, while these two question types have been extensively studied in reading comprehension QA, e.g., [13]. Besides, the questions in CronQuestions are in a non-forecasting style, where all questions are based on the TKG facts that happen in a fixed time period, and an extensive TKG that is fully observable in this period can be used to infer the answers, making the answer inference less challenging. For example, the TKG facts from *2003*, including (*Stephen Robert Jordan*, *member of sports team*, *Manchester City*, *2003*), are all observable to answer the question *Which team was Stephen Robert Jordan part of in 2003?*. CronQuestions manages to bridge the gap between TKGC and KGQA, however, no previous work manages to combine TKG forecasting with KGQA, where only past TKG information can be used for answer inference.

In this work, we propose a novel task: forecasting question answering over temporal knowledge graphs (forecasting TKGQA), together with a coupled large-scale dataset, i.e., ForecastTKGQuestions. We generate forecasting questions based on the Integrated Crisis Early Warning System (ICEWS) Dataverse [2], and label every question with a timestamp. To answer a forecasting question, QA models can only access the TKG information prior to the question timestamp. The contribution of our work is three-folded: (1) We propose forecasting TKGQA, a novel task aiming to test the forecasting ability of TKGQA models. To the best of our knowledge, this is the first work binding TKG forecasting with temporal KGQA; (2) We propose a large-scale benchmark TKGQA dataset: ForecastTKGQuestions. It contains three types of questions, i.e., entity prediction questions (EPQs), yes-unknown questions (YUQs), and fact reasoning questions (FRQs), where the last two types of questions have never been considered in previous KGQA datasets[1]; (3) We propose ForecastTKGQA, a model aiming to solve forecasting TKGQA. It employs a TKG forecasting module and a pre-trained language model (LM) for answer inference. Experimental results show that it achieves great performance on forecasting questions.

[1] YUQs are based on yes-no questions and FRQs are multiple-choice questions.

2 Preliminaries and Related Work

TKG Reasoning. Let $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{T}$ denote a finite set of entities, relations, and timestamps, respectively. A TKG $\mathcal{G}$ is defined as a finite set of TKG facts represented by quadruples, i.e., $\mathcal{G} = \{(s, r, o, t)|s, o \in \mathcal{E}, r \in \mathcal{R}, t \in \mathcal{T}\}$. We define the TKG forecasting task (also known as TKG extrapolation) as follows. Assume we have a query $(s_q, r_q, ?, t_q)$ (or $(?, r_q, o_q, t_q)$) derived from a target quadruple (s_q, r_q, o_q, t_q), and we denote all the ground-truth quadruples as $\mathcal{F}$. TKG forecasting aims to predict the missing entity in the query, given the observed **past** TKG facts $\mathcal{O} = \{(s_i, r_i, o_i, t_i) \in \mathcal{F}|t_i < t_q\}$. Such temporal restriction is not imposed in TKG completion (TKGC, also known as TKG interpolation), where the observed TKG facts from any timestamp, including t_q and the timestamps after t_q, can be used for prediction. In recent years, there have been extensive works done for both TKGC [6,15,16] and TKG forecasting [8,9,14,18,30]. We give a more detailed discussion about the forecasting methods. RE-NET [14] employs an autoregressive architecture and models fact occurrence as a probability distribution conditioned on the temporal sequences of past related TKG information. TANGO [9] employs neural ordinary differential equations to model temporal dependencies among graph information of different timestamps. CyGNet [30] uses the copy-generation mechanism to extract hints from historical facts for forecasting. xERTE [8] constructs a historical fact-based subgraph and selects prediction answers from it. TLogic [18] is the first rule-based TKG forecasting method that learns temporal logical rules in TKGs and achieves superior results.

Question Answering over KGs. Several datasets have been proposed for QA over non-temporal KGs, such as SimpleQuestions [1], WebQuestionsSP [28], ComplexWebQuestions [24], MetaQA [29], TempQuestions [11], and TimeQuestions [12]. Among these datasets, only TempQuestions and TimeQuestions involve temporal questions that require temporal reasoning for answer inference, however, their associated KGs are non-temporal. CronQuestions [21] contains questions based on a time-evolving TKG, i.e., Wikidata [27]. It is proposed for non-forecasting TKGQA. Two types of questions, i.e., entity prediction and time prediction questions, are included. To answer CronQuestions, Saxena et al. propose CronKGQA that uses TKGC methods, along with pre-trained LMs, which shows great effectiveness. A line of methods has been proposed on top of CronKGQA (TempoQR [19], TSQA [23], SubGTR [4]), where they better distinguish question time scopes and reason over subgraphs. CronQuestions is proposed based on the idea of TKGC, and it does not support TKG forecasting and contains no forecasting questions. One recent work, i.e., ForecastQA [13], proposes a QA dataset fully consisting of forecasting questions. However, ForecastQA is not related to KGQA. In ForecastQA, answers to its questions are inferred from text contexts, while KGQA/TKGQA requires models to find the answers from the coupled KGs/TKGs without providing any additional text contexts. As a result, the methods designed for ForecastQA have no ability to address TKGQA. To this end, we propose ForecastTKGQuestions,

Table 1. (a) KGQA dataset comparison. Statistics are taken from [12,21]. **T%** denotes the portion of temporal questions. (b) ForecastTKGQuestions statistics: number of questions of different types.

(a)

Datasets	TKG	Forecast	T%	# Questions
MetaQA	✗	✗	0%	400k
TempQuestions	✗	✗	100%	1271
TimeQuestions	✗	✗	100%	16k
CronQuestions	✓	✗	100%	410k
ForecastTKGQuestions	✓	✓	100%	727k

(b)

	Train	Valid	Test
1-Hop Entity Prediction	211,564	36,172	33,447
2-Hop Entity Prediction	85,088	12,266	10,765
Yes-Unknown	251,537	42,884	39,695
Fact Reasoning	3,164	514	517
Total	551,353	91,836	84,424

aiming to bridge the gap between TKG forecasting and KGQA. We compare ForecastTKGQuestions with recent KGQA datasets in Table 1.

Task Formulation: Forecasting TKGQA. Forecasting TKGQA aims to test the forecasting ability of TKGQA models. It requires QA models to predict future facts based on past TKG information. We formulate it as follows. Given a TKG $\mathcal{G}$ and a natural language question q generated based on a TKG fact whose valid timestamp is t_q, forecasting TKGQA aims to predict the answer to q. We label every question q with t_q, and constrain QA models to only use the TKG facts $\{(s_i, r_i, o_i, t_i)|t_i < t_q\}$ before t_q for answer inference. We propose three types of forecasting TKGQA questions, i.e., EPQs, YUQs, and FRQs. The answer to a EPQ is an entity $e \in \mathcal{E}$. The answer to a YUQ is either *yes* or *unknown*. We formulate FRQs as multiple choices and thus the answer to an FRQ corresponds to a choice c. As a novel task, forecasting TKGQA requires models to have the ability of both natural language understanding (NLU) and future forecasting. Compared with it, the traditional TKG forecasting task does not require NLU and non-forecasting TKGQA does not consider future forecasting. Thus, previous methods for TKG forecasting[2], e.g., RE-Net [14], and non-forecasting TKGQA, e.g., TempoQR [19], are not suitable for solving forecasting TKGQA.

3 ForecastTKGQuestions

3.1 Temporal Knowledge Base

A subset from ICEWS [2] is taken as the associated temporal KB for our proposed dataset. We construct a TKG ICEWS21 based on the events taken from the official website of the ICEWS weekly event data[3] [2]. ICEWS contains socio-political events in English. We take the events from Jan. 1, 2021, to Aug. 31,

[2] Relation set is provided in TKG forecasting and these methods explicitly learn relation representations. However, TKG relations are not annotated in forecasting TKGQA questions. Only question texts are provided and these methods have no way to process. Therefore, we do not consider them in experiments on our new task.

[3] https://dataverse.harvard.edu/dataverse/icews.

Table 2. ICEWS21 TKG statistics. N_{train}, N_{valid}, N_{test} denote the number of TKG facts in $\mathcal{G}_{\text{train}}$, $\mathcal{G}_{\text{valid}}$, $\mathcal{G}_{\text{test}}$, respectively. $|\mathcal{E}|$, $|\mathcal{R}|$, $|\mathcal{T}|$ denote ICEWS21's number of entities, relations, timestamps, respectively.

Dataset	N_{train}	N_{valid}	N_{test}	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{R}\rvert$	$\lvert\mathcal{T}\rvert$
ICEWS21	252,434	43,033	39,836	20,575	253	243

2021, and extract TKG facts in the following way. For every ICEWS event, we generate a TKG fact (s, r, o, t). We take the content of *Event Date* as the timestamp t of the TKG fact. We take the contents of *Source Name* and *Target Name* as the subject entity s and the object entity o of the TKG fact, respectively. We take the content of *Event Text* as the relation type r of the fact. We present the dataset statistics of ICEWS21 in Table 2. We split ICEWS21 into three parts $\mathcal{G}_{\text{train}} = \{(s, r, o, t) \in \mathcal{G} | t \in [t_0, t_1)\}$, $\mathcal{G}_{\text{valid}} = \{(s, r, o, t) \in \mathcal{G} | t \in [t_1, t_2)\}$, $\mathcal{G}_{\text{test}} = \{(s, r, o, t) \in \mathcal{G} | t \in [t_2, t_3]\}$, where t_0, t_1, t_2, t_3 correspond to *2021-01-01*, *2021-07-01*, *2021-08-01* and *2021-08-31*, respectively. We generate training/validation/test questions based on $\mathcal{G}_{\text{train}}/\mathcal{G}_{\text{valid}}/\mathcal{G}_{\text{test}}$. We ensure that there exists no temporal overlap between every two of them, i.e., $\mathcal{G}_{\text{train}} \cap \mathcal{G}_{\text{valid}} = \emptyset$, $\mathcal{G}_{\text{train}} \cap \mathcal{G}_{\text{test}} = \emptyset$ and $\mathcal{G}_{\text{valid}} \cap \mathcal{G}_{\text{test}} = \emptyset$. In this way, we prevent QA models from observing any information from the evaluation sets during training.

3.2 Question Categorization and Generation

We generate natural language questions based on the TKG facts in ICEWS21 and propose our QA dataset ForecastTKGQuestions. Every relation type in ICEWS21 is coupled with a CAMEO code (specified in the *CAMEO Code* column of the ICEWS weekly event data). In the official CAMEO codebook (can be found in ICEWS database), each CAMEO code is explained with examples and detailed descriptions. We use the official CAMEO codebook provided in the ICEWS dataverse for aiding the generation of natural language relation templates. We create relation templates for 250 out of 253 relation types for question generation[4]. For example, we create a relation template *engage in material cooperation with* for the relation type *engage in material cooperation, not specified below*. Questions in ForecastTKGQuestions are categorized into three categories, i.e., EPQs (including 1-hop and 2 hop EPQs), YUQs, and FRQs. We summarize the number of different types of questions in Table 1b. We use the relation templates to create natural language question templates for all types of questions (examples in Table 3) which are used for question generation. All question templates are presented in our supplementary source code and explained in Appendix C.2. Similar to previous KGQA datasets, e.g., CronQuestions, entity linking is considered as a separate problem and is not covered in our work. We assume complete entity and timestamp linking, and annotate the entities and timestamps in our questions. This applies to all three types of questions in our dataset. Distribution of question timestamps is specified in Appendix C.5.

[4] The rest three relation types are not ideal for question generation (Appendix C.1).

Table 3. Example question templates of all types. s_q and o_q are the annotated question entities. t_q is the annotated question timestamp. For FRQ, s_c, o_c, t_c are annotated choice entities and timestamp. We only write one choice in FRQ template for brevity. Better understand with details in Sect. 3.2.

Question Type	Example Template
1-Hop EPQ	*Who will* $\{s_q\}$ *engage in material cooperation with on* $\{t_q\}$*?*
2-Hop EPQ	*Who will threaten a country, while* $\{s_q\}$ *criticizes or denounces this country on* $\{t_q\}$*?*
YUQ	*Will* $\{s_q\}$ *make a pessimistic comment about* $\{o_q\}$ *on* $\{t_q\}$*?*
FRQ	*Why will* $\{s_q\}$ *appeal to* $\{o_q\}$ *to meet or negotiate on* $\{t_q\}$*?* *A:* $\{s_c\}$ *threatens* $\{o_c\}$ *on* $\{t_c\}$*; B:...*

Entity Prediction Questions. We generate two groups of EPQs, i.e., 1-hop and 2-hop EPQs. Each 1-hop EPQ is generated from a single TKG fact, e.g., the natural language question *Who will Sudan host on 2021-08-01?* is based on (*Sudan*, *host*, *Ramtane Lamamra*, *2021-08-01*). Question templates are used during question generation. The underlined parts in the question denote the annotated entities and timestamps for KGQA. We consider all the facts concerning the 250 selected relations and transform them into 1-hop EPQs. Each 2-hop EPQ is generated from two associated TKG facts in ICEWS21 where they contain common entities. An example is presented in Table 4. The answer to a 2-hop EPQ (*Israel*) corresponds to a 2-hop neighbor of its annotated entity (*Iran*) at the question timestamp (*2021-08-02*). We generate 2-hop questions by utilizing AnyBURL [20], a rule-based KG reasoning model. We first split ICEWS21 into snapshots, where each snapshot $\mathcal{G}_{t_i} = \{(s, r, o, t) \in \mathcal{G} | t = t_i\}$ contains all the TKG facts happening at the same timestamp. Then we train AnyBURL on each snapshot for rule extraction. We collect the 2-hop rules with a confidence higher than 0.5 returned by AnyBURL, and manually check if two associated TKG facts in each rule potentially have a logical causation or can be used to interpret positive/negative entity relationships. After excluding the rules not meeting this requirement, we create question templates based on the remaining ones. We search for the groundings in ICEWS21 at every timestamp, where each grounding corresponds to a 2-hop EPQ. See our source code for the complete list of extracted 2-hop rules and see Appendix C.3 for more EPQ generation details.

Yes-Unknown Questions. Based on the idea of triple classification in KG reasoning[5], we introduce yes-no questions into KGQA. We then turn yes-no questions into yes-unknown questions because, according to the Open World Assumption (OWA), the facts not observed in a given TKG are not necessarily wrong [7]. We generalize triple classification to quadruple classification[6], and then translate TKG facts into natural language questions. We take answering YUQs as solving

[5] For a KG fact (s, r, o), triple classification aims to predict whether this fact is valid or not.

[6] Quadruple classification has never been studied in previous works. We define it as predicting whether a TKG fact (s, r, o, t) is valid or unknown, under OWA.

Table 4. 2-hop EPQ example. To avoid overlong text, we use symbols to represent relations and timestamps in TKG facts and 2-hop rules. r_1 =*accuse*; r_2 =*engage in diplomatic cooperation*; t_1 =*2021-08-02*. m, n are two entities that are 2-hop neighbors of each other at t_1. X is their common 1-hop neighbor at t_1. The extracted rule describes the negative relationship between *Iran* and *Israel*.

Associated TKG Facts	2-Hop Rule	Generated 2-Hop Question	Answer
(*United States*, r_1, *Iran*, t_1) (*United States*, r_2, *Israel*, t_1)	(X, r_1, m) $=> (X, r_2, n)$	*Who will a country engage in diplomatic cooperation with, while this country accuses Iran on 2021-08-02?*	*Israel*

quadruple classification. For every TKG fact concerning the selected 250 relations, we generate either a true or an unknown question based on it. For example, for the fact (*Sudan*, *host*, *Ramtane Lamamra*, *2021-08-01*), a true question is generated as *Will Sudan host Ramtane Lamamra on 2021-08-01?* and we label *yes* as its answer. An unknown question is generated by randomly perturbing one entity or the relation type in this fact, e.g., *Will Germany host Ramtane Lamamra on 2021-08-01?*, and we label *unknown* as its answer. We ensure that the perturbed fact does not exist in the original TKG. We use 25% of total facts in ICEWS21 to generate true questions and the rest are used to generate unknown questions.

Fact Reasoning Questions. The motivation for proposing FRQs is to study the difference between humans and machines in finding supporting evidence for reasoning. We formulate FRQs in the form of multiple choices. Each question is coupled with four choices. Given a TKG fact from an FRQ, we ask the QA models to choose which fact in the choices is the most contributive to (the most relevant cause of) the fact mentioned in the question. We provide several examples in Fig. 1. We generate FRQs as follows. We first train a TKG forecasting model xERTE [8] on ICEWS21. Note that to predict a query $(s, r, ?, t)$, xERTE samples its related prior TKG facts and assigns contribution scores to them. It provides explainability by assigning higher scores to the more related prior facts. We perform TKG forecasting and collect the queries where the ground-truth missing entities are ranked as top 1 by xERTE. For each collected query, we find its corresponding TKG fact and pick out four related prior facts found by xERTE. We take the prior facts with the highest, the lowest, and median contribution scores as **Answer**, **Negative**, and **Median**, respectively. Inspired by InferWiki [3], we include a **Hard Negative** fact with the second highest contribution score, making it non-trivial for QA models to make the right decision. We generate each FRQ by turning the corresponding facts into a question and four choices (using templates), and manage to use xERTE to generate a large number of questions. However, since the answers to these questions are solely determined by xERTE, there exist numerous erroneous examples. For example, the **Hard Negative** of lots of them are more suitable than their **Answer** to be the answers. We ask five graduate students (major in computer science) to manually check all these questions and annotate them as reasonable or unreasonable according to their own knowledge or through search engines. If the majority annotate a question

as unreasonable, we filter it out. See Appendix C.4 for more details of FRQ generation and annotation, including the annotation instruction and interface.

