Toward Knowledge-Centric Natural Language Processing: Acquisition, Representation, Transfer, and Reasoning

Dissertation

By

Zhen Wang, B.Eng.

Graduate Program in Department of Computer Science and Engineering

The Ohio State University

2022

Dissertation Committee:

Dr. Huan Sun, Advisor

Dr. Srinivasan Parthasarathy

Dr. Yu Su

Dr. Wei-Lun Chao

Dr. Hea-Jin Lee

Abstract

Past decades have witnessed the great success of modern Artificial Intelligence (AI) via learning incredible statistical correlations from large-scale data. However, a knowledge gap still exists between the statistical learning of AI and the human-like learning process. Unlike machines, humans can first accumulate enormous background knowledge about how the world works and then quickly adapt it to new environments by understanding the underlying concepts. For example, given the limited life experience with mammals, a child can quickly learn the new concept of a dog to infer knowledge, like a dog is a mammal, a mammal has a heart, and thus, a dog has a heart. Then the child can generalize the concept to new cases, such as a golden retriever, a beagle, or a chihuahua. However, an AI system trained on a large-scale mammal but not dog-focused dataset cannot do such learning and generalization. AI techniques will fundamentally influence our everyday lives, and bridging this knowledge gap to empower existing AI systems with more explicit human knowledge is both timely and necessary to make them more generalizable, robust, trustworthy, interpretable, and efficient.

To close this gap, we seek inspiration from how humans learn, such as the ability to abstract knowledge from data, generalize knowledge to new tasks, and reason to solve complex problems. Inspired by the human learning process, in this dissertation, we present our research efforts to address the knowledge gap between AI and human learning with a

systematic study of the full life cycle of how to incorporate more explicit human knowledge in intelligent systems. Specifically, we need first to extract high-quality knowledge from the real world (knowledge acquisition), such as raw data or model parameters. We then transform various types of knowledge into neural representations (knowledge representation). We can also transfer existing knowledge between neural systems (knowledge transfer) or perform human-like complex reasoning to enable more transparent and generalizable inference (knowledge reasoning). All stages pose unique research challenges but are also intertwined, potentially leading to a unified framework of knowledge-centric natural language processing (NLP).

This dissertation demonstrates our established achievements along the previous four directions. The introduction first elaborates on our motivation and research vision to construct a holistic and systematic view of knowledge-centric natural language processing. We describe our contributions distributed in these four directions in each chapter separately. For *knowledge acquisition*, we study extracting structured knowledge (e.g., synonyms, relations) from the text corpus that can be leveraged to build a better knowledge space. We leverage the corpus-level co-occurrence statistics to preserve privacy and personal information better. Our proposed framework can fully utilize the surface form and global context information for advanced performance. For *knowledge representation*, we focus on graph representation learning and propose to learn better representations of node pairs for pairwise prediction tasks on graphs, such as link prediction or relation classification. Our proposed method encourages the interaction between local contexts and would generate more interpretable results. For *knowledge transfer*, we present two works. The first one transfers knowledge between structured (Knowledge Base) and unstructured (text corpus) knowledge sources, and the second one transfers knowledge from pre-trained large

language models (LLMs) to downstream tasks via multitask prompt tuning. For *knowledge reasoning*, we present two works. The first one shows a self-interpretable framework for medical relation prediction that can generate human-intuitive rationales to explain neural prediction. It relied on a recall and recognition process inspired by the human memory theory from cognitive science. We verify the trustworthiness of generated rationales by conducting a human evaluation of the medical expert. The second one focuses on commonsense reasoning for better word representation learning, in which an explicit reasoning module runs over a commonsense knowledge graph to perform multi-hop reasoning. The learned vector representations can benefit downstream tasks and show the reasoning steps as interpretations.

In the last chapter, we summarize our key contributions and outline future research directions toward knowledge-centric natural language processing. Ultimately, we envision that human knowledge and reasoning should be indispensable components for the next generation of AI techniques.

To my beloved parents and those who always inspire me along this beautiful journey.

Acknowledgments

First and foremost, I would like to thank my advisor, Prof. Huan Sun, for introducing me to the field of NLP and for her immeasurable guidance in my PhD journey, not only cultivating my research skills, but also reshaping my life perspectives fundamentally. She's always supported and encouraged me to reach higher goals and has been a constant source of knowledge and inspiration.