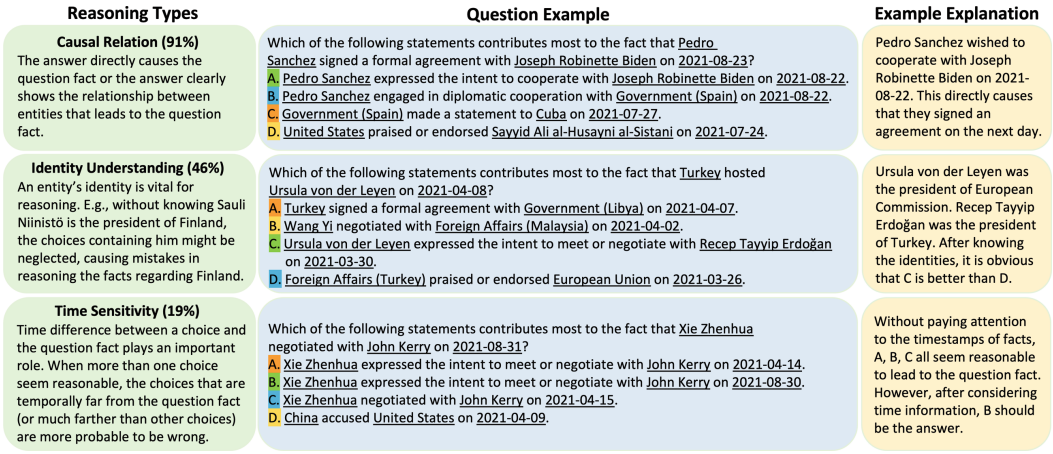


Fig. 1. Required reasoning types and proportions (%) in sampled FRQs, as well as FRQ examples. We sample 100 FRQs in each train/valid/test set. For choices, green for **Answer**, blue for **Hard Negative**, orange for **Median** and yellow for **Negative**. Multiple reasoning skills are required to answer each question, so the total proportion sum is not 100%. (Color figure online)

To better study the reasoning skills required to answer FRQs, we randomly sample 300 FRQs and manually annotate them with reasoning types. The required reasoning skills and their proportions are shown in Fig. 1.

4 ForecastTKGQA

ForecastTKGQA employs a TKG forecasting model TANGO [9] and a pre-trained LM BERT [5] for solving forecasting questions. We illustrate its model structure in Fig. 2 with three stages. In Stage 1, a TKG forecasting model TANGO [9] is used to generate the time-aware representation for each entity at each timestamp. In Stage 2, a pre-trained LM (e.g., BERT) is used to encode questions (and choices) into question (choice) representations. Finally, in Stage 3, answers are predicted according to the scores computed using the representations from Stage 1 and 2.

4.1 TKG Forecasting Model

We train TANGO on ICEWS21 with the TKG forecasting task. We use ComplEx [26] as its scoring function. We learn the entity and relation representations in the complex space $\mathbb{C}^d$, where d is the dimension of complex vectors. The training set corresponds to all the TKG facts in $\mathcal{G}_{\text{train}}$, and we evaluate the trained model on $\mathcal{G}_{\text{valid}}$ and $\mathcal{G}_{\text{test}}$. After training, we perform a one time inference on $\mathcal{G}_{\text{valid}}$ and $\mathcal{G}_{\text{test}}$. Following the default setting of TANGO, to compute entity and

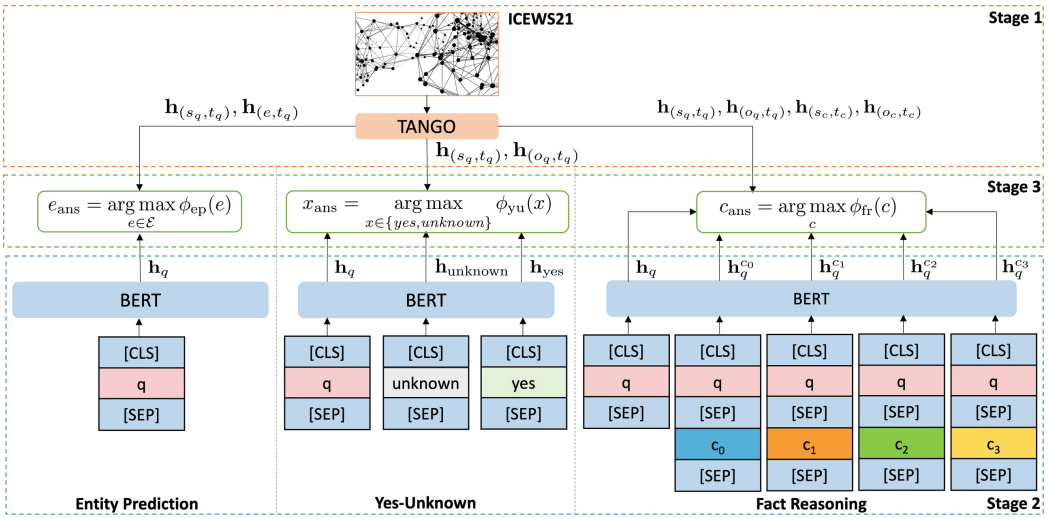


Fig. 2. Model structure of ForecastTKGQA.

relation representations at every timestamp t, we recurrently input all the TKG facts from $t-4$ to $t-1$, i.e., snapshots from $\mathcal{G}_{t-4}$ to $\mathcal{G}_{t-1}$, into TANGO and take the output representations. Note that it infers representations based on the prior facts, thus not violating our forecasting setting. We compute the entity and relation representations at every timestamp in ICEWS21 and keep them for aiding the QA systems in Stage 1 (Fig. 2). See Appendix B.1 for more details of TANGO training and inference. To leverage the complex representations computed by TANGO with ComplEx, we map the output of BERT to $\mathbb{C}^d$. For each natural language input, we take the output representation of the [CLS] token computed by BERT and project it to a $2d$ real space to form a $2d$ real-valued vector. We take the first and second half of it as the real and imaginary part of a d-dimensional complex vector, respectively. All the representations output by BERT have already been mapped to $\mathbb{C}^d$ without further notice.

4.2 QA Model

Entity Prediction. For every EPQ q, we compute an entity score for every entity $e \in \mathcal{E}$. The entity with the highest score is predicted as the answer e_{ans}. To compute the score for e, we first input q into BERT and map its output to $\mathbb{C}^d$ to get the question representation $\mathbf{h}_q$. Inspired by ComplEx, we then define e's entity score as

$$\phi_{\text{ep}}(e) = \text{Re}\left(< \mathbf{h}'_{(s_q,t_q)}, \mathbf{h}_q, \bar{\mathbf{h}}'_{(e,t_q)} >\right). \tag{1}$$

$\mathbf{h}'_{(s_q,t_q)} = f_{\text{ep}}\left(\mathbf{h}_{(s_q,t_q)}\right)$, $\mathbf{h}'_{(e,t_q)} = f_{\text{ep}}\left(\mathbf{h}_{(e,t_q)}\right)$, where f_{ep} denotes a neural network aligning TKG representations to EPQs. $\mathbf{h}_{(s_q,t_q)}$ and $\mathbf{h}_{(e,t_q)}$ denote the TANGO representations of the annotated entity s_q and the entity e at the question timestamp t_q, respectively. Re means taking the real part of a complex vector and $\bar{\mathbf{h}}'_{(e,t_q)}$ means the complex conjugate of $\mathbf{h}'_{(e,t_q)}$.

Yes-Unknown Judgment. For a YUQ, we compute a score for each candidate answer $x \in \{yes, unknown\}$. We first encode each x into a d-dimensional complex representation $\mathbf{h}_x$ with BERT. Inspired by TComplEx [16], we then compute scores as

$$\phi_{\text{yu}}(x) = \text{Re}\left(< \mathbf{h}'_{(s_q,t_q)}, \mathbf{h}_q, \bar{\mathbf{h}}'_{(o_q,t_q)}, \mathbf{h}_x >\right). \tag{2}$$

$\mathbf{h}'_{(s_q,t_q)} = f_{\text{yu}}\left(\mathbf{h}_{(s_q,t_q)}\right)$,$\mathbf{h}'_{(o_q,t_q)} = f_{\text{yu}}\left(\mathbf{h}_{(o_q,t_q)}\right)$, where f_{yu} denotes a neural network aligning TKG representations to YUQs. $\mathbf{h}_{(s_q,t_q)}$ and $\mathbf{h}_{(o_q,t_q)}$ denote the TANGO representations of the annotated subject entity s_q and object entity o_q at t_q, respectively. $\mathbf{h}_q$ is the BERT encoded question representation. We take the candidate answer with the higher score as the predicted answer x_{ans}.

Fact Reasoning. We compute a choice score for every choice c in an FRQ by using the following scoring function:

$$\phi_{\text{fr}}(c) = \text{Re}\left(< \mathbf{h}'_{(s_c,t_c)}, \mathbf{h}^c_q, \bar{\mathbf{h}}'_{(o_c,t_c)}, \mathbf{h}'_q >\right), \tag{3}$$

$\mathbf{h}^c_q$ is the output of BERT mapped to $\mathbb{C}^d$ given the concatenation of q and c. $\mathbf{h}'_{(s_c,t_c)} = f_{\text{fr}}\left(\mathbf{h}_{(s_c,t_c)}\right)$ and $\mathbf{h}'_{(o_c,t_c)} = f_{\text{fr}}\left(\mathbf{h}_{(o_c,t_c)}\right)$. f_{fr} is a projection network and $\mathbf{h}_{(s_c,t_c)}$, $\mathbf{h}_{(o_c,t_c)}$ denote the TANGO representations of the entities annotated in c. $\mathbf{h}'_q = f\left(f_{\text{fr}}\left(\mathbf{h}_{(s_q,t_q)}\right) \| \mathbf{h}^c_q \| f_{\text{fr}}\left(\mathbf{h}_{(o_q,t_q)}\right)\right)$, where f serves as a projection and $\|$ denotes concatenation. $\mathbf{h}_{(s_q,t_q)}$ and $\mathbf{h}_{(o_q,t_q)}$ denote the TANGO representations of the entities annotated in the question q. We take the choice with the highest choice score as our predicted answer c_{ans}. We give a more detailed description of Eq. 1, 2 and 3 in Appendix A.

Parameter Learning. We use cross-entropy loss to train ForecastTKGQA on each type of questions separately. The loss functions of EPQs, FRQs and YUQs are given by $\mathcal{L}_{\text{ep}} = -\sum_{q\in\mathcal{Q}^{\text{ep}}} \log\left(\frac{\phi_{\text{ep}}(e_{\text{ans}})}{\sum_{e\in\mathcal{E}}\phi_{\text{ep}}(e)}\right)$, $\mathcal{L}_{\text{fr}} = -\sum_{q\in\mathcal{Q}^{\text{fr}}} \log\left(\frac{\phi_{\text{fr}}(c_{\text{ans}})}{\sum_{c}\phi_{\text{fr}}(c)}\right)$ and $\mathcal{L}_{\text{yu}} = -\sum_{q\in\mathcal{Q}^{\text{yu}}} \log\left(\frac{\phi_{\text{yu}}(x_{\text{ans}})}{\sum_{x\in\{yes,unknown\}}\phi_{\text{yu}}(x)}\right)$, respectively. $\mathcal{Q}^{\text{ep}}/\mathcal{Q}^{\text{yu}}/\mathcal{Q}^{\text{fr}}$ denotes all EPQs/YUQs/FRQs and $e_{\text{ans}}/x_{\text{ans}}/c_{\text{ans}}$ is the answer to question q.

5 Experiments

We answer several research questions (RQs) with experiments[7]. **RQ1** (Sect. 5.2, 5.4): Can a TKG forecasting model better support forecasting TKGQA than a TKGC model? **RQ2** (Sect. 5.2, 5.4): Does ForecastTKGQA perform well in forecasting TKGQA? **RQ3** (Sect. 5.3, 5.5): Are the questions in our dataset answerable? **RQ4** (Sect. 5.7): Is the proposed dataset efficient? **RQ5** (Sect. 5.6): What are the challenges of forecasting TKGQA?

[7] Implementation details and further analysis of ForecastTKGQA in Appendix B.3 and G.

5.1 Experimental Setting

Evaluation Metrics. We use mean reciprocal rank (MRR) and Hits@k as the evaluation metrics of the EPQs. For each EPQ, we compute the rank of the ground-truth answer entity among all the TKG entities. Test MRR is then computed as $\frac{1}{|\mathcal{Q}^{\text{ep}}_{\text{test}}|}\sum_{q\in\mathcal{Q}^{\text{ep}}_{\text{test}}}\frac{1}{\text{rank}_q}$, where $\mathcal{Q}^{\text{ep}}_{\text{test}}$ denotes all EPQs in the test set and rank_q is the rank of the ground-truth answer entity of question q. Hits@k is the proportion of the answered questions where the ground-truth answer entity is ranked as top k. For YUQs and FRQs, we employ accuracy for evaluation. Accuracy is the proportion of the correctly answered questions out of all questions.

Baseline Methods. We consider two pre-trained LMs, BERT [5] and RoBERTa [17] as baselines. For EPQs and YUQs, we add a prediction head on top of the question representations computed by LMs, and use softmax function to compute answer probabilities. For every FRQ, we input into each LM the concatenation of the question with each choice, and follow the same prediction structure. Besides, we derive two model variants for each LM by introducing TKG representations. We train TComplEx on ICEWS21. For every EPQ and YUQ, we concatenate the question representation with the TComplEx representations of the entities and timestamps annotated in the question, and then perform prediction with a prediction head and softmax. For FRQs, we further include TComplEx representations into choices in the same way. We call this type of variant BERT_int and RoBERTa_int since TComplEx is a TKGC (TKG interpolation) method. Similarly, we also introduce TANGO representations into LMs and derive BERT_ext and RoBERTa_ext, where TANGO serves as a TKG extrapolation backend. Detailed model derivations are presented in Appendix B.2. We also consider one KGQA method EmbedKGQA [22], and two TKGQA methods, i.e., CronKGQA [21] and TempoQR [19] as baselines. We run EmbedKGQA on top of the KG representations trained with ComplEx on ICEWS21, and run TKGQA baselines on top of the TKG representations trained with TComplEx.

5.2 Main Results

We report the experimental results in Table 5. In Table 5a, we show that our entity prediction model outperforms all baseline methods. We observe that EmbedKGQA achieves a better performance than BERT and RoBERTa, showing that employing KG representations helps TKGQA. Besides, LM variants outperform their original LMs, indicating that TKG representations help LMs perform better in TKGQA. Further, BERT_ext shows stronger performance than BERT_int (this also applies to RoBERTa_int and RoBERTa_ext), which proves that TKG forecasting models provide greater help than TKGC models in forecasting TKGQA. CronKGQA and TempoQR employ TComplEx representations as supporting information and perform poorly, implying that employing TKG representations provided by TKGC methods may include noisy information in forecasting TKGQA. ForecastTKGQA injects TANGO representations

Table 5. Experimental results over ForecastTKGQuestions. The best results are marked in bold.

(a) EPQs. Overall results in Appendix D.

	MRR		Hits@1		Hits@10	
Model	1-Hop	2-Hop	1-Hop	2-Hop	1-Hop	2-Hop
RoBERTa	0.166	0.149	0.104	0.085	0.288	0.268
BERT	0.279	0.182	0.192	0.106	0.451	0.342
EmbedKGQA	0.317	0.185	0.228	0.112	0.489	0.333
RoBERTa_int	0.283	0.157	0.190	0.094	0.467	0.290
BERT_int	0.314	0.183	0.223	0.107	0.490	0.344
CronKGQA	0.131	0.090	0.081	0.042	0.231	0.187
TempoQR	0.145	0.107	0.094	0.061	0.243	0.199
RoBERTa_ext	0.306	0.180	0.216	0.108	0.497	0.323
BERT_ext	0.331	0.208	0.239	0.128	0.508	0.369
ForecastTKGQA	**0.339**	**0.216**	**0.248**	**0.129**	**0.517**	**0.386**

(b) YUQs and FRQs.

	Accuracy	
Model	YUQ	FRQ
RoBERTa	0.721	0.645
BERT	0.813	0.634
RoBERTa_int	0.768	0.693
BERT_int	0.829	0.682
RoBERTa_ext	0.798	0.707
BERT_ext	0.837	0.746
ForecastTKGQA	**0.870**	**0.769**
Human Performance (a)	-	0.936
Human Performance (b)	-	0.954

into a scoring module, showing its great effectiveness on EPQs. For YUQs and FRQs, ForecastTKGQA also achieves the best performance. Table 5b shows that it is helpful to include TKG representations for answering YUQs and FRQs and our scoring functions are effective.

5.3 Human Vs. Machine on FRQs

To study the difference between humans and models in fact reasoning, we further benchmark human performance on FRQs with a survey (See Appendix E for details). We ask five graduate students to answer 100 questions randomly sampled from the test set. We consider two settings: (a) Humans answer FRQs with their own knowledge and inference ability. **Search engines are not allowed**; (b) Humans can turn to search engines and use the web information published **before the question timestamp** for aiding QA. Table 5 shows that humans achieve much stronger performance than all QA models (even in setting (a)). This calls for a great effort to build better fact reasoning TKGQA models.

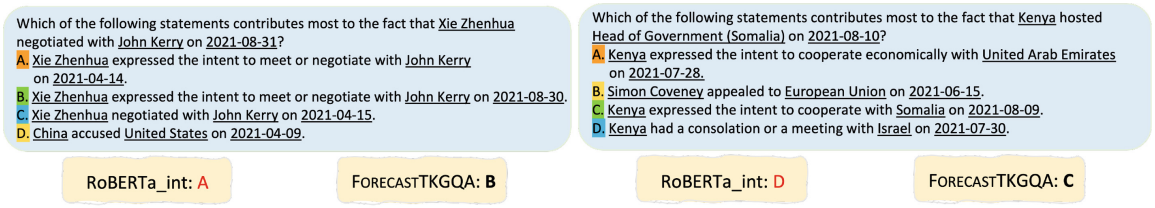
5.4 Performance over FRQs with Different Reasoning Types

Considering the reasoning types listed in Fig. 1, we compare RoBERTa_int with ForecastTKGQA on the 100 sampled test questions that are annotated with reasoning types, to justify performance gain brought by TKG forecasting model on FRQs. Experimental results in Table 6 imply that employing TKG forecasting model helps QA models better deal with any reasoning type on FRQs. We use two cases in Fig. 3 to provide insights into performance gain.

Case 1. Two reasoning skills, i.e., Causal Relation and Time Sensitivity (shown in Fig. 1), are required to correctly answer the question in Case 1. Without considering the timestamps of choices, A, B, C all seem at least somehow reasonable.

Table 6. Performance comparison across FRQs with different reasoning types.

Model	Accuracy		
	Causal Relation	Identity Understanding	Time Sensitivity
RoBERTa_int	0.670	0.529	0.444
ForecastTKGQA	**0.787**	**0.735**	**0.611**



(a) Case 1. (b) Case 2.

Fig. 3. Case Studies on FRQs. We mark green for **Answer**, blue for **Hard Negative**, orange for **Median** and yellow for **Negative**. (Color figure online)

However, after considering choice timestamps, B should be the most contributive reason for the question fact. First, the timestamp of B (*2021-08-30*) is much closer to the question timestamp (*2021-08-31*). Moreover, the fact in choice B directly causes the question fact. RoBERTa_int manages to capture the causation, but fails to correctly deal with time sensitivity, while ForecastTKGQA achieves better reasoning on both reasoning types.

Case 2. Two reasoning skills, i.e., Causal Relation and Identity Understanding (shown in Fig. 1), are required to correctly answer the question in Case 2. *Head of Government (Somalia)* and *Somalia* are two different entities in TKG, however, both entities are about Somalia. By understanding this, we are able to choose the correct answer. ForecastTKGQA manages to understand the identity of *Head of Government (Somalia)*, match it with *Somalia* and find the cause of the question fact. RoBERTa_int makes a mistake because as a model equipped with TComplEx, it has no well-trained timestamp representations of the question and choice timestamps, which would introduce noise in decision making.

5.5 Answerability of ForecastTKGQuestions

To validate the answerability of the questions in ForecastTKGQuestions. We train TComplEx and TANGO over the whole ICEWS21, i.e., $\mathcal{G}_{\text{train}} \cup \mathcal{G}_{\text{valid}} \cup \mathcal{G}_{\text{test}}$, and use them to support QA. Note that this violates the forecasting setting of forecasting TKGQA, and thus we call the TKG models trained in this way as cheating TComplEx (CTComplEx) and cheating TANGO (CTANGO). Answering EPQs with cheating TKG models is same as non-forecasting TKGQA. We couple TempoQR with CTComplEx and see a huge performance increase

Table 7. Answerability study. Models with α means using CTComplEx and β means using CTANGO. $\uparrow$ denotes relative improvement (%) from the results in Table 5. Acc means Accuracy.

(a) EPQs.

	MRR				Hits@10			
Model	1-Hop	$\uparrow$	2-Hop	$\uparrow$	1-Hop	$\uparrow$	2-Hop	$\uparrow$
TempoQR$^{\alpha}$	0.713	391.7	0.233	117.8	0.883	263.4	0.419	110.6
MHS$^{\alpha}$	0.868	-	0.647	-	0.992	-	0.904	-
MHS$^{\beta}$	0.771	-	0.556	-	0.961	-	0.828	-

(b) YUQs and FRQs.

	YUQ		FRQ	
Model	Acc	$\uparrow$	Acc	$\uparrow$
BERT_int$^{\alpha}$	0.855	19.6	0.816	14.4
BERT_ext$^{\beta}$	0.873	4.3	0.836	12.1
ForecastTKGQA$^{\beta}$	0.925	6.3	0.821	6.8

(Table 7a). Besides, inspired by [10], we develop a new TKGQA model Multi-Hop Scorer[8] (MHS) for EPQs. Starting from the annotated entity s_q of an EPQ, MHS updates the scores of outer entities for n-hops ($n = 2$ in our experiments) until all s_q's n-hop neighbors on the snapshot $\mathcal{G}_{t_q}$ are visited. Initially, MHS assigns a score of 1 to s_q and 0 to any other unvisited entity. For each unvisited entity e, it then computes e's score as: $\phi_{\text{ep}}(e) = \frac{1}{|\mathcal{N}_e(t_q)|}\sum_{(e',r)\in\mathcal{N}_e(t_q)}(\gamma \cdot \phi_{\text{ep}}(e') + \psi(e', r, e, t_q))$, where $\mathcal{N}_e(t_q) = \{(e', r)|(e', r, e, t_q) \in \mathcal{G}_{t_q}\}$ is e's 1-hop neighborhood on $\mathcal{G}_{t_q}$ and γ is a discount factor. We couple MHS with CTComplEx and CTANGO, and define $\psi(e', r, e, t_q)$ separately. For MHS + CTComplEx, $\psi(e', r, e, t_q) = f_2(f_1(\mathbf{h}_{e'} \| \mathbf{h}_r \| \mathbf{h}_e \| \mathbf{h}_{t_q} \| \mathbf{h}_q))$. f_1 and f_2 are two neural networks. $\mathbf{h}_e, \mathbf{h}_{e'}, \mathbf{h}_r, \mathbf{h}_{t_q}$ are the CTComplEx representations of entities e, e', relation r and timestamp t_q, respectively. For MHS + CTANGO, we take the idea of ForecastTKGQA: $\psi(e', r, e, t_q) = \text{Re}\left(< \mathbf{h}_{(e',t_q)}, \mathbf{h}_r, \bar{\mathbf{h}}_{(e,t_q)}, \mathbf{h}_q >\right)$. $\mathbf{h}_{(e,t_q)}$, $\mathbf{h}_{(e',t_q)}$, $\mathbf{h}_r$ are the CTANGO representations of entities e, e' at t_q, and relation r, respectively. $\mathbf{h}_q$ is BERT encoded question representation. We find that MHS achieves superior performance (even on 2-hop EPQs). This is because MHS not only uses cheating TKG models, but also considers ground-truth multi-hop structural information of TKGs at t_q (which is unavailable in the forecasting setting). For YUQs and FRQs, Table 7b shows that cheating TKG models help improve performance, especially on FRQs. These results imply that given the ground-truth TKG information at question timestamps, our forecasting TKGQA questions are answerable.

5.6 Challenges of Forecasting TKGQA over Forecast TKG Questions

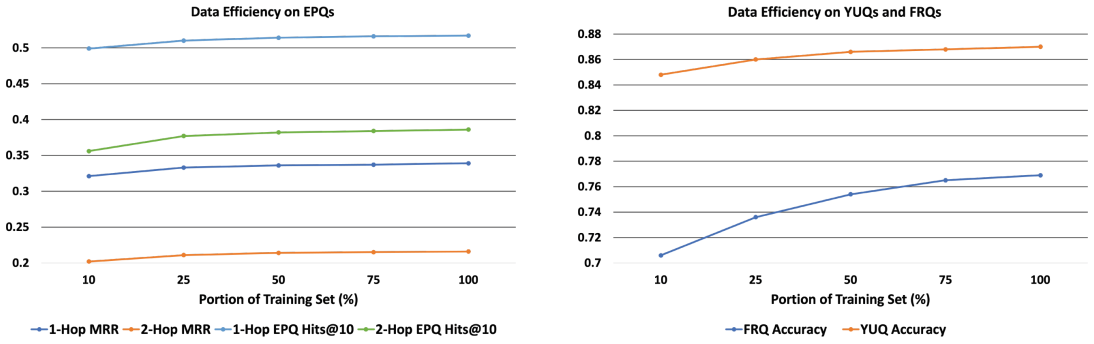
From the experiments discussed in Sect. 5.3 and 5.5, we summarize the challenges of forecasting TKGQA: (1) Inferring the ground-truth TKG information $\mathcal{G}_{t_q}$ at the question timestamp t_q accurately; (2) Effectively performing multi-hop reasoning for forecasting TKGQA; (3) Developing TKGQA models for better fact reasoning. In Sect. 5.5, we have trained cheating TKG models and used them to support QA. We show in Table 7 that QA models substantially improve their performance on forecasting TKGQA with cheating TKG models. This implies

[8] See Appendix F for detailed model explanation and model structure illustration.

that accurately inferring the ground-truth TKG information at t_q is crucial in our task and how to optimally achieve it remains a challenge. We also observe that MHS with cheating TKG models achieves much better results on EPQs (especially on 2-hop). MHS utilizes multi-hop information of the ground-truth TKG at t_q ($\mathcal{G}_{t_q}$) for better QA. In forecasting TKGQA, by only knowing the TKG facts before t_q and not observing $\mathcal{G}_{t_q}$, it is impossible for MHS to directly utilize the ground-truth multi-hop information at t_q. This implies that how to effectively infer and exploit multi-hop information for QA in the forecasting scenario remains a challenge. Moreover, as discussed in Sect. 5.3, current TKGQA models still trail humans with great margin on FRQs. It is challenging to design novel forecasting TKGQA models for better fact reasoning.

5.7 Study of Data Efficiency

We want to know how the models will be affected with less/more training data. For each type of questions, we modify the size of its training set. We train ForecastTKGQA on the modified training sets and evaluate our model on the original test sets. We randomly sample 10%, 25%, 50%, and 75% of the training examples to form new training sets. Figure 4 shows that for every type of question, the performance of ForecastTKGQA steadily improves as the size of the training sets increase. This proves that our proposed dataset is efficient and useful for training forecasting TKGQA models.



(a) Data efficiency on EPQs. (b) Data efficiency on YUQs, FRQs.

Fig. 4. Data efficiency analysis.

6 Justification of Task Validity from Two Perspectives

(1) Perspective from Underlying TKG. We take a commonly used temporal KB, i.e., ICEWS, as the KB for constructing underlying TKG ICEWS21. ICEWS-based TKGs contain socio-political facts. It is meaningful to perform forecasting over them because this can help to improve early warning in critical socio-political situations around the globe. [25] has shown with case studies that ICEWS-based TKG datasets have underlying cause-and-effect temporal

patterns and TKG forecasting models are built to capture them. This indicates that performing TKG forecasting over ICEWS-based TKGs is also valid. And therefore, developing forecasting TKGQA on top of ICEWS21 is meaningful and valid. **(2) Perspective from the Motivation of Proposing Different Types of Questions.** The motivation of proposing EPQs is to introduce TKG link forecasting (future link prediction) into KGQA, while proposing YUQs is to introduce quadruple classification (stemming from triple classification) and yes-no type questions. We view quadruple classification in the forecasting scenario as deciding if the unseen TKG facts are valid based on previously known TKG facts. To answer EPQs and YUQs, models can be considered as understanding natural language questions first and then performing TKG reasoning tasks. Since TKG reasoning tasks are considered solvable and widely studied in the TKG community, our task over EPQs and YUQs is valid. We propose FRQs aiming to study the difference between humans and machines in fact reasoning. We have summarized the reasoning skills that are required to answer every FRQ in Fig. 1, which also implies the potential direction for QA models to achieve improvement in fact reasoning in the future. We have shown in Sect. 5.3 that our proposed FRQs are answerable to humans, which directly indicates the validity of our FRQs. Thus, answering FRQs in forecasting TKGQA is also valid and meaningful.

7 Conclusion

In this work, we propose a novel task: forecasting TKGQA. To the best of our knowledge, it is the first work combining TKG forecasting with KGQA. We propose a coupled benchmark dataset ForecastTKGQuestions that contains various types of questions including EPQs, YUQs and FRQs. To solve forecasting TKGQA, we propose ForecastTKGQA, a QA model that leverages a TKG forecasting model with a pre-trained LM. Though experimental results show that our model achieves great performance, there still exists a large room for improvement compared with humans. We hope our work can benefit future research and draw attention to studying the forecasting power of TKGQA methods.

Supplemental Material Statement: Source code and data are uploaded here[9]. Appendices are published in the arXiv version[10]. We have referred to the corresponding parts in the main body. Please check accordingly.

Acknowledgement. This work has been supported by the German Federal Ministry for Economic Affairs and Climate Action (BMWK) as part of the project CoyPu under grant number 01MK21007K.

[9] https://github.com/ZifengDing/ForecastTKGQA.
[10] https://arxiv.org/abs/2208.06501.

References

1. Bordes, A., Usunier, N., Chopra, S., Weston, J.: Large-scale simple question answering with memory networks (2015). arxiv.org:1506.02075
2. Boschee, E., Lautenschlager, J., O'Brien, S., Shellman, S., Starz, J., Ward, M.: ICEWS Coded Event Data (2015). https://doi.org/10.7910/DVN/28075
3. Cao, Y., Ji, X., Lv, X., Li, J., Wen, Y., Zhang, H.: Are missing links predictable? an inferential benchmark for knowledge graph completion. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1–6, 2021, pp. 6855–6865. Association for Computational Linguistics (2021). https://doi.org/10.18653/v1/2021.acl-long.534
4. Chen, Z., Zhao, X., Liao, J., Li, X., Kanoulas, E.: Temporal knowledge graph question answering via subgraph reasoning. Knowl. Based Syst. **251**, 109134 (2022). https://doi.org/10.1016/j.knosys.2022.109134
5. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2–7, 2019, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/n19-1423
6. Ding, Z., Ma, Y., He, B., Han, Z., Tresp, V.: A simple but powerful graph encoder for temporal knowledge graph completion. In: NeurIPS 2022 Temporal Graph Learning Workshop (2022). https://openreview.net/forum?id=DYG8RbgAIo
7. Galárraga, L.A., Teflioudi, C., Hose, K., Suchanek, F.M.: AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In: Schwabe, D., Almeida, V.A.F., Glaser, H., Baeza-Yates, R., Moon, S.B. (eds.) 22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13–17, 2013, pp. 413–422. International World Wide Web Conferences Steering Committee / ACM (2013). https://doi.org/10.1145/2488388.2488425
8. Han, Z., Chen, P., Ma, Y., Tresp, V.: Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3–7, 2021. OpenReview.net (2021). https://openreview.net/forum?id=pGIHq1m7PU
9. Han, Z., Ding, Z., Ma, Y., Gu, Y., Tresp, V.: Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In: Moens, M., Huang, X., Specia, L., Yih, S.W. (eds.) Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7–11 November, 2021, pp. 8352–8364. Association for Computational Linguistics (2021). https://doi.org/10.18653/v1/2021.emnlp-main.658
10. Ji, H., Ke, P., Huang, S., Wei, F., Zhu, X., Huang, M.: Language generation with multi-hop reasoning on commonsense knowledge graph. In: Webber, B., Cohn, T., He, Y., Liu, Y. (eds.) Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16–20, 2020, pp. 725–736. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.emnlp-main.54

11. Jia, Z., Abujabal, A., Roy, R.S., Strötgen, J., Weikum, G.: Tempquestions: A benchmark for temporal question answering. In: Champin, P., Gandon, F., Lalmas, M., Ipeirotis, P.G. (eds.) Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon, France, April 23–27, 2018, pp. 1057–1062. ACM (2018). https://doi.org/10.1145/3184558.3191536
12. Jia, Z., Pramanik, S., Roy, R.S., Weikum, G.: Complex temporal question answering on knowledge graphs. In: Demartini, G., Zuccon, G., Culpepper, J.S., Huang, Z., Tong, H. (eds.) CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1–5, 2021, pp. 792–802. ACM (2021). https://doi.org/10.1145/3459637.3482416
13. Jin, W., et al.: Forecastqa: A question answering challenge for event forecasting with temporal text data. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1–6, 2021, pp. 4636–4650. Association for Computational Linguistics (2021). https://doi.org/10.18653/v1/2021.acl-long.357
14. Jin, W., Qu, M., Jin, X., Ren, X.: Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In: Webber, B., Cohn, T., He, Y., Liu, Y. (eds.) Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16–20, 2020, pp. 6669–6683. Association for Computational Linguistics (2020). https://doi.org/10.18653/v1/2020.emnlp-main.541
15. Jung, J., Jung, J., Kang, U.: Learning to walk across time for interpretable temporal knowledge graph completion. In: Zhu, F., Ooi, B.C., Miao, C. (eds.) KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14–18, 2021, pp. 786–795. ACM (2021). https://doi.org/10.1145/3447548.3467292
16. Lacroix, T., Obozinski, G., Usunier, N.: Tensor decompositions for temporal knowledge base completion. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26–30, 2020. OpenReview.net (2020), https://openreview.net/forum?id=rke2P1BFwS
17. Liu, Y., et al.: Roberta: A robustly optimized BERT pretraining approach (2019). https://doi.org/10.48550/ARXIV.1907.11692
18. Liu, Y., Ma, Y., Hildebrandt, M., Joblin, M., Tresp, V.: Tlogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. In: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022, pp. 4120–4127. AAAI Press (2022). https://ojs.aaai.org/index.php/AAAI/article/view/20330
19. Mavromatis, C., et al.: Tempoqr: Temporal question reasoning over knowledge graphs. In: Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022, pp. 5825–5833. AAAI Press (2022). https://ojs.aaai.org/index.php/AAAI/article/view/20526
20. Meilicke, C., Chekol, M.W., Fink, M., Stuckenschmidt, H.: Reinforced anytime bottom up rule learning for knowledge graph completion (2020). arxiv.org:2004.04412

21. Saxena, A., Chakrabarti, S., Talukdar, P.P.: Question answering over temporal knowledge graphs. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1–6, 2021, pp. 6663–6676. Association for Computational Linguistics (2021). https://doi.org/10.18653/v1/2021.acl-long.520
22. Saxena, A., Tripathi, A., Talukdar, P.: Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 4498–4507. Association for Computational Linguistics, Online (2020). https://doi.org/10.18653/v1/2020.acl-main.412, https://aclanthology.org/2020.acl-main.412
23. Shang, C., Wang, G., Qi, P., Huang, J.: Improving time sensitivity for question answering over temporal knowledge graphs. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22–27, 2022, pp. 8017–8026. Association for Computational Linguistics (2022). https://aclanthology.org/2022.acl-long.552
24. Talmor, A., Berant, J.: The web as a knowledge-base for answering complex questions. In: Walker, M.A., Ji, H., Stent, A. (eds.) Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1–6, 2018, Volume 1 (Long Papers), pp. 641–651. Association for Computational Linguistics (2018). https://doi.org/10.18653/v1/n18-1059
25. Trivedi, R., Dai, H., Wang, Y., Song, L.: Know-evolve: Deep temporal reasoning for dynamic knowledge graphs. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6–11 August 2017. Proceedings of Machine Learning Research, vol. 70, pp. 3462–3471. PMLR (2017). http://proceedings.mlr.press/v70/trivedi17a.html
26. Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., Bouchard, G.: Complex embeddings for simple link prediction. In: Balcan, M., Weinberger, K.Q. (eds.) Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19–24, 2016. JMLR Workshop and Conference Proceedings, vol. 48, pp. 2071–2080. JMLR.org (2016), http://proceedings.mlr.press/v48/trouillon16.html
27. Vrandecic, D., Krötzsch, M.: Wikidata: a free collaborative knowledgebase. Commun. ACM **57**(10), 78–85 (2014). https://doi.org/10.1145/2629489
28. Yih, W., Chang, M., He, X., Gao, J.: Semantic parsing via staged query graph generation: Question answering with knowledge base. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26–31, 2015, Beijing, China, Volume 1: Long Papers, pp. 1321–1331. The Association for Computer Linguistics (2015). https://doi.org/10.3115/v1/p15-1128
29. Zhang, Y., Dai, H., Kozareva, Z., Smola, A.J., Song, L.: Variational reasoning for question answering with knowledge graph. In: McIlraith, S.A., Weinberger, K.Q. (eds.) Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2–7, 2018, pp. 6069–6076.