I'm also grateful to my committee members, Prof. Srinivasan Parthasarathy, Prof. Yu Su, and Prof. Wei-Lun (Harry) Chao. Prof. Srini has always been a mentor and role model to me for his research taste and for always providing great advice on my research and future career. Prof. Su and Prof. Chao have always been supportive of my candidacy and dissertation exams.

I would like to give special thanks to Bo Zong, Nebojsa Jojic, Rameswar Panda, and Yoon Kim for mentoring me throughout three wonderful summer internships; Weifeng Liu, my undergraduate mentor for introducing me to the world of machine learning; Mingyang Zhang for his friendship and inspiration when the winter of Columbus was cold that year.

More broadly, I am thankful to many of my collaborators and mentors whom I'm lucky to have the chance to work with and learn from: Bo Zong, Wei Cheng, Xuchao Zhang, Yanchi Liu, Wenchao Yu, Jingchao Ni, Haifeng Chen at NEC Labs America; Nebojsa Jojic, Matthew Richardson at Microsoft Research; Nikolay Malkin at Mila; Batu Ozturkler at Stanford; Rameswar Panda, Leonid Karlinsky, Rogerio Feris at MIT-IBM Watson AI

Lab; Yoon Kim, Bailin Wang at MIT; Simon Lin, Jennifer Lee, Soheil Moosavinasab, Yungui Huang at Nationwide Children's Hospital. I would also like to thank many of my dear friends, without whom I could not survive throughout the PhD: Fan Bai, Cheng Zhang, Shuaichen Chang, Wuwei Lan, Wei Sun, Yang Xia, Haiyang Shi, Lingyan Yin, Bei Zhou, Amelia and Haoming, Xiao Zha, Luoshang Pan, Xiangming Gu, Hao Zhang, Qianli Feng, Guanyu Xu, Tao Hou, Longhua Wu, Kun Jin, Guowei Yan, Chenyang Xu, Qi Zhao, Shijie Ma, Jinjie Yao, Dingying Lu, Cheng Xin, Rui Zhang, Haicheng Chen, Huimin Du, Zijian Hu, Boyuan Pan, Yuantong Li at OSU (most of them are my dear friends in my early PhD years); Ziyu Yao, Jie Zhao, Yu Gu, Xiang Deng, Xiang Yue, Lingbo Mo, Shijie Chen, Ziru Chen, Tianshu Zhang, Samuel Stevens, Ashley Lewis, Bernal Jiménez Gutiérrez, Vardaan Pahuja, Boshi Wang, Jayavardhan Reddy Peddamail at OSU NLP family; Peihao Wang at UT Austin; Dongkuan Xu at NC State.

Also, many thanks to the funding agencies and work conducted in this dissertation was sponsored in part by the Patient Centered Outcomes Research Institute Funding ME-2017C1-6413, the Army Research Office under cooperative agreements W911NF-17-1-0412, NSF Grant IIS1815674, NSF CAREER #1942980, and Ohio Supercomputer Center (Center, 1987).

Finally, I am thankful to my parents and my beloved Yaoyao for their unconditional love and support throughout my PhD journey.

Vita

2016 - 2022	Ph.D., Computer Science and Engineering, The Ohio State University, USA.
Autumn 2017, 2022	Graduate Teaching Associate, Computer Science and Engineering, The Ohio State University, USA.
January 2018 - May 2022	Graduate Research Associate, Computer Science and Engineering, The Ohio State University, USA.
Summer 2022	Research Intern, MIT-IBM Watson AI Lab, USA.
Summer 2021	Research Intern, Microsoft Research Redmond, USA.
Summer 2020	Research Intern, NEC Labs America, USA.
2011 - 2015	B.Eng., Electronic Information Engineering, China University of Petroleum, China.

Publications

Zhen Wang, Rameswar Panda, Leonid Karlinsky, Rogerio Feris, Huan Sun, Yoon Kim. "Multitask Prompt Tuning Enables Parameter-Efficient Transfer Learning." Under *Review at the 11th International Conference on Learning Representations.* 2022.

Lingbo Mo*, **Zhen Wang***, Jie Zhao, Huan Sun. "Knowledge Transfer between Structured and Unstructured Sources for Complex Question Answering." In *NAACL 2022 Structured and Unstructured Knowledge Integration (SUKI).* (*Equal contribution)

Nikolay Malkin, **Zhen Wang**, Nebojsa Jojic. "Coherence Boosting: When Your Pretrained Language Model is Not Paying Enough Attention." In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8214-8236. 2022.

Batu Ozturkler, Nikolay Malkin, **Zhen Wang**, Nebojsa Jojic. "ThinkSum: Probabilistic Reasoning Over Sets Using Large Language Models." In *arXiv preprint*, arXiv:2210.01293. 2022.

Janvijay Singh, Fan Bai, **Zhen Wang**. "Frustratingly Simple Entity Tracking with Effective Use of Multi-Task Learning Models." In *arXiv preprint*, arXiv:2210.06444. 2022.

Shijie Chen, Ziru Chen, Xiang Deng, Ashley Lewis, Lingbo Mo, Samuel Stevens, **Zhen Wang**, Xiang Yue, Tianshu Zhang, Yu Su, Huan Sun. "Bootstrapping a User-Centered Task-Oriented Dialogue System." In *1st Proceedings of Alexa Prize TaskBot (Alexa Prize 2021)*

Zhen Wang, Bo Zong, Huan Sun. "Modeling Context Pair Interaction for Pairwise Tasks on Graphs." In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pp. 851-859. 2021.

Zhen Wang, Bo Zong, Wei Cheng, Xuchao Zhang, Yanchi Liu, Wenchao Yu, Jingchao Ni, Haifeng Chen, Huan Sun. "Learning Interpretable Word Representations by Commonsense Knowledge Reasoning." In *Manuscript*. 2020.

Zhen Wang, Jennifer Lee, Simon Lin, Huan Sun. "Rationalizing Medical Relation Prediction from Corpus-level Statistics." In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8078-8092. 2020.

Xiang Yue, **Zhen Wang**, Jingong Huang, Srinivasan Parthasarathy, Soheil Moosavinasab, Yungui Huang, Simon M. Lin, Wen Zhang, Ping Zhang, Huan Sun. "Graph Embedding on Biomedical Networks: Methods, Applications, and Evaluations." *Bioinformatics*, 36, no. 4 (2020): 1241-1251.

Zhen Wang, Xiang Yue, Soheil Moosavinasab, Yungui Huang, Simon Lin, Huan Sun. "SurfCon: Synonym Discovery on Privacy-Aware Clinical Data." In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1578-1586. 2019.

Jayavardhan Reddy Peddamail, Ziyu Yao, **Zhen Wang**, Huan Sun. "A Comprehensive Study of StaQC for Deep Code Summarization." In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, (Deep Learning Day, SPOTLIGHT), 2018

Weifeng Liu, **Zhen Wang**, Dapeng Tao, Jun Yu. "Hessian Regularized Sparse Coding for Human Action Recognition." In *International Conference on Multimedia Modeling*, pp. 502-511. Springer, Cham, 2015.

Fields of Study

Major Field: Computer Science and Engineering

Studies in:

Artificial Intelligence	Dr. Huan Sun
Data Mining	Dr. Srinivasan Parthasarathy
Statistics	Dr. Lo-Bin Chang

Table of Contents

List of Tables

Table **Page**

List of Figures

Part I: Introduction

Chapter 1: Introduction

1.1 Motivation: A Knowledge Gap

Human intelligence is widely considered to be characterized by human language and communication, which contribute significantly to human cognitive uniqueness (Premack, 2004; Corballis, 2014; Bickerton, 2017). Teaching machines to understand natural language and endowing them with human-level intelligence have been long-held aspirations for researchers in areas of Natural Language Processing (NLP), Machine learning (ML), and, more generally, Artificial Intelligence (AI).

Understanding human text requires the ability to reason over world knowledge both implicitly and explicitly mentioned by the text. Such knowledge, to name a few, includes factual, linguistic, scientific, and commonsense knowledge. This core challenge can be decomposed into several sub-problems, such as 1) how to acquire high-quality knowledge from data, 2) how to represent knowledge in neural systems, 3) how to transfer knowledge between systems and tasks, and 4) how to reason over knowledge for complex problem solving, etc.

Dating back to the era of rule-based systems in the last century (Winograd, 1972; Shortliffe, 1974; Clancey and Letsinger, 1982; Barker et al., 1989; Swartout et al., 1991), symbolic systems were developed in which researchers had tried to enumerate as many rules as

possible and built up logic-based systems to reason over them. Despite the great explainability of such systems and their success in specific applications, they mainly suffer two drawbacks, 1) the high cost of human efforts in building and maintaining the expert system and 2) the low generalizability of them that makes them brittle for out-of-distribution samples. At that time, knowledge is explicitly defined as logical rules, and the expert system makes the prediction by logical reasoning.