AAAI Press (2018). https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16983

30. Zhu, C., Chen, M., Fan, C., Cheng, G., Zhang, Y.: Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In: Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2–9, 2021, pp. 4732–4740. AAAI Press (2021). https://ojs.aaai.org/index.php/AAAI/article/view/16604

A Scoring Function Details

A.1 Entity Prediction

The detailed definition of the EPQs' scoring function is defined as

$$
\begin{aligned}
\phi_{\mathrm{ep}}(e) &= \mathrm{Re}\left(< \mathbf{h}'_{(s_q,t_q)}, \mathbf{h}_q, \bar{\mathbf{h}}'_{(e,t_q)} >\right) \\
&= \mathrm{Re}\left(\sum_{k=1}^{d} \mathbf{h}'_{(s_q,t_q)}(k) \cdot \mathbf{h}_q(k) \cdot \bar{\mathbf{h}}'_{(e,t_q)}(k)\right) \\
&=< \mathrm{Re}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Re}\left(\mathbf{h}_q\right), \mathrm{Re}\left(\mathbf{h}'_{(e,t_q)}\right) > \\
&\quad + < \mathrm{Re}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Im}\left(\mathbf{h}_q\right), \mathrm{Im}\left(\mathbf{h}'_{(e,t_q)}\right) > \\
&\quad + < \mathrm{Im}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Re}\left(\mathbf{h}_q\right), \mathrm{Im}\left(\mathbf{h}'_{(e,t_q)}\right) > \\
&\quad - < \mathrm{Im}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Im}\left(\mathbf{h}_q\right), \mathrm{Re}\left(\mathbf{h}'_{(e,t_q)}\right) > .
\end{aligned} \tag{1}
$$

Re and Im denote taking the real part and the imaginary part of the complex vector, respectively. $\mathbf{h}'_{(s_q,t_q)}, \mathbf{h}_q, \mathbf{h}'_{(e,t_q)} \in \mathbb{C}^d$. $\mathbf{h}'_{(s_q,t_q)}(k)$ denotes the kth element of it (same for $\mathbf{h}_q$ and $\mathbf{h}'_{(e,t_q)}$). $< \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 >= \sum_{k=1}^{d} \mathbf{v}_1(k) \cdot \mathbf{v}_2(k) \cdot \mathbf{v}_3(k)$ denotes the dot product of three d-dimensional complex vectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3 \in \mathbb{C}^d$.

A.2 Yes-Unknown

The detailed definition of the YUQs' scoring function is defined as

$$
\begin{aligned}
\phi_{\mathrm{yu}}(x) &= \mathrm{Re}\left(< \mathbf{h}'_{(s_q,t_q)}, \mathbf{h}_q, \bar{\mathbf{h}}'_{(o_q,t_q)}, \mathbf{h}_x >\right) \\
&= \mathrm{Re}\left(\sum_{k=1}^{d} \mathbf{h}'_{(s_q,t_q)}(k) \cdot \mathbf{h}_q(k) \cdot \bar{\mathbf{h}}'_{(o_q,t_q)}(k) \cdot \mathbf{h}_x(k)\right) \\
&=< \mathrm{Re}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Re}\left(\mathbf{h}_q\right), \mathrm{Re}\left(\mathbf{h}'_{(o_q,t_q)}\right), \mathrm{Re}\left(\mathbf{h}_x\right) > \\
&\quad + < \mathrm{Re}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Im}\left(\mathbf{h}_q\right), \mathrm{Im}\left(\mathbf{h}'_{(o_q,t_q)}\right), \mathrm{Re}\left(\mathbf{h}_x\right) > \\
&\quad + < \mathrm{Im}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Re}\left(\mathbf{h}_q\right), \mathrm{Im}\left(\mathbf{h}'_{(o_q,t_q)}\right), \mathrm{Re}\left(\mathbf{h}_x\right) > \\
&\quad + < \mathrm{Re}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Re}\left(\mathbf{h}_q\right), \mathrm{Im}\left(\mathbf{h}'_{(o_q,t_q)}\right), \mathrm{Im}\left(\mathbf{h}_x\right) > \\
&\quad - < \mathrm{Im}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Im}\left(\mathbf{h}_q\right), \mathrm{Re}\left(\mathbf{h}'_{(o_q,t_q)}\right), \mathrm{Re}\left(\mathbf{h}_x\right) > \\
&\quad - < \mathrm{Im}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Re}\left(\mathbf{h}_q\right), \mathrm{Re}\left(\mathbf{h}'_{(o_q,t_q)}\right), \mathrm{Im}\left(\mathbf{h}_x\right) > \\
&\quad - < \mathrm{Im}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Im}\left(\mathbf{h}_q\right), \mathrm{Im}\left(\mathbf{h}'_{(o_q,t_q)}\right), \mathrm{Im}\left(\mathbf{h}_x\right) > \\
&\quad - < \mathrm{Re}\left(\mathbf{h}'_{(s_q,t_q)}\right), \mathrm{Im}\left(\mathbf{h}_q\right), \mathrm{Re}\left(\mathbf{h}'_{(o_q,t_q)}\right), \mathrm{Im}\left(\mathbf{h}_x\right) > .
\end{aligned} \tag{2}
$$

A.3 Fact Reasoning

The detailed definition of FRQs' scoring function is defined as

$$
\begin{aligned}
\phi_{\mathrm{fr}}(c) &= \mathrm{Re}\left(< \mathbf{h}'_{(s_c,t_c)}, \mathbf{h}^c_q, \bar{\mathbf{h}}'_{(o_c,t_c)}, \mathbf{h}'_q >\right) \\
&= \mathrm{Re}\left(\sum_{k=1}^{d} \mathbf{h}'_{(s_c,t_c)}(k) \cdot \mathbf{h}^c_q(k) \cdot \bar{\mathbf{h}}'_{(o_c,t_c)}(k) \cdot \mathbf{h}'_q(k)\right) \\
&=< \mathrm{Re}\left(\mathbf{h}'_{(s_c,t_c)}\right), \mathrm{Re}\left(\mathbf{h}^c_q\right), \mathrm{Re}\left(\mathbf{h}'_{(o_c,t_c)}\right), \mathrm{Re}\left(\mathbf{h}'_q(k)\right) > \\
&\quad + < \mathrm{Re}\left(\mathbf{h}'_{(s_c,t_c)}\right), \mathrm{Im}\left(\mathbf{h}^c_q\right), \mathrm{Im}\left(\mathbf{h}'_{(o_c,t_c)}\right), \mathrm{Re}\left(\mathbf{h}'_q(k)\right) > \\
&\quad + < \mathrm{Im}\left(\mathbf{h}'_{(s_c,t_c)}\right), \mathrm{Re}\left(\mathbf{h}^c_q\right), \mathrm{Im}\left(\mathbf{h}'_{(o_c,t_c)}\right), \mathrm{Re}\left(\mathbf{h}'_q(k)\right) > \\
&\quad + < \mathrm{Re}\left(\mathbf{h}'_{(s_c,t_c)}\right), \mathrm{Re}\left(\mathbf{h}^c_q\right), \mathrm{Im}\left(\mathbf{h}'_{(o_c,t_c)}\right), \mathrm{Im}\left(\mathbf{h}'_q(k)\right) > \\
&\quad - < \mathrm{Im}\left(\mathbf{h}'_{(s_c,t_c)}\right), \mathrm{Im}\left(\mathbf{h}^c_q\right), \mathrm{Re}\left(\mathbf{h}'_{(o_c,t_c)}\right), \mathrm{Re}\left(\mathbf{h}'_q(k)\right) > \\
&\quad - < \mathrm{Im}\left(\mathbf{h}'_{(s_c,t_c)}\right), \mathrm{Re}\left(\mathbf{h}^c_q\right), \mathrm{Re}\left(\mathbf{h}'_{(o_c,t_c)}\right), \mathrm{Im}\left(\mathbf{h}'_q(k)\right) > \\
&\quad - < \mathrm{Im}\left(\mathbf{h}'_{(s_c,t_c)}\right), \mathrm{Im}\left(\mathbf{h}^c_q\right), \mathrm{Im}\left(\mathbf{h}'_{(o_c,t_c)}\right), \mathrm{Im}\left(\mathbf{h}'_q(k)\right) > \\
&\quad - < \mathrm{Re}\left(\mathbf{h}'_{(s_c,t_c)}\right), \mathrm{Im}\left(\mathbf{h}^c_q\right), \mathrm{Re}\left(\mathbf{h}'_{(o_c,t_c)}\right), \mathrm{Im}\left(\mathbf{h}'_q(k)\right) > .
\end{aligned} \tag{3}
$$

B Implementation Details

We implement all the experiments with PyTorch [6] on an NVIDIA A40 with 48GB memory and a 2.6GHZ AMD EPYC 7513 32-Core Processor.

B.1 TKG Forecasting

We train TANGO and TComplEx to perform TKG forecasting on ICEWS21. We implement TANGO with the official implementation[1]. We switch its scoring function to ComplEx and perform a grid search for the embedding size (the dimension d of the entity and relation representations). We keep the rest hyperparameters as TANGO's default setting of the ICEWS05-15 dataset. We train TComplEx with the official implementation[2]. We perform a grid search for the embedding size and keep the other hyperparameters as their default values. Table 1 provides the searching spaces of the grid searches for both methods. For each method, we run TKG forecasting experiments with different embedding sizes and choose the setting that leads to the best validation MRR as the best hyperparameter setting. We further run TANGO + TuckER with the best hyperparameters searched with TANGO + ComplEx for studying the effectiveness of different KG representations.

[1] https://github.com/TemporalKGTeam/TANGO
[2] https://github.com/facebookresearch/tkbc

Table 1: Embedding size search space of TANGO and TComplEx. The embedding sizes leading to the best validation results are marked as bold. Note that the numbers represent the dimensions of complex space. Dimensions of real valued vectors are doubled, e.g., a complex vector with embedding size 100 will be transformed into a real valued vector with embedding size 200. The embedding size search spaces are taken from the default search space stated in the original papers of TANGO and TComplEx.

	Emebdding Size Search Space
TANGO	{50, **100**, 150}
TComplEx	{**100**, 136, 174}

Besides, we train ComplEx on ICEWS21 for TKG forecasting. We use the implementation provided in the repository of TComplEx. Since ComplEx is not designed for processing temporal information, we transform every quadruple (s, r, o, t) into a corresponding triplet (s, r, o). We do not remove the repeated triplet. For example, if (s, r, o, t_1) and (s, r, o, t_2) both exist in the training set of ICEWS21, we train ComplEx with two identical triplets (s, r, o). This preserves the inductive bias brought by the temporal knowledge base. To achieve a fairer comparison between ComplEx and TComplEx, we set the embedding size of ComplEx to 100 (same as the embedding size of TComplEx).

We report in Table 2 the validation results of the trained TKG models of all three KG reasoning methods on ICEWS21. We observe that TComplEx underperforms ComplEx in TKG forecasting. We attribute this to the excessive noise introduced by TComplEx's representations of unseen timestamps. Note that TComplEx is a TKG completion method. The validation timestamps are unseen during training, thus causing TComplEx to leverage the untrained timestamp representations during evaluation. ComplEx does not consider temporal information, which enables it to avoid the negative influence of the timestamps unseen in the training set. TANGO is designed for TKG forecasting. It outperforms the other methods greatly. Although TANGO + TuckER performs better than TANGO + ComplEx on ICEWS21, we choose the latter one for forecasting TKGQA since it aligns to our QA scoring function better (see Appendix G for detailed discussion).

Table 2: Validation results of KG reasoning models for TKG forecasting on ICEWS21.

Metrics	**MRR**	**Hits@1**	**Hits@3**	**Hits@10**
ComplEx	0.278	0.188	0.312	0.456
TComplEx	0.250	0.164	0.279	0.420
TANGO + TuckER	0.402	0.327	0.431	0.546
TANGO + ComplEx	0.389	0.324	0.411	0.515

B.2 Baseline Details

We use the library HuggingFace's Transformers [9] to implement the pre-trained LMs, i.e., BERT and RoBERTa. Following CronKGQA and TempoQR, we choose DistilBERT [7] as the BERT model used throughout our work to save computational budget. For every natural language input, e.g., a natural language question, we take the output representation of the [CLS] token computed by an LM as its LM encoded representation.

Pre-trained LM baselines for TKGQA We provide detailed information of our pre-trained LM baselines. For EPQs, BERT and RoBERTa compute the scores of all entities with a prediction head $f_{\text{ep}}^{\text{lm}} : \mathbb{R}^{2d} \rightarrow \mathbb{R}^{|\mathcal{E}|}$ as

$$\boldsymbol{\Phi}_{\text{ep}} = f_{\text{ep}}^{\text{lm}}\left(\mathbf{h}_q\right). \tag{4}$$

$\boldsymbol{\Phi}_{\text{ep}}$ is a $|\mathcal{E}|$-dimensional real valued vector where each element corresponds to the score of an entity. $\mathbf{h}_q$ is the question representation output by BERT or RoBERTa with a projection to a $2d$ real space. Note that in ForecastTKGQA, we further map the $2d$ real valued vector to a d-dimensional complex vector. This step does not exist when we implement pre-trained LM baselines without including any TKG representation. We choose the entity with the highest score as the predicted answer. BERT_int and RoBERTa_int compute the score of each entity e with a prediction head $f_{\text{ep}}^{\text{lm\_int}} : \mathbb{R}^{8d} \rightarrow \mathbb{R}^1$ as

$$\phi_{\text{ep}}(e) = f_{\text{ep}}^{\text{lm\_int}}\left(\mathbf{h}_s \| \mathbf{h}_q \| \mathbf{h}_e \| \mathbf{h}_{t_q}\right), \tag{5}$$

where $\mathbf{h}_s$, $\mathbf{h}_{t_q}$, and $\mathbf{h}_o$ denote the TComplEx representations of the question's subject entity, the question's timestamp, and the entity e, respectively. Similarly, BERT_ext and RoBERTa_ext compute the score of each entity e with a prediction head $f_{\text{ep}}^{\text{lm\_ext}} : \mathbb{R}^{6d} \rightarrow \mathbb{R}^1$ as

$$\phi_{\text{ep}}(e) = f_{\text{ep}}^{\text{lm\_ext}}(\mathbf{h}_{(s_q,t_q)} \| \mathbf{h}_q \| \mathbf{h}_{(e,t_q)}), \tag{6}$$

where $\mathbf{h}_{(s_q,t_q)}$ and $\mathbf{h}_{(e,t_q)}$ denote the TANGO representations of the question's subject entity and the entity e, respectively. Since TANGO and TComplEx representations are complex vectors in $\mathbb{C}^d$, we expand them into $2d$ real valued vectors, where the first half of every real valued vector is the real part of the original vector and the second half is the imaginary part. This applies to all the TKG representations used in pre-trained LM baselines for answering all three types of questions.

For yes-unknown questions, BERT and RoBERTa compute the scores of *yes* and *unknown* with a prediction head $f_{\text{yu}}^{\text{lm}} : \mathbb{R}^{2d} \rightarrow \mathbb{R}^2$ as

$$\boldsymbol{\Phi}_{\text{yu}} = f_{\text{yu}}^{\text{lm}}(\mathbf{h}_q). \tag{7}$$

$\boldsymbol{\Phi}_{\text{yu}}$ is a 2-dimensional real valued vector where each element corresponds to the score of either *yes* or *unknown*. BERT_int and RoBERTa_int compute the score of each $x \in \{yes, unknown\}$ with a prediction head $f_{\text{yn}}^{\text{lm\_int}} : \mathbb{R}^{8d} \rightarrow \mathbb{R}^1$ as

$$\phi_{\text{yu}}(x) = f_{\text{yn}}^{\text{lm\_int}}\left(\mathbf{h}_{s_q} \| \mathbf{h}_q \| \mathbf{h}_{o_q} \| \mathbf{h}_{t_q}\right). \tag{8}$$

And BERT_ext and RoBERTa_ext compute the score of each $x \in \{yes, unknown\}$ with a prediction head $f_{\text{yu}}^{\text{lm\_ext}} : \mathbb{R}^{6d} \rightarrow \mathbb{R}^1$ as

$$\phi_{\text{yu}}(x) = f_{\text{yu}}^{\text{lm\_ext}}\left(\mathbf{h}_{(s_q,t_q)} \| \mathbf{h}_q \| \mathbf{h}_{(o_q,t_q)}\right). \tag{9}$$

We choose the one (either *yes* or *unknown*) with the higher score as the predicted answer.

For every fact reasoning question, BERT and RoBERTa compute the score of the choice c as

$$\phi_{\text{fr}}(c) = f_{\text{fr}}^{\text{lm}}(\mathbf{h}_q^c). \tag{10}$$

$\mathbf{h}_q^c$ is the output of a pre-trained LM when the concatenation of the question q and the choice c is given as the input. $f_{\text{fr}}^{\text{lm}} : \mathbb{R}^{2d} \rightarrow \mathbb{R}^1$ is a layer of neural network for score computation. BERT_int and RoBERTa_int compute the score of the choice c as

$$\phi_{\text{fr}}(c) = f_{\text{fr}}^{\text{lm\_int}}\left(\mathbf{h}_q^{\text{lm\_int}} \| \mathbf{h}_c^{\text{lm\_int}}\right). \tag{11}$$

$\mathbf{h}_q^{\text{lm\_int}} = \mathbf{h}_{s_q} \| \mathbf{h}_q^c \| \mathbf{h}_{o_q} \| \mathbf{h}_{t_q}$, where $\mathbf{h}_{s_q}$, $\mathbf{h}_{o_q}$ and $\mathbf{h}_{t_q}$ denote the TComplEx representations of the question's subject entity, object entity and timestamp, respectively. $\mathbf{h}_c^{\text{lm\_int}} = \mathbf{h}_{s_c} \| \mathbf{h}_q^c \| \mathbf{h}_{o_c} \| \mathbf{h}_{t_c}$, where $\mathbf{h}_{s_c}$, $\mathbf{h}_{o_c}$ and $\mathbf{h}_{t_c}$ denote the TComplEx representations of the choice's subject entity, object entity and timestamp, respectively. $f_{\text{fr}}^{\text{lm\_int}} : \mathbb{R}^{16d} \rightarrow \mathbb{R}^1$ is a layer of neural network for score computation. Similarly, BERT_ext and RoBERTa_ext compute the score of the choice c as

$$\phi_{\text{fr}}(c) = f_{\text{fr}}^{\text{lm\_ext}}(\mathbf{h}_q^{\text{lm\_ext}} \| \mathbf{h}_c^{\text{lm\_ext}}). \tag{12}$$

$\mathbf{h}_q^{\text{lm\_ext}} = \mathbf{h}_{(s_q,t_q)} \| \mathbf{h}_q^c \| \mathbf{h}_{(o_q,t_q)}$, where $\mathbf{h}_{(s_q,t_q)}$ and $\mathbf{h}_{(o_q,t_q)}$ denote the time-aware TANGO representations of the question's subject entity and object entity, respectively. $\mathbf{h}_c^{\text{lm\_ext}} = \mathbf{h}_{(s_c,t_c)} \| \mathbf{h}_q^c \| \mathbf{h}_{(o_c,t_c)}$, where $\mathbf{h}_{(s_c,t_c)}$ and $\mathbf{h}_{(o_c,t_c)}$ denote the time-aware TANGO representations of the choice's subject entity and object entity, respectively. $f_{\text{fr}}^{\text{lm\_ext}} : \mathbb{R}^{12d} \rightarrow \mathbb{R}^1$ is a layer of neural network for score computation.

KGQA & TKGQA Baselines For EmbedKGQA, we use the trained ComplEx representations as its supporting KG information. For CronKGQA and TempoQR, we use the trained TComplEx representations as their supporting TKG information. We use the EmbedKGQA and CronKGQA implementation provided in the repository of CronKGQA[3]. We use the official implementation of TempoQR[4]. Since we annotate the timestamps for every entity prediction question in ForecastTKGQuestions, we do not implement soft/hard supervision proposed in TempoQR. We skip the soft/hard supervision and keep everything else as same as the original implementation. We implement all the KGQA baselines with their default hyperparameter settings.

[3] https://github.com/apoorvumang/CronKGQA
[4] https://github.com/cmavro/TempoQR

Table 3: ForecastTKGQA hyperparameter searching strategy.

Hyperparameter	Search Space
TKG Model	{TuckER, ComplEx}
Language Model	{DistilBERT, RoBERTa}
Dropout	{0.2, 0.3, 0.5}
Batch Size	{32, 64, 128, 256, 512}

Table 4: Best hyperparameter setting.

Question Type	**Entity Prediction**	**Yes-Unknown**	**Fact Reasoning**
Hyperparameter			
TKG Model	ComplEx	ComplEx	ComplEx
Language Model	DistilBERT	DistilBERT	DistilBERT
Dropout	0.3	0.3	0.3
Batch Size	512	256	256

Table 5: Experimental results of EPQs on the validation set. Evaluation metrics are MRR and Hits@1/10.

	MRR			**Hits@1**			**Hits@10**		
Model	Overall	1-Hop	2-Hop	Overall	1-Hop	2-Hop	Overall	1-Hop	2-Hop
ForecastTKGQA	0.297	0.342	0.192	0.206	0.247	0.111	0.475	0.526	0.353

B.3 ForecastTKGQA

We search hyperparameters of ForecastTKGQA following Table 3. For every type of question, we do 60 trials, and let our model run for 50 epochs. We select the trial leading to the best performance on the validation set and take this hyperparameter setting as our best configuration. We train our model five times with different random seeds and report averaged results. The best hyperparameters concerning all three types of questions are shown in Table 4. We also report the model performance on the validation sets in Table 5 and Table 7. We further report the standard deviation of the results on the test sets in Table 6 and Table 8. The GPU memory usage is reported in Table 9. The training time and test time of our model are presented in Table 10 and Table 11. The number of parameters of our model is presented in Table 12.

Table 6: Standard deviation of the results of EPQs on the test set.

	MRR			Hits@1			Hits@10		
Model	Overall	1-Hop	2-Hop	Overall	1-Hop	2-Hop	Overall	1-Hop	2-Hop
ForecastTKGQA	0.0004	0.0004	0.0009	0.0006	0.0007	0.0007	0.0008	0.0008	0.0018

Table 7: Experimental results of YUQs and FRQs on the validation sets. The evaluation metric is accuracy.

	Accuracy	
Question Type	Yes-Unknown	Fact Reasoning
ForecastTKGQA	0.873	0.758

Table 8: Standard deviation of the results of YUQs and FRQs on the test set.

	Accuracy	
Question Type	Yes-Unknown	Fact Reasoning
ForecastTKGQA	0.0013	0.0052

Table 9: GPU memory usage.

Question Type	**Entity Prediction**	**Yes-Unknown**	**Fact Reasoning**
Model	GPU Memory	GPU Memory	GPU Memory
ForecastTKGQA	45,239MB	12,241MB	22,719MB

Table 10: Training time (second) of ForecastTKGQA on all types of questions.

Question Type	**Entity Prediction**	**Yes-Unknown**	**Fact Reasoning**
Model			
ForecastTKGQA	63,840	3,700	5000

Table 11: Test time (second) of ForecastTKGQA on all types of questions.

Question Type	**Entity Prediction**	**Yes-Unknown**	**Fact Reasoning**
Model			
ForecastTKGQA	48	33	3

C ForecastTKGQuestions Details

C.1 Natural Language Relation Template

After we get ICEWS21, we get a TKG with 253 relation types. We create natural language relation templates for 250 out of 253 relation types for question genera-

Table 12: Number of parameters of FORECASTTKGQA on all types of questions.

Question Type	Entity Prediction	Yes-Unknown	Fact Reasoning
Model			
FORECASTTKGQA	234,600	234,600	354,800

tion. The rest three relation types in ICEWS21 are not taken into consideration because either the verb is not suited for a question in the future tense (*Attempt to assassinate*) or there is no clear description for the subject-object-relationship of the relation type in [2] (*Demobilize armed forces* and *Demonstrate military or police power*). We use the generated relation templates for question generation of all three types of questions. For fact reasoning questions, we also use these relation templates to generate natural language choices.

C.2 Natural Language Question Template

All question templates are presented in *Question_Generation/template_icews.xlsx* which is attached with the submission in Easychair. 2-hop EPQs and their templates are generated with *Question_Generation/generate_qa_anyburl.py*.

C.3 2-Hop EPQ Generation Details

We generate 2-hop questions by utilizing AnyBURL [5], a rule-based KG reasoning model. We first split ICEWS21 into TKG snapshots, where each snapshot $\mathcal{G}_{t_i} = \{(s, r, o, t) \in \mathcal{G} | t = t_i\}$ contains all the TKG facts happening at the same timestamp. We treat every TKG snapshot as a non-temporal KG and train an AnyBURL model with the KG completion task on each TKG snapshot for rule extraction (KG completion aims to predict the missing entity from every query $(s, r, ?)$). Since AnyBURL is a KG reasoning method that cannot process temporal information, we transform every quadruple (s, r, o, t) into a corresponding triplet (s, r, o). For each TKG snapshot, we keep the 2-hop rules with a confidence higher than 0.5 extracted by AnyBURL, and manually check if two associated TKG facts in each rule potentially have a logical causation or can be used to interpret positive/negative entity relationships. After this process, we take the remaining 2-hop rules as the drafts for generating 2-hop EPQ templates. The complete list of extracted 2-hop rules is presented in *Question_Generation/anyburl_ICEWS.txt*. 2-hop EPQs and their templates are generated with *Question_Generation/generate_qa_anyburl.py*, given the extracted rules.

C.4 FRQ Generation Details

We train xERTE [3] on ICEWS21 for TKG forecasting, and pick out all the link prediction queries $(s, r, ?, t)$ whose ground-truth missing entities are ranked

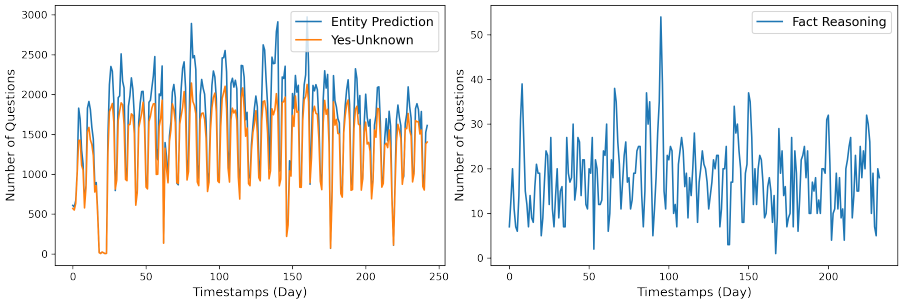
by xERTE as top 1. We collect the TKG facts corresponding to these queries for question generation. The intuition of this step is that we assume that the better xERTE performs on a link prediction query, the more reasonable the returned prior facts are for explainability. Ranking the ground-truth missing entities as top 1 indicates that xERTE performs very well on these link prediction queries. We wish to use xERTE to generate reasonable fact reasoning questions, therefore, we want it to find reasonable supporting evidence of the TKG facts by returning relevant prior facts. For each collected top 1 fact, we take the prior facts with the highest contribution score, the lowest contribution score, the median contribution score, and the second highest contribution score as the facts for generating the choices **Answer**, **Negative**, **Median** and **Hard Negative**, respectively. In this way, we can generate a large number of question candidates by fitting the corresponding facts into question templates.

After we collect all the question candidates, we have 78,606 questions. We find that there exist a large number of question candidates whose question and **Answer** share the same s, r, o. For example, the TKG fact of a question candidate is (*Sudan*, *host*, *Ramtane Lamamra*, *2021-08-01*), and the TKG fact of its **Answer** is (*Sudan*, *host*, *Ramtane Lamamra*, *2021-07-29*). We filter out all the question candidates with this pattern since we think that they are not satisfying our motivation for proposing fact reasoning questions. We wish to generate the questions that require fact reasoning, rather than finding the repeated facts happening at different timestamps. A good example of the questions we want to generate is as follows. For the question whose associated fact is (*Envoy (United States)*, *visit*, *China*, *2021-08-31*), the associated fact of its **Answer** is (*Envoy (United States)*, *express the intent to meet or negotiate*, *China*, *2021-08-30*). From human knowledge, **Answer**'s fact serves as a highly possible reason for the fact in the question, and it is also diverse from the question fact. To this end, we have 50,379 question candidates left.

We then ask five graduate students (major in computer science) to further annotate the remaining question candidates by deciding whether each of them is reasonable or not. Students are allowed to use their own knowledge and search engines to help annotation. If the students think that a question's **Answer** is not the most contributive to the question, they are asked to annotate this question as unreasonable, otherwise, they are asked to annotate it as reasonable. For every question candidate, if it is annotated as unreasonable by three students, we filter it out. As a result, we have 4,195 questions left. We use Fleiss' kappa to measure inter-annotator agreement. Fleiss' kappa is 0.63 in our annotation process. The estimated annotation time for each student is 320 hours. The annotation instruction and interface are presented in Fig. 5 and 6, respectively.

C.5 Question Time Distribution

We provide the distribution of the questions along the time axis of our dataset in Fig. 1a and Fig. 1b. We plot the number of questions at every timestamp



(a) EPQs and YUQs. (b) FRQs.

Fig. 1: Question distribution of different types of questions along the time axis.

for all three types of questions. The numbers on the horizontal axis denote how many days away from 2021-01-01.

D Full Experimental Results on EPQs

We present Table 13 as the supplement of the main results regarding EPQs in the main paper. We present the aggregated overall performance of MRR and Hits@k.

Table 13: Complete experimental results of EPQs. The best results are marked in bold.

	MRR			**Hits@1**			**Hits@10**		
Model	Overall	1-Hop	2-Hop	Overall	1-Hop	2-Hop	Overall	1-Hop	2-Hop
RoBERTa	0.161	0.166	0.149	0.098	0.104	0.085	0.282	0.288	0.268
BERT	0.253	0.279	0.182	0.168	0.192	0.106	0.421	0.451	0.342
EmbedKGQA	0.278	0.317	0.185	0.194	0.228	0.112	0.443	0.489	0.333
RoBERTa_int	0.246	0.283	0.157	0.162	0.190	0.094	0.415	0.467	0.290
BERT_int	0.275	0.314	0.183	0.189	0.223	0.107	0.447	0.490	0.344
CronKGQA	0.119	0.131	0.090	0.069	0.081	0.042	0.218	0.231	0.187
TempoQR	0.134	0.145	0.107	0.085	0.094	0.061	0.230	0.243	0.199
RoBERTa_ext	0.269	0.306	0.180	0.184	0.216	0.108	0.433	0.497	0.323
BERT_ext	0.295	0.331	0.208	0.206	0.239	0.128	0.467	0.508	0.369
ForecastTKGQA	**0.303**	**0.339**	**0.216**	**0.213**	**0.248**	**0.129**	**0.478**	**0.517**	**0.386**

E Human Benchmark Details

We ask five graduate students (major in computer science, not participating in annotation during FRQ generation) to answer 100 questions randomly sampled from the test set of FRQs. We consider two settings: (a) Humans answer FRQs with their own knowledge and inference ability. **Search engines are not allowed**; (b) Humans can turn to search engines and use the web information published **before the question timestamp** for aiding QA. We create a survey that contains the selected 100 questions. Fig. 3a and 3b show the instruction of survey and the interface of answering. We first ask the students to do the survey in setting (a), and then ask them to do it once again in setting (b). The ground-truth answers to survey questions are not shown to students throughout the whole process. Also, students have no idea which question they answer incorrectly. Thus, they cannot use this information to exclude wrong choices when they do the survey for the second time. From Table 5 of the main paper, we observe that with search engines, humans can better answer FRQs, although humans can already reach 0.936 accuracy without any additional information source.

Example to explain accuracy improvement from setting (a) to (b). We present an example explaining the human performance improvement from setting (a) to (b). Fig. 2 shows a question in the generated survey for human benchmark. In setting (a), 3 of 5 students make a mistake by choosing A. After being allowed to use search engines in setting (b), they all choose the correct choice B. This is because in setting (a), most students have no idea that *Alberto Fernández* is the president of Argentina. But after using search engines, they know the identity of *Alberto Fernández* and manage to achieve correct reasoning.

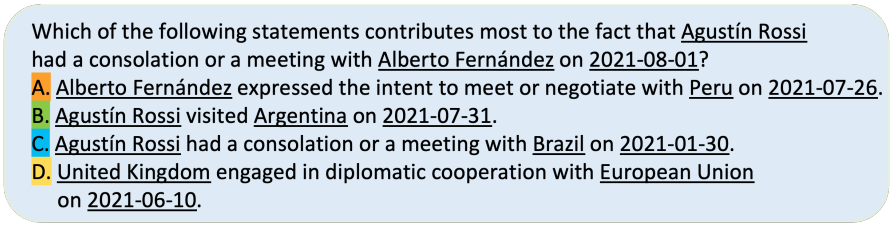


Fig. 2: Example question in the human benchmark survey.

F Details of Multi-Hop Scorer

We develop a QA model, i.e., Multi-Hop Scorer (MHS), for **non-forecasting TKGQA** (the TKGQA task proposed in [8]). We use it to prove that given the

(a) Survey insturction.

(b) Survey interface.

Fig. 3: Human benchmark survey instruction and interface.

ground-truth TKG information at the question timestamp t_q (same setting as non-forecasting TKGQA), the EPQs in ForecastTKGQuestions are answerable. Considering the non-forecasting setting, we equip MHS with two cheating TKG models (CTComplEx and CTANGO) and also design MHS by considering the multi-hop graphical structure of the snapshot $\mathcal{G}_{t_q} = \{(s, r, o, t) \in \mathcal{G} | t = t_q\}$. We illustrate MHS's model structure with an example in Fig. 4. Starting from the annotated subject entity s_q of an EPQ, MHS updates the scores of outer entities for n-hops ($n = 2$ in our experiments) until all s_q's n-hop neighbors on the snapshot $\mathcal{G}_{t_q}$ are visited. Initially, MHS assigns a score of 1 to s_q and 0 to any other unvisited entity. For each unvisited entity e, it then computes e's score as:

$$\begin{aligned} \bar{\phi}_{\text{ep}}(e) &= \sum_{(e',r) \in \mathcal{N}_e(t_q)} (\gamma \cdot \phi_{\text{ep}}(e') + \psi(e', r, e, t_q)), \\ \phi_{\text{ep}}(e) &= \frac{1}{|\mathcal{N}_e(t_q)|} \bar{\phi}_{\text{ep}}(e), \end{aligned} \tag{13}$$

where $\mathcal{N}_e(t_q) = \{(e', r) | (e', r, e, t_q) \in \mathcal{G}_{t_q}\}$ is e's 1-hop neighborhood on the snapshot $\mathcal{G}_{t_q}$ and γ is a discount factor. We couple MHS with CTComplEx and CTANGO, and define $\psi(e', r, e, t_q)$ separately. For MHS + CTComplEx, we define

$$\psi(e', r, e, t_q) = f_2(f_1(\mathbf{h}_{e'} \| \mathbf{h}_r \| \mathbf{h}_e \| \mathbf{h}_{t_q} \| \mathbf{h}_q)). \tag{14}$$

$f_1 : \mathbb{R}^{10d} \to \mathbb{R}^{2d}$, $f_2 : \mathbb{R}^{2d} \to \mathbb{R}^1$ are two neural networks. $\mathbf{h}_e, \mathbf{h}_{e'}, \mathbf{h}_r, \mathbf{h}_{t_q}$ are the CTComplEx representations of entities e, e', relation r and timestamp t_q, respectively. For MHS + CTANGO, we take the idea of ForecastTKGQA and define

$$\psi(e', r, e, t_q) = \text{Re}\left(< \mathbf{h}_{(e',t_q)}, \mathbf{h}_r, \bar{\mathbf{h}}_{(e,t_q)}, \mathbf{h}_q >\right). \tag{15}$$

$\mathbf{h}_{(e,t_q)}$, $\mathbf{h}_{(e',t_q)}$ are the CTANGO entity representations of e, e' at t_q, respectively. $\mathbf{h}_r$ is the CTANGO relation representation of r. $\mathbf{h}_q$ is BERT encoded question representation.

G Further Analysis on ForecastTKGQA

Ablation on KG Representations We conduct an ablation study by comparing the performance of ForecastTKGQA coupled with different KG (TKG) representations. We first train ComplEx on ICEWS21 and provide our model with its representations. We observe in Table 14 that TANGO representations are more effective than static KG representations in our proposed model. Besides, we switch TANGO's scoring function to TuckER [1] when we train TANGO on ICEWS21. Table 14 shows that TANGO + ComplEx aligns better to our QA module.