The last two decades have witnessed the rising of statistical models (Manning and Schutze, 1999) that automatically learn relevant knowledge from data via statistical learning. In the beginning, feature engineering is still required (e.g., part-of-speech features) to produce good results with machine learning models, like Support Vector Machines (SVMs). With the increase of the model's expressiveness power, less explicit knowledge or features are needed, but the prediction model that makes predictions by fitting statistical correlations becomes a black-box gradually.

More recently, deep learning-based or neural models (Krizhevsky et al., 2012; Goldberg, 2016; LeCun et al., 2015; Goodfellow et al., 2016) that take raw data (e.g., text, images) as input have revolutionized NLP-related areas and achieved superior performance across many tasks that were considered very hard to be solved previously, such as machine translation, image captioning, etc. At this stage, knowledge is primarily stored in the data implicitly, and people rely on training on large-scale data to reconstruct implicit knowledge for better performance.

Starting from around 2018, pre-trained large language models (LLMs) (Devlin et al., 2019) have demonstrated more impressive and expressive power on natural language understanding. With the fundamental building block as the Transformer (Vaswani et al., 2017), LLMs adopts simple objectives as either traditional left-to-right language modeling (e.g.,

	1980s	2000s	2010s	2018–
Knowledge source	Human-defined rules	Manually extracted features	Raw data	Model parameters
Prediction model	Expert systems	Statistical models	Deep learning models	LLMs

Table 1.1: The evolution of knowledge sources and prediction model for machine intelligence.

GPT; (Radford et al., 2018), GPT-2; (Radford et al., 2019), GPT-3; (Brown et al., 2020)) or bidirectional masked language modeling (e.g., BERT; (Devlin et al., 2019)). LLMs usually contain millions to billions of parameters and are pre-trained over large-scale text data. A new paradigm is to first pre-train an LLM and then fine-tune it on various downstream tasks. In this sense, the knowledge can be considered stored in the model parameters.

With a brief review of the development history in NLP-related areas, we summarize the evolution of knowledge sources and prediction models for advanced AI models from different eras in Table 1.1. As we can see, despite the recent advanced models' higher and higher performance, more and more implicit knowledge is directly learned from large-scale data as statistical correlations and stored in large-scale model parameters as black-boxes. On the contrary, humans can accumulate enormous background knowledge about the world to quickly adapt to new scenarios and explain their decisions by reasoning processes and rationales. Thus, compared to human-like learning processes, what is missing for advanced AI systems nowadays includes more explicit knowledge and the associated interpretable reasoning processes. We refer to the difference between such two distinct learning systems as the **Knowledge Gap** (Bajaj et al., 2022) indicating that explicit human knowledge

and reasoning are missed in advanced AI systems today, which lead to the following challenges.

1. Generalizability. Due to the expressive power of neural models, they tend to learn the spurious statistical correlations between the input and training labels. Such correlations make the model hard to generalize to different tasks. With explicit knowledge and reasoning as a form of regularization, we expect the model to make a more accurate prediction with a similar reasoning process as humans, which will lead to greater generalizability.

2. Data Efficiency. Another drawback of existing neural models is that they usually consume large-scale data for training. Since explicit knowledge will serve as a good initialization for learning and transfer, and thus, we expect knowledge-enhanced systems would require fewer labels and have high data efficiency, which is especially useful in the low-resource domains.

3. Interpretability. One of the biggest benefits of explicit knowledge and reasoning is the enhanced interpretability of neural models, where there are serious concerns about the nature of black-box models. We expect this enhanced interpretability will obtain human trust easily due to the shared knowledge space with humans and the straightforwardness of the reasoning process for people to understand with minimal knowledge barrier.

To summarize, in this dissertation, we seek to ask the research question, "Can we bridge this gap by explicit knowledge and reasoning". And to achieve this goal, we are making