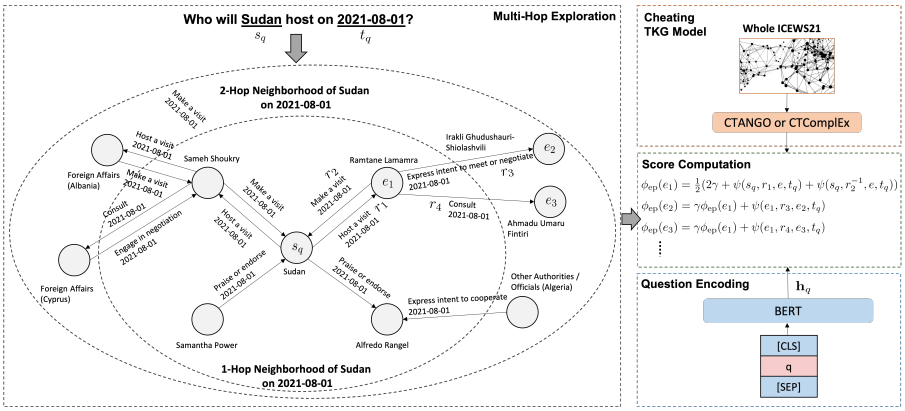


Fig. 4: Assume we have a question: *Who will Sudan host on 2021-08-01?* The annotated subject entity s_q is *Sudan* and the annotated timestamp t_q is *2021-08-01*. We first pick the snapshot $\mathcal{G}_{t_q}$ and find s_q's n-hop ($n = 2$ in our case) neighbors on $\mathcal{G}_{t_q}$. Starting from s_q, MHS updates the scores of outer entities for 2-hops until all s_q's 2-hop neighbors on $\mathcal{G}_{t_q}$ ({*Ramtane Lamamra* (e_1), *Sameh Shoukry*, *Samantha Power*, *Alfredo Rangel*, *Irakli Ghudushauri-Shiolashvili* (e_2), *Ahmadu Umaru Fintiri* (e_3), *Other Authorities/Officials (Algeria)*, *Foreign Affairs (Albania)*, *Foreign Affairs (Cyprus)* } in our example) are visited. Initially, MHS assigns a score of 1 to s_q and 0 to any other unvisited entity. To be specific, MHS first propagates scores to s_q's 1-hop neighbors on $\mathcal{G}_{t_q}$, e.g., e_1. Then through the visited 1-hop neighbors, MHS propagates scores to s_q's 2-hop neighbors. Score computation for e_1, e_2, e_3 is presented in this figure. r_2^{-1} denotes the inverse relation of r_2 that points from s_q to e_1. We transform r_2 to r_2^{-1} because we define the 1-hop neighbor of an entity with its incoming edges (following TANGO [4]). Scores are computed by considering the graphical structure of $\mathcal{G}_{t_q}$. After the score propagation process, the entity with the highest score is taken as the predicted answer e_{ans}.

Table 14: Comparison of different KG representations. w. means with. EPQ, YUQ, FRQ represent entity prediction, yes-unknown and fact reasoning questions, respectively.

Question Type	**EPQ**									**YUQ**	**FRQ**
	MRR			**Hits@1**			**Hits@10**			**Accuracy**	**Accuracy**
Model	Overall	1-Hop	2-Hop	Overall	1-Hop	2-Hop	Overall	1-Hop	2-Hop		
ForecastTKGQA w. ComplEx	0.296	0.338	0.196	0.207	0.245	0.114	0.470	0.516	0.358	0.863	0.752
ForecastTKGQA w. TANGO + TuckER	0.298	0.335	0.211	0.210	0.245	0.125	0.474	0.511	0.385	0.867	0.757
ForecastTKGQA w. TANGO + ComplEx	**0.303**	**0.339**	**0.216**	**0.213**	**0.248**	**0.129**	**0.478**	**0.517**	**0.386**	**0.870**	**0.769**

Annotation Instruction

You will be given a number of machine-generated multiple-choice questions. Each of them is coupled with four choices. You will also be given the answer labeled by machines. For every question, your task is to distinguish whether the machine labeled answer is correct (i.e., reasonable) or incorrect (i.e., unreasonable). If the machine labels correctly, please annotate the corresponding question as "reasonable", otherwise, please annotate the question as "unreasonable".
Each question is centered around a political fact (i.e., **centered fact**). **Each choice denotes a different fact that happens before the centered fact**. The answer to each question should be the choice that serves as the **most relevant evidence (cause) of the centered fact among all choices.**

Note:

(1) Pay attention to the time information specified in the facts.
(2) If none of four choices potentially leads to the centered fact, please annotate the corresponding question as unreasonable.
(3) If more than one choices seem relevant and you cannot decide which choice is the best, please also annotate as unreasonable.
(4) Feel free to use search engines, e.g., Google, to support your annotation process.

Here are two examples explaining which kind of questions should be annotated as "reasonable" and which should be annotated as "unreasonable".
Example 1:
Which of the following statements contributes most to the fact that Pedro Sanchez signed a formal agreement with Joseph Robinette Biden on 2021-08-23?
A. Pedro Sanchez expressed the intent to cooperate with Joseph Robinette Biden on 2021-08-22.
B. Pedro Sanchez engaged in diplomatic cooperation with Government (Spain) on 2021-08-22.
C. Government (Spain) made a statement to Cuba on 2021-07-27.
D. United States praised or endorsed Sayyid Ali al-Husayni al-Sistani on 2021-07-24.

Machine labeled A as the answer to this question. From human perspective, A is the strong cause of the centered fact and B, C, D are not relevant compared with A. Therefore, this question should be annotated as "reasonable".

Example 2:
Which of the following statements contributes most to the fact that Emmanuel Macron negotiated with Kamala Harris on 2021-02-18.?
A. Emmanuel Macron had a consolation or a meeting with Saad Hariri on 2021-02-14.
B. Emmanuel Macron negotiated with Saad Hariri on 2021-02-12.
C. Military (France) attacked France using aerial weapons on 2021-01-08.
D. Vladimir Putin made a statement to Iran on 2021-02-09.

Machine labeled A as the answer to this question. In fact, from human perspective, all four choices cannot serve as an obviously relevant cause of the centered fact. Therefore, this question should be annotated as "unreasonable".

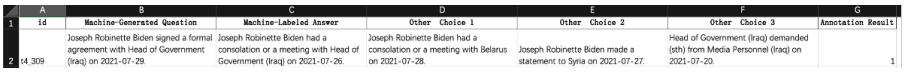
How to annotate?

You will be given an excel form containing questions and choices. The question is in Column B (Machine-Generated Question), and the machine-labeled answer is in Column C (Machine-Labeled Answer). Column D, E, F contain other choices generated by machines. Each row in the excel form corresponds to one multiple-choice question. If you think the question is "reasonable" please write 1 in Column G (Annotation Result) in the corresponding row, otherwise, please write 0. For example, if you think the question in row 1337 is unreasonable, please write 0 at G1337 of the excel form.

Why annotate?

The annotation process of the machine-generated questions will help to generate a dataset that tests machines' ability of fact reasoning and forecasting in the context of temporal knowledge question answering (TKGQA). This annotation process also aims to promote high quality dataset generation. Further, more humans will be asked to answer the sampled questions in the generated dataset for studying the difference between humans and machines in fact reasoning.

Fig. 5: Human annotation instruction for fact reasoning questions.



	A	B	C	D	E	F	G
1	id	Machine-Generated Question	Machine-Labeled Answer	Other Choice 1	Other Choice 2	Other Choice 3	Annotation Result
2	t4_309	Joseph Robinette Biden signed a formal agreement with Head of Government (Iraq) on 2021-07-29.	Joseph Robinette Biden had a consolation or a meeting with Head of Government (Iraq) on 2021-07-26.	Joseph Robinette Biden had a consolation or a meeting with Belarus on 2021-07-28.	Joseph Robinette Biden made a statement to Syria on 2021-07-27.	Head of Government (Iraq) demanded (sth) from Media Personnel (Iraq) on 2021-07-20.	1

Fig. 6: Human annotation interface for fact reasoning questions.

References

1. Balazevic, I., Allen, C., Hospedales, T.M.: Tucker: Tensor factorization for knowledge graph completion. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) 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. pp. 5184–5193. Association for Computational Linguistics (2019). https://doi.org/10.18653/v1/D19-1522, https://doi.org/10.18653/v1/D19-1522
2. Boschee, E., Lautenschlager, J., O'Brien, S., Shellman, S., Starz, J., Ward, M.: ICEWS Coded Event Data (2015). https://doi.org/10.7910/DVN/28075, https://doi.org/10.7910/DVN/28075
3. Han, Z., Chen, P., Ma, Y., Tresp, V.: Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In: 9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net (2021), https://openreview.net/forum?id=pGIHq1m7PU
4. Han, Z., Ding, Z., Ma, Y., Gu, Y., Tresp, V.: Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In: Moens, M., Huang, X., Specia, L., Yih, S.W. (eds.) Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021. pp. 8352–8364. Association for Computational Linguistics (2021). https://doi.org/10.18653/v1/2021.emnlp-main.658, https://doi.org/10.18653/v1/2021.emnlp-main.658
5. Meilicke, C., Chekol, M.W., Fink, M., Stuckenschmidt, H.: Reinforced anytime bottom up rule learning for knowledge graph completion (2020). https://doi.org/10.48550/ARXIV.2004.04412, https://arxiv.org/abs/2004.04412
6. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E.Z., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. pp. 8024–8035 (2019), https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html
7. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. CoRR **abs/1910.01108** (2019), http://arxiv.org/abs/1910.01108
8. Saxena, A., Chakrabarti, S., Talukdar, P.P.: Question answering over temporal knowledge graphs. In: Zong, C., Xia, F., Li, W., Navigli, R. (eds.) Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on

Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021. pp. 6663–6676. Association for Computational Linguistics (2021). https://doi.org/10.18653/v1/2021.acl-long.520, https://doi.org/10.18653/v1/2021.acl-long.520

9. Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Brew, J.: Huggingface's transformers: State-of-the-art natural language processing. CoRR **abs/1910.03771** (2019), http://arxiv.org/abs/1910.03771

Chapter 7

Conclusion

We have thoroughly discussed two emerging ML tasks related to TKGs: inductive knowledge representation learning and natural language QA on TKGs. Specifically, we provide a chapter-by-chapter summary of our conclusions, from Chapter 3 to Chapter 6.

- In Chapter 3, we propose a new task: TKG few-shot OOG link prediction, introducing the inductive entity representation learning problem into TKGs. We develop a model FILT dedicated to solve this new task. Given only a few edges associated to each newly-emerged entity, FILT employs a meta-learning framework that enables inductive knowledge transfer from seen entities to new unseen entities. FILT uses a time-aware graph encoder to incorporate temporal information in representation learning. It learns contextualized representations of unseen entities based on the few-shot data as well as the concept-aware information extracted from the temporal knowledge bases. A concept modeling component is used to represent entity concepts for all entities, incorporating prior knowledge into the representations of unseen entities. To evaluate models on the new task, we introduce three new datasets for TKG few-shot OOG link prediction and compare FILT with related baselines. Experimental results demonstrate that our meta-learning framework, combined with concept-aware information, enhances inductive learning for emerging entities on TKGs.

- In Chapter 4, we keep focusing on inductive entity representation learning on TKGs and present an RL-based TKG link prediction method FITCARL to solve TKG few-shot OOG LP. To predict a link involving an unseen entity, FITCARL starts at the unseen entity and traverses the background graph to find the answer entity, with the traversal process guided by a learned policy. Same as FILT, FITCARL is also a

meta-learning-based model trained with episodic training. It learns representations of newly-emerged entities by using a time-aware Transformer, with a customized time-aware positional encoding. To further alleviate the negative impact of the data scarcity problem brought by the few-shot setting, a confidence learner is proposed to be coupled with the policy network for making better decisions. A parameter-free concept regularizer is also developed to better exploit concept-aware information provided by temporal knowledge bases. Experimental results show that FITCARL achieves a new state-of-the-art and provides explainability, outperforming FILT with a grat margin.

- In Chapter 5, we focus on inductive relation representation learning on TKGs. We propose a new task: zero-shot TKG forecasting to study zero-shot relational learning in the context of TKG extrapolation. We design an LLM-empowered approach, i.e., zrLLM. zrLLM extracts the semantic information of KG relations from LLMs and introduces it into TKG representation learning. It first uses GPT-3.5[1] to generate enriched relation descriptions based on the relation texts provided by TKG datasets. Then it inputs the enriched descriptions into the encoder of T5-11B [122] and aligns the output to TKG embedding space. An extra relation history learner is designed to capture the temporal relation patterns for better reasoning, and meanwhile promote the embedding space alignment between text and TKGs. To evaluate models on the new task and prevent information leak from LLMs, we introduce three new datasets for zero-shot TKG forecasting. We couple zrLLM with a wide range of embedding-based TKG forecasting models and find that zrLLM provides huge help in forecasting the facts with zero-shot relations, and moreover, it maintains models' performance over seen relations.

- In Chapter 6, we focus on natural language QA on TKGs, in particular in the forecasting scenario. We propose a novel task: forecasting TKGQA, which is the first work combining TKG forecasting with KGQA. We propose a coupled large-scale benchmark dataset ForecastTKGQuestions that contains various types of questions including entity prediction questions, yes-unknown questions and fact reasoning questions. To solve forecasting TKGQA, we propose ForecastTKGQA, a QA model that leverages a TKG forecasting model with a pre-trained LM. We benchmark ForecastTKGQA together with several popular baselines on our dataset and

[1]https://platform.openai.com/docs/model-index-for-researchers

demonstrate that our model greatly outperforms previous methods due to its strong forecasting capability. Despite the great performance achieved by ForecastTKGQA, we further show that there still exists a large room for improvement compared with humans in answering forecasting questions. We summarize the challenges of our new task and hope our work can benefit future research in studying the forecasting power of TKGQA methods.

Based on our findings and the current interests of the research community, we outline four key future research directions.

- **Inductive Representation Learning on Both TKG Entities and Relations.** As discussed in Chapter 2, Section 2.5.6, limited efforts have been made to address unseen entities and relations simultaneously, and few-shot or zero-shot cases remain underexplored. Given that these challenges are more reflective of real-world scenarios, they represent a highly important direction for future research.

- **Inductive Representation Learning on New Data Structures.** Following [54, 75], several recent studies have introduced qualifiers into TKGs, proposing more expressive forms of TKGs, such as hyper-relational TKGs [45] and N-tuple TKGs [74]. For each TKG fact, qualifiers[2] provide additional contextual information, offering a more precise description of the facts. This introduces new challenges in developing methods that effectively model both qualifiers and the temporal dynamics in TKGs. Additionally, preliminary research has explored multimodal TKGs [98], where texts and images serve as supplementary information sources to enrich factual data. This raises new research problems, such as forecasting future facts on multimodal TKGs by integrating information across various modalities. Inductive learning on these new data structures has not been thoroughly explored, highlighting the need for further investigation. Qualifiers, along with texts, images, and potentially other modalities, offer critical insights for reasoning and may inspire novel approaches in inductive representation learning on TKGs.

- **Leveraging New Generation of Language Models in TKGQA.** Recent LLMs, e.g., GPT-4 [117], exhibit remarkable performance across various NLP tasks. Two key characteristics of these models are: (1) they unify all tasks into a text-to-text framework; (2) they use a decoder-only architecture, making it difficult to apply them

[2]Each qualifier consists of a relation along with an entity.

in the same way as text encoders like BERT [38] in ForecastTKGQA (Chapter 6). This poses a new challenge in leveraging recent LLMs for TKGQA. One recent work [57] has made the first attempt at solving TKGQA in a generative manner using modern LLMs. It decomposes TKGQA into two stages. First, a pre-trained LLM retrieves relevant factual evidence from the underlying TKG, and second, an LLM is fine-tuned to generate answers using this information. This approach marks a shift in the TKGQA framework in the era of LLMs, highlighting new challenges and opportunities, such as developing more powerful retrieval modules and introducing more efficient reasoning techniques that do not rely on fine-tuning large Transformer-based models which requires huge computational resources. One further direction is to leverage LLMs for forecasting TKGQA. Recent study [170] has demonstrated that LLMs struggle with accurate prediction and reliable confidence estimation, highlighting the importance of improved uncertainty modeling and confidence calibration. Addressing these limitations is crucial for effectively utilizing LLMs in forecasting TKGQA. Another point worth noting is that due to the quadratic complexity of self-attention in Transformer [149], recent efforts have focused on designing more efficient modules for sequence modeling, such as state space models (SSMs). One of the most popular modules is the Mamba SSM [64] which has proven as effective as Transformer in long sequence modeling while being much more efficient. Several follow-up studies have demonstrated that Mamba can replace Transformers to efficiently model various data structures, including images [181, 105], text [34, 118], and graphs [154, 43]. Notably, some new language models have been built on top of Mamba, showing promising results in language modeling. This opens up opportunities to incorporate SSM-based LMs into TKGQA, improving the efficiency of the reasoning process without compromising model performance.

- **Exploring Further Applications of TKGs.** This thesis focuses on two emerging ML tasks on TKGs and does not discuss how they can be applied to address challenges in various AI applications. KGs have recently gained significant attention in AI due to their structured nature and capacity to enhance explainability and trustworthiness [137]. They have been incorporated into a wide range of applications such as recommender system development [173] and supply chain management [68]. Building on this, an increasing number of studies are now investigating how to integrate TKGs into various applications to enable dynamic decision-making, paving the way for more adaptive and time-aware AI solutions. This opens up new opportunities

for TKGs in areas like finance, healthcare, and industrial applications.

Bibliography

[1] Insaf Adjabi, Abdeldjalil Ouahabi, Amir Benzaoui, and Abdelmalik Taleb-Ahmed. Past, present, and future of face recognition: A review. *Electronics*, 9(8), 2020.

[2] Mehdi Ali, Max Berrendorf, Mikhail Galkin, Veronika Thost, Tengfei Ma, Volker Tresp, and Jens Lehmann. Improving inductive link prediction using hyper-relational facts. In Andreas Hotho, Eva Blomqvist, Stefan Dietze, Achille Fokoue, Ying Ding, Payam M. Barnaghi, Armin Haller, Mauro Dragoni, and Harith Alani, editors, *The Semantic Web - ISWC 2021 - 20th International Semantic Web Conference, ISWC 2021, Virtual Event, October 24-28, 2021, Proceedings*, volume 12922 of *Lecture Notes in Computer Science*, pages 74–92. Springer, 2021.

[3] Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Slav Petrov, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy P. Lillicrap, Angeliki Lazaridou, Orhan Firat, James Molloy, Michael Isard, Paul Ronald Barham, Tom Hennigan, Benjamin Lee, Fabio Viola, Malcolm Reynolds, Yuanzhong Xu, Ryan Doherty, Eli Collins, Clemens Meyer, Eliza Rutherford, Erica Moreira, Kareem Ayoub, Megha Goel, George Tucker, Enrique Piqueras, Maxim Krikun, Iain Barr, Nikolay Savinov, Ivo Danihelka, Becca Roelofs, Anaïs White, Anders Andreassen, Tamara von Glehn, Lakshman Yagati, Mehran Kazemi, Lucas Gonzalez, Misha Khalman, Jakub Sygnowski, and et al. Gemini: A family of highly capable multimodal models. *CoRR*, abs/2312.11805, 2023.

[4] Farah Atif, Ola El Khatib, and Djellel Eddine Difallah. Beamqa: Multi-hop knowledge graph question answering with sequence-to-sequence prediction and beam search. In Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete, editors, *Proceedings of the 46th International*

ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023, pages 781–790. ACM, 2023.