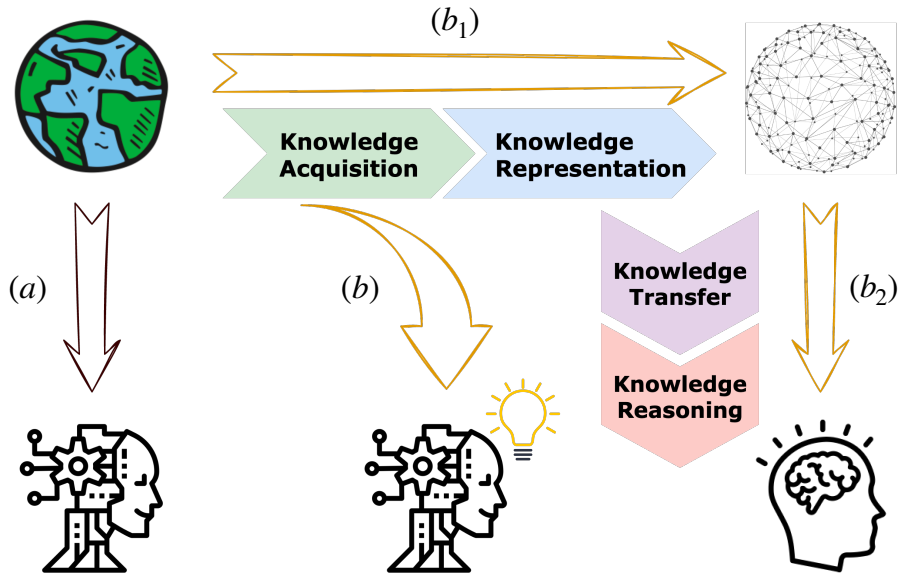

Figure 1.1: Knowledge-centric NLP (b). Compared with existing advanced AI systems (a) that directly learn statistical correlations from the real world (left top sphere), future NLP systems should be empowered by more explicit human knowledge (top right sphere) with steps of knowledge accumulation including acquisition and representation (b_1) and knowledge adaptation including transfer and reasoning (b_2) to augment existing AI systems.

efforts in the following directions: 1) discover explicit knowledge, 2) learn better knowledge representation, 3) transfer knowledge between systems and tasks, and 4) design novel reasoning-driven models.

1.2 Background and Problem Statement

In the previous section, we point out four directions, as shown in Figure 1.1, toward bridging the knowledge gap between statistical learning models and humans, i.e., knowledge acquisition, knowledge representation, knowledge transfer, and knowledge reasoning. In this section, we further introduce their core research problems and define their scope in

this dissertation. We also discuss the relationship between them to picture a blueprint for our dissertation.

Before we dive into how to manipulate the knowledge, we first clearly define what types of knowledge we are focusing on in this dissertation.

Structured and Unstructured Knowledge. There are many perspectives of defining types of knowledge (Goldstein and Papert, 1977; Barnett et al., 1990; Davis and Marcus, 2015). From the standpoint of NLP research, we focus on the external world knowledge, and without the loss of generality, we define two types of knowledge in terms of their structuredness, i.e., structured and unstructured knowledge[1]. Specifically, structured knowledge refers to highly organized data with pre-defined schemas, like relations, entities, fields, etc. The typical structured knowledge includes relational databases, graphs, etc. On the other hand, unstructured knowledge is less organized and nosier without predetermined designs, which mainly refers to open text in the context of NLP. We comprehensively summarize the pros and cons for both knowledges in Table 1.2. As we can see, they are highly complementary to each other, and how to leverage both knowledges for better understanding text becomes the core challenge for the following four directions.

Knowledge Acquisition. The core challenge of knowledge acquisition lies in where and how to acquire knowledge for machine intelligence. Though recent neural models can learn implicit knowledge directly from data, they usually suffer from 1) large-scale human supervision is required for training and 2) the learned knowledge is uninterpretable and hard to be transferred to other tasks. In the context of language, researchers have made extensive efforts to extract information (e.g., phrase, event, relation, entity, etc.) from text

[1]In the middle ground of structured and unstructured knowledge, there could be a semi-structured or partially structured knowledge, such as tables. For simplicity, we can categorize them as unstructured knowledge as they are not fully structured and still contain a significant amount of noise.

	Unstructured Knowledge	Structured Knowledge
Examples	Text	Graphs
Organization	Less organized	Well organized
Expressiveness	Rich	Limited
Noise level	High	Low
Human efforts for construction	Less human intervention	Need human efforts for high quality; Automatic methods are less satisfied
Interpretability	Less interpretable	High interpretable
Representative modeling methods	Recurrent Neural Networks Language Models	Graph Embeddings Graph Neural Networks

Table 1.2: Comparison of unstructured and structured knowledge.

data to construct structured knowledge (e.g., knowledge graphs (Suchanek et al., 2007; Bollacker et al., 2008; Carlson et al., 2010; Lehmann et al., 2015)). Our work mainly focuses on extracting structured knowledge from the text corpus of noisy user data with weakly supervised labels.