[5] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. Dbpedia: A nucleus for a web of open data. In Karl Aberer, Key-Sun Choi, Natasha Fridman Noy, Dean Allemang, Kyung-Il Lee, Lyndon J. B. Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer, 2007.

[6] Franz Baader, Ian Horrocks, Carsten Lutz, and Ulrike Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017.

[7] Jinheon Baek, Dong Bok Lee, and Sung Ju Hwang. Learning to extrapolate knowledge: Transductive few-shot out-of-graph link prediction. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[8] Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. Multi-relational poincaré graph embeddings. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 4465–4475, 2019.

[9] Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. Tucker: Tensor factorization for knowledge graph completion. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *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 5184–5193. Association for Computational Linguistics, 2019.

[10] Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. Constraint-based question answering with knowledge graph. In Nicoletta Calzolari, Yuji Matsumoto, and Rashmi Prasad, editors, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, December 11-16, 2016, Osaka, Japan*, pages 2503–2514. ACL, 2016.

[11] Uzair Aslam Bhatti, Hao Tang, Guilu Wu, Shah Marjan, and Aamir Hussain. Deep learning with graph convolutional networks: An overview and latest applications in computational intelligence. *Int. J. Intell. Syst.*, 2023:1–28, 2023.

[12] Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In Jason Tsong-Li Wang, editor, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250. ACM, 2008.

[13] Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795, 2013.

[14] Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. ICEWS Coded Event Data, 2015.

[15] Michael M. Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: Going beyond euclidean data. *IEEE Signal Process. Mag.*, 34(4):18–42, 2017.

[16] 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. Language models are few-shot learners. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[17] Borui Cai, Yong Xiang, Longxiang Gao, He Zhang, Yunfeng Li, and Jianxin Li. Temporal knowledge graph completion: A survey. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 6545–6553. ijcai.org, 2023.

[18] Li Cai, Xin Mao, Yuhao Zhou, Zhaoguang Long, Changxu Wu, and Man Lan. A survey on temporal knowledge graph: Representation learning and applications. *CoRR*, abs/2403.04782, 2024.

[19] Jiahang Cao, Jinyuan Fang, Zaiqiao Meng, and Shangsong Liang. Knowledge graph embedding: A survey from the perspective of representation spaces. *ACM Comput. Surv.*, 56(6):159:1–159:42, 2024.

[20] Ines Chami, Adva Wolf, Da-Cheng Juan, Frederic Sala, Sujith Ravi, and Christopher Ré. Low-dimensional hyperbolic knowledge graph embeddings. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6901–6914. Association for Computational Linguistics, 2020.

[21] Jiajun Chen, Huarui He, Feng Wu, and Jie Wang. Topology-aware correlations between relations for inductive link prediction in knowledge graphs. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6271–6278. AAAI Press, 2021.

[22] Kai Chen, Ye Wang, Yitong Li, and Aiping Li. Rotateqvs: Representing temporal information as rotations in quaternion vector space for temporal knowledge graph completion. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 5843–5857. Association for Computational Linguistics, 2022.

[23] Mingyang Chen, Wen Zhang, Yuxia Geng, Zezhong Xu, Jeff Z. Pan, and Huajun Chen. Generalizing to unseen elements: A survey on knowledge extrapolation for knowledge graphs. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence, IJCAI 2023, 19th-25th August 2023, Macao, SAR, China*, pages 6574–6582. ijcai.org, 2023.

[24] Mingyang Chen, Wen Zhang, Zhen Yao, Xiangnan Chen, Mengxiao Ding, Fei Huang, and Huajun Chen. Meta-learning based knowledge extrapolation for knowledge graphs in the federated setting. In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 1966–1972. ijcai.org, 2022.

[25] Mingyang Chen, Wen Zhang, Wei Zhang, Qiang Chen, and Huajun Chen. Meta relational learning for few-shot link prediction in knowledge graphs. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *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 4216–4225. Association for Computational Linguistics, 2019.

[26] Mingyang Chen, Wen Zhang, Yushan Zhu, Hongting Zhou, Zonggang Yuan, Changliang Xu, and Huajun Chen. Meta-knowledge transfer for inductive knowledge graph embedding. In Enrique Amigó, Pablo Castells, Julio Gonzalo, Ben Carterette, J. Shane Culpepper, and Gabriella Kazai, editors, *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, Madrid, Spain, July 11 - 15, 2022*, pages 927–937. ACM, 2022.

[27] Yongrui Chen, Huiying Li, Guilin Qi, Tianxing Wu, and Tenggou Wang. Outlining and filling: Hierarchical query graph generation for answering complex questions over knowledge graphs. *IEEE Trans. Knowl. Data Eng.*, 35(8):8343–8357, 2023.

[28] Zhongwu Chen, Chengjin Xu, Fenglong Su, Zhen Huang, and Yong Dou. Incorporating structured sentences with time-enhanced BERT for fully-inductive temporal relation prediction. In Hsin-Hsi Chen, Wei-Jou (Edward) Duh, Hen-Hsen Huang, Makoto P. Kato, Josiane Mothe, and Barbara Poblete, editors, *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Infor-*

mation Retrieval, SIGIR 2023, Taipei, Taiwan, July 23-27, 2023, pages 889–899. ACM, 2023.

[29] Zhongwu Chen, Chengjin Xu, Fenglong Su, Zhen Huang, and Yong Dou. Meta-learning based knowledge extrapolation for temporal knowledge graph. In Ying Ding, Jie Tang, Juan F. Sequeda, Lora Aroyo, Carlos Castillo, and Geert-Jan Houben, editors, *Proceedings of the ACM Web Conference 2023, WWW 2023, Austin, TX, USA, 30 April 2023 - 4 May 2023*, pages 2433–2443. ACM, 2023.

[30] Zhuo Chen, Zhao Zhang, Zixuan Li, Fei Wang, Yutao Zeng, Xiaolong Jin, and Yongjun Xu. Self-improvement programming for temporal knowledge graph question answering. In Nicoletta Calzolari, Min-Yen Kan, Véronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue, editors, *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation, LREC/COLING 2024, 20-25 May, 2024, Torino, Italy*, pages 14579–14594. ELRA and ICCL, 2024.

[31] Ziyang Chen, Jinzhi Liao, and Xiang Zhao. Multi-granularity temporal question answering over knowledge graphs. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki, editors, *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 11378–11392. Association for Computational Linguistics, 2023.

[32] Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1724–1734. ACL, 2014.

[33] Andrew M. Dai and Quoc V. Le. Semi-supervised sequence learning. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3079–3087, 2015.

[34] Tri Dao and Albert Gu. Transformers are ssms: Generalized models and efficient algorithms through structured state space duality. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[35] Suresh Dara, Swetha Dhamercherla, Surender Singh Jadav, Ch Madhu Babu, and Mohamed Jawed Ahsan. Machine learning in drug discovery: A review. *Artif. Intell. Rev.*, 55(3):1947–1999, 2022.

[36] Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. Case-based reasoning for natural language queries over knowledge bases. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 9594–9611. Association for Computational Linguistics, 2021.

[37] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3837–3845, 2016.

[38] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019.

[39] Bolin Ding, Jeffrey Xu Yu, Shan Wang, Lu Qin, Xiao Zhang, and Xuemin Lin. Finding top-k min-cost connected trees in databases. In Rada Chirkova, Asuman Dogac, M. Tamer Özsu, and Timos K. Sellis, editors, *Proceedings of the 23rd International Conference on Data Engineering, ICDE 2007, The Marmara Hotel, Istanbul, Turkey, April 15-20, 2007*, pages 836–845. IEEE Computer Society, 2007.

[40] Wentao Ding, Hao Chen, Huayu Li, and Yuzhong Qu. Semantic framework based query generation for temporal question answering over knowledge graphs. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 1867–1877. Association for Computational Linguistics, 2022.

[41] Zifeng Ding, Heling Cai, Jingpei Wu, Yunpu Ma, Ruotong Liao, Bo Xiong, and Volker Tresp. zrllm: Zero-shot relational learning on temporal knowledge graphs with large language models. In Kevin Duh, Helena Gómez-Adorno, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), NAACL 2024, Mexico City, Mexico, June 16-21, 2024*, pages 1877–1895. Association for Computational Linguistics, 2024.

[42] Zifeng Ding, Bailan He, Jingpei Wu, Yunpu Ma, Zhen Han, and Volker Tresp. Learning meta-representations of one-shot relations for temporal knowledge graph link prediction. In *International Joint Conference on Neural Networks, IJCNN 2023, Gold Coast, Australia, June 18-23, 2023*, pages 1–10. IEEE, 2023.

[43] Zifeng Ding, Yifeng Li, Yuan He, Antonio Norelli, Jingcheng Wu, Volker Tresp, Yunpu Ma, and Michael Bronstein. Dygmamba: Efficiently modeling long-term temporal dependency on continuous-time dynamic graphs with state space models. *arXiv preprint arXiv:2408.04713*, 2024.

[44] Zifeng Ding, Yunpu Ma, Bailan He, Zhen Han, and Volker Tresp. A simple but powerful graph encoder for temporal knowledge graph completion. In *NeurIPS 2022 Temporal Graph Learning Workshop*, 2022.

[45] Zifeng Ding, Jingcheng Wu, Jingpei Wu, Yan Xia, and Volker Tresp. Exploring link prediction over hyper-relational temporal knowledge graphs enhanced with time-invariant relational knowledge. *CoRR*, abs/2307.10219, 2023.

[46] Zifeng Ding, Jingpei Wu, Bailan He, Yunpu Ma, Zhen Han, and Volker Tresp. Few-shot inductive learning on temporal knowledge graphs using concept-aware information. In *4th Conference on Automated Knowledge Base Construction*, 2022.

[47] Zifeng Ding, Jingpei Wu, Zongyue Li, Yunpu Ma, and Volker Tresp. Improving few-shot inductive learning on temporal knowledge graphs using confidence-augmented reinforcement learning. In Danai Koutra, Claudia Plant, Manuel Gomez Rodriguez, Elena Baralis, and Francesco Bonchi, editors, *Machine Learning and Knowledge Discovery in Databases: Research Track - European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part III*, volume 14171 of *Lecture Notes in Computer Science*, pages 550–566. Springer, 2023.

[48] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024.

[49] Lisa Ehrlinger and Wolfram Wöß. Towards a definition of knowledge graphs. In Michael Martin, Martí Cuquet, and Erwin Folmer, editors, *Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems - SEMANTiCS2016 and the 1st International Workshop on Semantic Change & Evolving Semantics (SuCCESS'16) co-located with the 12th International Con-*

ference on Semantic Systems (SEMANTiCS 2016), Leipzig, Germany, September 12-15, 2016, volume 1695 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2016.

[50] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR, 2017.

[51] Luis Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. Fast rule mining in ontological knowledge bases with AMIE+. *VLDB J.*, 24(6):707–730, 2015.

[52] Luis Antonio Galárraga, Christina Teflioudi, Katja Hose, and Fabian M. Suchanek. AMIE: association rule mining under incomplete evidence in ontological knowledge bases. In Daniel Schwabe, Virgílio A. F. Almeida, Hartmut Glaser, Ricardo Baeza-Yates, and Sue B. Moon, editors, *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013*, pages 413–422. International World Wide Web Conferences Steering Committee / ACM, 2013.

[53] Mikhail Galkin, Etienne G. Denis, Jiapeng Wu, and William L. Hamilton. Nodepiece: Compositional and parameter-efficient representations of large knowledge graphs. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[54] Mikhail Galkin, Priyansh Trivedi, Gaurav Maheshwari, Ricardo Usbeck, and Jens Lehmann. Message passing for hyper-relational knowledge graphs. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7346–7359. Association for Computational Linguistics, 2020.

[55] Mikhail Galkin, Xinyu Yuan, Hesham Mostafa, Jian Tang, and Zhaocheng Zhu. Towards foundation models for knowledge graph reasoning. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024.

[56] Octavian-Eugen Ganea, Gary Bécigneul, and Thomas Hofmann. Hyperbolic neural networks. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information*

Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada, pages 5350–5360, 2018.

[57] Yifu Gao, Linbo Qiao, Zhigang Kan, Zhihua Wen, Yongquan He, and Dongsheng Li. Two-stage generative question answering on temporal knowledge graph using large language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 6719–6734. Association for Computational Linguistics, 2024.

[58] Alberto García-Durán, Sebastijan Dumancic, and Mathias Niepert. Learning sequence encoders for temporal knowledge graph completion. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 4816–4821. Association for Computational Linguistics, 2018.

[59] Yuxia Geng, Jiaoyan Chen, Zhuo Chen, Jeff Z. Pan, Zhiquan Ye, Zonggang Yuan, Yantao Jia, and Huajun Chen. Ontozsl: Ontology-enhanced zero-shot learning. In Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia, editors, *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 3325–3336. ACM / IW3C2, 2021.

[60] Yuxia Geng, Jiaoyan Chen, Jeff Z. Pan, Mingyang Chen, Song Jiang, Wen Zhang, and Huajun Chen. Relational message passing for fully inductive knowledge graph completion. In *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, CA, USA, April 3-7, 2023*, pages 1221–1233. IEEE, 2023.

[61] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. PMLR, 2017.

[62] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. Generative adversarial nets. In

Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2672–2680, 2014.

[63] Alex Graves. Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850, 2013.

[64] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *CoRR*, abs/2312.00752, 2023.

[65] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. A survey on knowledge graph-based recommender systems. *IEEE Trans. Knowl. Data Eng.*, 34(8):3549–3568, 2022.

[66] Takuo Hamaguchi, Hidekazu Oiwa, Masashi Shimbo, and Yuji Matsumoto. Knowledge transfer for out-of-knowledge-base entities : A graph neural network approach. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1802–1808. ijcai.org, 2017.

[67] William Rowan Hamilton. Lxxviii. on quaternions; or on a new system of imaginaries in algebra. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 25(169):489–495, 1844.

[68] Yaomengxi Han, Zifeng Ding, Yushan Liu, Bailan He, and Volker Tresp. Critical path identification in supply chain knowledge graphs with large language models.

[69] Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. Dyernie: Dynamic evolution of riemannian manifold embeddings for temporal knowledge graph completion. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 7301–7316. Association for Computational Linguistics, 2020.

[70] Zhen Han, Peng Chen, Yunpu Ma, and Volker Tresp. Explainable subgraph reasoning for forecasting on temporal knowledge graphs. In *9th International Conference*

on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021. OpenReview.net, 2021.

[71] Zhen Han, Zifeng Ding, Yunpu Ma, Yujia Gu, and Volker Tresp. Learning neural ordinary equations for forecasting future links on temporal knowledge graphs. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8352–8364. Association for Computational Linguistics, 2021.

[72] Zhen Han, Ruotong Liao, Jindong Gu, Yao Zhang, Zifeng Ding, Yujia Gu, Heinz Koeppl, Hinrich Schütze, and Volker Tresp. ECOLA: enhancing temporal knowledge embeddings with contextualized language representations. In Anna Rogers, Jordan L. Boyd-Graber, and Naoaki Okazaki, editors, *Findings of the Association for Computational Linguistics: ACL 2023, Toronto, Canada, July 9-14, 2023*, pages 5433–5447. Association for Computational Linguistics, 2023.

[73] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[74] Zhongni Hou, Xiaolong Jin, Zixuan Li, Long Bai, Saiping Guan, Yutao Zeng, Jiafeng Guo, and Xueqi Cheng. Temporal knowledge graph reasoning based on n-tuple modeling. In Houda Bouamor, Juan Pino, and Kalika Bali, editors, *Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023*, pages 1090–1100. Association for Computational Linguistics, 2023.

[75] Xingyue Huang, Miguel A. Romero Orth, Pablo Barceló, Michael M. Bronstein, and İsmail İlkan Ceylan. Link prediction with relational hypergraphs. *CoRR*, abs/2402.04062, 2024.

[76] Xingyue Huang, Miguel Romero, İsmail İlkan Ceylan, and Pablo Barceló. A theory of link prediction via relational weisfeiler-leman on knowledge graphs. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[77] Nicolas Hubert, Pierre Monnin, and Heiko Paulheim. Beyond transduction: A survey on inductive, few shot, and zero shot link prediction in knowledge graphs. *CoRR*, abs/2312.04997, 2023.

[78] Shaoxiong Ji, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Networks Learn. Syst.*, 33(2):494–514, 2022.

[79] Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. Tempquestions: A benchmark for temporal question answering. In Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis, editors, *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1057–1062. ACM, 2018.

[80] Zhen Jia, Abdalghani Abujabal, Rishiraj Saha Roy, Jannik Strötgen, and Gerhard Weikum. TEQUILA: temporal question answering over knowledge bases. In Alfredo Cuzzocrea, James Allan, Norman W. Paton, Divesh Srivastava, Rakesh Agrawal, Andrei Z. Broder, Mohammed J. Zaki, K. Selçuk Candan, Alexandros Labrinidis, Assaf Schuster, and Haixun Wang, editors, *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018*, pages 1807–1810. ACM, 2018.

[81] Zhen Jia, Soumajit Pramanik, Rishiraj Saha Roy, and Gerhard Weikum. Complex temporal question answering on knowledge graphs. In Gianluca Demartini, Guido Zuccon, J. Shane Culpepper, Zi Huang, and Hanghang Tong, editors, *CIKM '21: The 30th ACM International Conference on Information and Knowledge Management, Virtual Event, Queensland, Australia, November 1 - 5, 2021*, pages 792–802. ACM, 2021.

[82] Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de Las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. Mixtral of experts. *CoRR*, abs/2401.04088, 2024.

[83] Woojeong Jin, Rahul Khanna, Suji Kim, Dong-Ho Lee, Fred Morstatter, Aram Galstyan, and Xiang Ren. Forecastqa: A question answering challenge for event forecasting with temporal text data. In Chengqing Zong, Fei Xia, Wenjie Li, and Roberto Navigli, editors, *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4636–4650. Association for Computational Linguistics, 2021.

[84] Woojeong Jin, Meng Qu, Xisen Jin, and Xiang Ren. Recurrent event network: Autoregressive structure inferenceover temporal knowledge graphs. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6669–6683. Association for Computational Linguistics, 2020.

[85] Wei Ju, Zheng Fang, Yiyang Gu, Zequn Liu, Qingqing Long, Ziyue Qiao, Yifang Qin, Jianhao Shen, Fang Sun, Zhiping Xiao, Junwei Yang, Jingyang Yuan, Yusheng Zhao, Yifan Wang, Xiao Luo, and Ming Zhang. A comprehensive survey on deep graph representation learning. *Neural Networks*, 173:106207, 2024.

[86] Jaehun Jung, Jinhong Jung, and U Kang. Learning to walk across time for interpretable temporal knowledge graph completion. In Feida Zhu, Beng Chin Ooi, and Chunyan Miao, editors, *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 786–795. ACM, 2021.

[87] Seyed Mehran Kazemi and David Poole. Simple embedding for link prediction in knowledge graphs. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 4289–4300, 2018.

[88] Shristi Shakya Khanal, P. W. C. Prasad, Abeer Alsadoon, and Angelika Maag. A systematic review: machine learning based recommendation systems for e-learning. *Educ. Inf. Technol.*, 25(4):2635–2664, 2020.

[89] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[90] Timothee Lacroix, Guillaume Obozinski, and Nicolas Usunier. Tensor decompositions for temporal knowledge base completion. In *The 8th International Conference on Learning Representations*, 2020.

[91] Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. Tensor decompositions for temporal knowledge base completion. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[92] Steinar Laenen and Luca Bertinetto. On episodes, prototypical networks, and few-shot learning. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 24581–24592, 2021.

[93] Steinar Laenen and Luca Bertinetto. On episodes, prototypical networks, and few-shot learning. *Advances in Neural Information Processing Systems*, 34:24581–24592, 2021.

[94] Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. Complex knowledge base question answering: A survey. *IEEE Trans. Knowl. Data Eng.*, 35(11):11196–11215, 2023.

[95] Julien Leblay and Melisachew Wudage Chekol. Deriving validity time in knowledge graph. In Pierre-Antoine Champin, Fabien Gandon, Mounia Lalmas, and Panagiotis G. Ipeirotis, editors, *Companion of the The Web Conference 2018 on The Web Conference 2018, WWW 2018, Lyon , France, April 23-27, 2018*, pages 1771–1776. ACM, 2018.

[96] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86(11):2278–2324, 1998.

[97] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 7871–7880. Association for Computational Linguistics, 2020.

[98] Haoxuan Li, Zhengmao Yang, Yunshan Ma, Yi Bin, Yang Yang, and Tat-Seng Chua. Mm-forecast: A multimodal approach to temporal event forecasting with large language models. *CoRR*, abs/2408.04388, 2024.

[99] Yujia Li, Shiliang Sun, and Jing Zhao. Tirgn: Time-guided recurrent graph network with local-global historical patterns for temporal knowledge graph reasoning. In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 2152–2158. ijcai.org, 2022.

[100] Zixuan Li, Xiaolong Jin, Wei Li, Saiping Guan, Jiafeng Guo, Huawei Shen, Yuanzhuo Wang, and Xueqi Cheng. Temporal knowledge graph reasoning based on evolutional representation learning. In Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai, editors, *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 408–417. ACM, 2021.

[101] Chen Liang, Jonathan Berant, Quoc V. Le, Kenneth D. Forbus, and Ni Lao. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 23–33. Association for Computational Linguistics, 2017.

[102] Ruotong Liao, Xu Jia, Yangzhe Li, Yunpu Ma, and Volker Tresp. GenTKG: Generative forecasting on temporal knowledge graph with large language models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 4303–4317, Mexico City, Mexico, June 2024. Association for Computational Linguistics.

[103] Kangzheng Liu, Feng Zhao, Guandong Xu, Xianzhi Wang, and Hai Jin. RETIA: relation-entity twin-interact aggregation for temporal knowledge graph extrapolation. In *39th IEEE International Conference on Data Engineering, ICDE 2023, Anaheim, CA, USA, April 3-7, 2023*, pages 1761–1774. IEEE, 2023.

[104] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[105] Yue Liu, Yunjie Tian, Yuzhong Zhao, Hongtian Yu, Lingxi Xie, Yaowei Wang, Qixiang Ye, and Yunfan Liu. Vmamba: Visual state space model. *CoRR*, abs/2401.10166, 2024.

[106] Yushan Liu, Yunpu Ma, Marcel Hildebrandt, Mitchell Joblin, and Volker Tresp. Tlogic: Temporal logical rules for explainable link forecasting on temporal knowledge graphs. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 4120–4127. AAAI Press, 2022.

[107] Sijie Mai, Shuangjia Zheng, Yuedong Yang, and Haifeng Hu. Communicative message passing for inductive relation reasoning. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 4294–4302. AAAI Press, 2021.

[108] Costas Mavromatis, Prasanna Lakkur Subramanyam, Vassilis N. Ioannidis, Adesoji Adeshina, Phillip Ryan Howard, Tetiana Grinberg, Nagib Hakim, and George Karypis. Tempoqr: Temporal question reasoning over knowledge graphs. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 5825–5833. AAAI Press, 2022.

[109] Christian Meilicke, Melisachew Wudage Chekol, Manuel Fink, and Heiner Stuckenschmidt. Reinforced anytime bottom up rule learning for knowledge graph completion. *CoRR*, abs/2004.04412, 2020.

[110] Tomás Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In Yoshua Bengio and Yann LeCun, editors, *1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May 2-4, 2013, Workshop Track Proceedings*, 2013.

[111] Bonan Min, Ralph Grishman, Li Wan, Chang Wang, and David Gondek. Distant supervision for relation extraction with an incomplete knowledge base. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, editors, *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 777–782. The Association for Computational Linguistics, 2013.

[112] Mehrnoosh Mirtaheri, Mohammad Rostami, Xiang Ren, Fred Morstatter, and Aram Galstyan. One-shot learning for temporal knowledge graphs. In Danqi Chen, Jonathan Berant, Andrew McCallum, and Sameer Singh, editors, *3rd Conference on Automated Knowledge Base Construction, AKBC 2021, Virtual, October 4-8, 2021*, 2021.

[113] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014.

[114] Maximilian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 6338–6347, 2017.

[115] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning, ICML 2011, Bellevue, Washington, USA, June 28 - July 2, 2011*, pages 809–816. Omnipress, 2011.

[116] Guanglin Niu, Yang Li, Chengguang Tang, Ruiying Geng, Jian Dai, Qiao Liu, Hao Wang, Jian Sun, Fei Huang, and Luo Si. Relational learning with gated and attentive neighbor aggregator for few-shot knowledge graph completion. In Fernando Diaz, Chirag Shah, Torsten Suel, Pablo Castells, Rosie Jones, and Tetsuya Sakai, editors, *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 213–222. ACM, 2021.

[117] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[118] Maciej Pióro, Kamil Ciebiera, Krystian Król, Jan Ludziejewski, and Sebastian Jaszczur. Moe-mamba: Efficient selective state space models with mixture of experts. *CoRR*, abs/2401.04081, 2024.

[119] Pengda Qin, Xin Wang, Wenhu Chen, Chunyun Zhang, Weiran Xu, and William Yang Wang. Generative adversarial zero-shot relational learning for knowledge graphs. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8673–8680. AAAI Press, 2020.

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

[121] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020.

[122] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[123] Hongyu Ren, Hanjun Dai, Bo Dai, Xinyun Chen, Michihiro Yasunaga, Haitian Sun, Dale Schuurmans, Jure Leskovec, and Denny Zhou. LEGO: latent execution-guided reasoning for multi-hop question answering on knowledge graphs. In Marina Meila

and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8959–8970. PMLR, 2021.

[124] David E. Rumelhart and James L. McClelland. *Learning Internal Representations by Error Propagation*, pages 318–362. 1987.

[125] Ali Sadeghian, Mohammadreza Armandpour, Anthony Colas, and Daisy Zhe Wang. Chronor: Rotation based temporal knowledge graph embedding. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 6471–6479. AAAI Press, 2021.

[126] Apoorv Saxena, Soumen Chakrabarti, and Partha P. Talukdar. Question answering over temporal knowledge graphs. In *ACL/IJCNLP (1)*, pages 6663–6676. Association for Computational Linguistics, 2021.

[127] Apoorv Saxena, Aditay Tripathi, and Partha P. Talukdar. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 4498–4507. Association for Computational Linguistics, 2020.

[128] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam, editors, *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer, 2018.

[129] Phillip Schneider, Tim Schopf, Juraj Vladika, Michael Galkin, Elena Simperl, and Florian Matthes. A decade of knowledge graphs in natural language processing: A survey. In Yulan He, Heng Ji, Yang Liu, Sujian Li, Chia-Hui Chang, Soujanya Poria,

Chenghua Lin, Wray L. Buntine, Maria Liakata, Hanqi Yan, Zonghan Yan, Sebastian Ruder, Xiaojun Wan, Miguel Arana-Catania, Zhongyu Wei, Hen-Hsen Huang, Jheng-Long Wu, Min-Yuh Day, Pengfei Liu, and Ruifeng Xu, editors, *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2022 - Volume 1: Long Papers, Online Only, November 20-23, 2022*, pages 601–614. Association for Computational Linguistics, 2022.

[130] Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. End-to-end structure-aware convolutional networks for knowledge base completion. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 3060–3067. AAAI Press, 2019.

[131] C. E. Shannon. Prediction and entropy of printed english. *The Bell System Technical Journal*, 30(1):50–64, 1951.

[132] Noam Shazeer. GLU variants improve transformer. *CoRR*, abs/2002.05202, 2020.

[133] Jiawei Sheng, Shu Guo, Zhenyu Chen, Juwei Yue, Lihong Wang, Tingwen Liu, and Hongbo Xu. Adaptive attentional network for few-shot knowledge graph completion. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 1681–1691. Association for Computational Linguistics, 2020.

[134] Baoxu Shi and Tim Weninger. Open-world knowledge graph completion. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 1957–1964. AAAI Press, 2018.

[135] Jianlin Su, Murtadha H. M. Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[136] Miao Su, Zixuan Li, Zhuo Chen, Long Bai, Xiaolong Jin, and Jiafeng Guo. Temporal knowledge graph question answering: A survey. *CoRR*, abs/2406.14191, 2024.

[137] Yuan Sui, Yufei He, Zifeng Ding, and Bryan Hooi. Can knowledge graphs make large language models more trustworthy? an empirical study over open-ended question answering. In *ICLR 2025 Workshop on Building Trust in Language Models and Applications*, 2025.

[138] Haohai Sun, Jialun Zhong, Yunpu Ma, Zhen Han, and Kun He. Timetraveler: Reinforcement learning for temporal knowledge graph forecasting. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 8306–8319. Association for Computational Linguistics, 2021.

[139] Yawei Sun, Lingling Zhang, Gong Cheng, and Yuzhong Qu. SPARQA: skeleton-based semantic parsing for complex questions over knowledge bases. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 8952–8959. AAAI Press, 2020.

[140] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[141] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112, 2014.

[142] Tao Tang, Xiaocai Zhang, Yuansheng Liu, Hui Peng, Binshuang Zheng, Yanlin Yin, and Xiangxiang Zeng. Machine learning on protein–protein interaction prediction: models, challenges and trends. *Briefings in Bioinformatics*, 24(2):bbad076, 03 2023.

[143] Komal K. Teru, Etienne G. Denis, and William L. Hamilton. Inductive relation prediction by subgraph reasoning. 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 9448–9457. PMLR, 2020.

[144] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

[145] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In Maria-Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48 of *JMLR Workshop and Conference Proceedings*, pages 2071–2080. JMLR.org, 2016.

[146] Ledyard R. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.

[147] Abraham Albert Ungar. Hyperbolic trigonometry and its application in the poincaré ball model of hyperbolic geometry. *Computers & Mathematics With Applications*, 41:135–147, 2001.

[148] Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, and Partha P. Talukdar. Composition-based multi-relational graph convolutional networks. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[149] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N.

Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017.

[150] Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

[151] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In Daniel D. Lee, Masashi Sugiyama, Ulrike von Luxburg, Isabelle Guyon, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3630–3638, 2016.

[152] Denny Vrandecic and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, 2014.

[153] Changjian Wang, Xiaofei Zhou, Shirui Pan, Linhua Dong, Zeliang Song, and Ying Sha. Exploring relational semantics for inductive knowledge graph completion. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 4184–4192. AAAI Press, 2022.

[154] Chloe Wang, Oleksii Tsepa, Jun Ma, and Bo Wang. Graph-mamba: Towards long-range graph sequence modeling with selective state spaces. *CoRR*, abs/2402.00789, 2024.

[155] Jiapu Wang, Boyue Wang, Meikang Qiu, Shirui Pan, Bo Xiong, Heng Liu, Linhao Luo, Tengfei Liu, Yongli Hu, Baocai Yin, and Wen Gao. A survey on temporal knowledge graph completion: Taxonomy, progress, and prospects. *CoRR*, abs/2308.02457, 2023.

[156] Liang Wang, Wei Zhao, Zhuoyu Wei, and Jingming Liu. Simkgc: Simple contrastive knowledge graph completion with pre-trained language models. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual*

Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022, pages 4281–4294. Association for Computational Linguistics, 2022.

[157] Peifeng Wang, Jialong Han, Chenliang Li, and Rong Pan. Logic attention based neighborhood aggregation for inductive knowledge graph embedding. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 7152–7159. AAAI Press, 2019.

[158] Ruijie Wang, Zheng Li, Dachun Sun, Shengzhong Liu, Jinning Li, Bing Yin, and Tarek F. Abdelzaher. Learning to sample and aggregate: Few-shot reasoning over temporal knowledge graphs. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022.

[159] Xiaozhi Wang, Tianyu Gao, Zhaocheng Zhu, Zhengyan Zhang, Zhiyuan Liu, Juanzi Li, and Jian Tang. KEPLER: A unified model for knowledge embedding and pre-trained language representation. *Trans. Assoc. Comput. Linguistics*, 9:176–194, 2021.

[160] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In Carla E. Brodley and Peter Stone, editors, *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada*, pages 1112–1119. AAAI Press, 2014.

[161] Jiapeng Wu, Meng Cao, Jackie Chi Kit Cheung, and William L. Hamilton. Temp: Temporal message passing for temporal knowledge graph completion. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 5730–5746. Association for Computational Linguistics, 2020.

[162] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Networks Learn. Syst.*, 32(1):4–24, 2021.

[163] Yuwei Xia, Mengqi Zhang, Qiang Liu, Shu Wu, and Xiao-Yu Zhang. Metatkg: Learning evolutionary meta-knowledge for temporal knowledge graph reasoning. In Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang, editors, *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 7230–7240. Association for Computational Linguistics, 2022.

[164] Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. One-shot relational learning for knowledge graphs. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pages 1980–1990. Association for Computational Linguistics, 2018.

[165] Da Xu, Chuanwei Ruan, Evren Körpeoglu, Sushant Kumar, and Kannan Achan. Inductive representation learning on temporal graphs. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[166] Xiaohan Xu, Peng Zhang, Yongquan He, Chengpeng Chao, and Chaoyang Yan. Subgraph neighboring relations infomax for inductive link prediction on knowledge graphs. In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 2341–2347. ijcai.org, 2022.

[167] Yi Xu, Junjie Ou, Hui Xu, and Luoyi Fu. Temporal knowledge graph reasoning with historical contrastive learning. In Brian Williams, Yiling Chen, and Jennifer Neville, editors, *Thirty-Seventh AAAI Conference on Artificial Intelligence, AAAI 2023, Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence, IAAI 2023, Thirteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2023, Washington, DC, USA, February 7-14, 2023*, pages 4765–4773. AAAI Press, 2023.

[168] Ikuya Yamada, Akari Asai, Jin Sakuma, Hiroyuki Shindo, Hideaki Takeda, Yoshiyasu Takefuji, and Yuji Matsumoto. Wikipedia2vec: An efficient toolkit for learning and visualizing the embeddings of words and entities from wikipedia. In Qun Liu and David Schlangen, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 23–30. Association for Computational Linguistics, 2020.

[169] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[170] Zhangdie Yuan, Zifeng Ding, and Andreas Vlachos. The future outcome reasoning and confidence assessment benchmark. *arXiv preprint arXiv:2502.19676*, 2025.

[171] Lotfi A. Zadeh. Fuzzy logic. *Computer*, 21(4):83–93, 1988.

[172] Chuxu Zhang, Huaxiu Yao, Chao Huang, Meng Jiang, Zhenhui Li, and Nitesh V. Chawla. Few-shot knowledge graph completion. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 3041–3048. AAAI Press, 2020.

[173] Jin-Cheng Zhang, Azlan Mohd Zain, Kai-Qing Zhou, Xi Chen, and Ren-Min Zhang. A review of recommender systems based on knowledge graph embedding. *Expert Syst. Appl.*, 250:123876, 2024.

[174] Jing Zhang, Bo Chen, Lingxi Zhang, Xirui Ke, and Haipeng Ding. Neural, symbolic and neural-symbolic reasoning on knowledge graphs. *AI Open*, 2:14–35, 2021.

[175] Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors,

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022, pages 5773–5784. Association for Computational Linguistics, 2022.

[176] Shuai Zhang, Yi Tay, Lina Yao, and Qi Liu. Quaternion knowledge graph embeddings. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 2731–2741, 2019.

[177] Tingyi Zhang, Jiaan Wang, Zhixu Li, Jianfeng Qu, An Liu, Zhigang Chen, and Hongping Zhi. Mustq: A temporal knowledge graph question answering dataset for multi-step temporal reasoning. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Findings of the Association for Computational Linguistics, ACL 2024, Bangkok, Thailand and virtual meeting, August 11-16, 2024*, pages 11688–11699. Association for Computational Linguistics, 2024.

[178] Xuchao Zhang, Wei Cheng, Bo Zong, Yuncong Chen, Jianwu Xu, Ding Li, and Haifeng Chen. Temporal context-aware representation learning for question routing. In James Caverlee, Xia (Ben) Hu, Mounia Lalmas, and Wei Wang, editors, *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020*, pages 753–761. ACM, 2020.

[179] Yongqi Zhang and Quanming Yao. Knowledge graph reasoning with relational digraph. In Frédérique Laforest, Raphaël Troncy, Elena Simperl, Deepak Agarwal, Aristides Gionis, Ivan Herman, and Lionel Médini, editors, *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 912–924. ACM, 2022.

[180] Cunchao Zhu, Muhao Chen, Changjun Fan, Guangquan Cheng, and Yan Zhang. Learning from history: Modeling temporal knowledge graphs with sequential copy-generation networks. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*, pages 4732–4740. AAAI Press, 2021.

[181] Lianghui Zhu, Bencheng Liao, Qian Zhang, Xinlong Wang, Wenyu Liu, and Xinggang Wang. Vision mamba: Efficient visual representation learning with bidirectional state space model. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[182] Zhaocheng Zhu, Zuobai Zhang, Louis-Pascal A. C. Xhonneux, and Jian Tang. Neural bellman-ford networks: A general graph neural network framework for link prediction. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 29476–29490, 2021.