Knowledge Representation. Due to the discrete nature of language, learning better neural representations for both structured and unstructured knowledge has become the core research problem. Depending on the type of knowledge, their methodologies are dramatically different. Learning the representation for words and sentences has been through the development of word embeddings (Mikolov et al., 2013a,b; Pennington et al., 2014) and recurrent neural networks (Hochreiter and Schmidhuber, 1997; Cho et al., 2014) to the latest LPLMs. On the other hand, representation learning on structured knowledge has focused chiefly on graph learning, including Graph Embeddings (Perozzi et al., 2014; Grover and Leskovec, 2016b) and Graph Neural Networks (Gilmer et al., 2017; Hamilton et al., 2017).

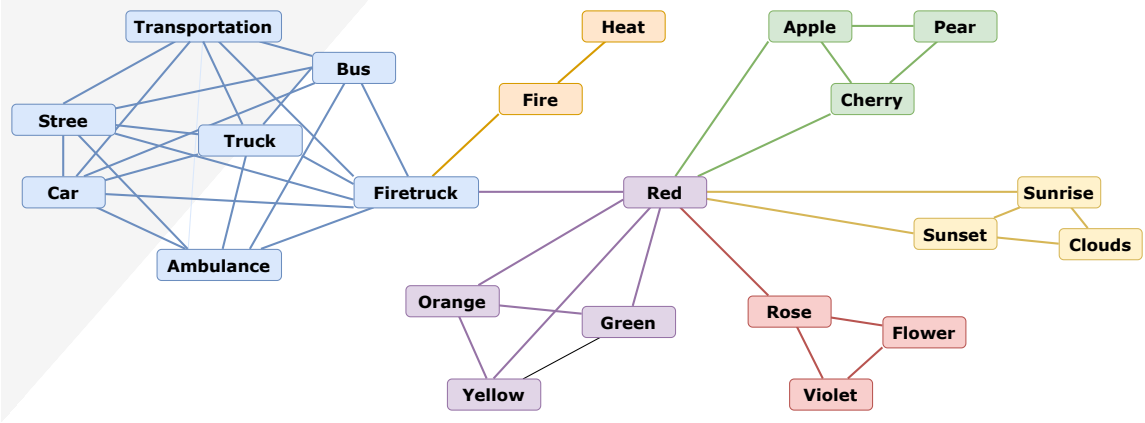

Figure 1.2: An example of semantic network (Collins and Loftus, 1975). Example adapted from Gazzaniga et al. (2006). Semantically associated words have similar colors and are closely connected (clustered).

One example of how structure helps text understanding is presented in Figure 1.2, where the semantic network could induce the meaning of words by their clustering. Such a semantic network can be inferred from a large text corpus by calculating the co-occurrence statistics of words. Our work focuses on learning advanced graph representations to support more interpretable predictions.

Knowledge Transfer. Transferring knowledge is a crucial step in the life cycle of knowledge because it allows information to be transferred between different representations. It involves many traditional transfer learning techniques (Pan and Yang, 2009) include multi-task learning, domain adaptation, etc (Liu et al., 2019; Clark et al., 2019b; Ruder et al., 2019b). But more importantly, with the new pre-training and fine-tuning paradigm of LLMs (Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019; Raffel et al., 2020), how to leverage pre-trained models for efficient adaption and better transfer their knowledge to downstream tasks are worthy of thorough exploration. We also study how to transfer the

knowledge between structured and unstructured knowledge sources, specifically, knowledge base and text corpus, where we investigate whether we can leverage one source to boost the performance of the other.

Knowledge Reasoning. As the core of this dissertation, we argue that explicit reasoning is the necessary cornerstone towards artificial intelligence (Duan et al., 2020). More complex tasks require the model to understand the relationship between high-level variables in the data and perform interpretable reasoning to derive the prediction. The history of reasoning can be traced back to the rule-based expert systems for pure symbolic reasoning. More recently, probabilistic reasoning (Pearl, 2014) and neuro-symbolic reasoning(e.g., knowledge graph reasoning (Xiong et al., 2017; Das et al., 2017a; Lin et al., 2018)) have enjoyed the merge of symbolic and neural worlds as well as their benefits. Our work focuses on neuro-symbolic reasoning, where we incorporate the neural representation of structured symbolic knowledge. This step combines the explicit knowledge and neural representation from previous Knowledge Acquisition and Representation. We summarize the relationship between these steps and two types of knowledge in Figure 1.3.

With the above important research directions, in this dissertation, we seek to answer the following research questions:

- How can we extract different structured knowledge (e.g., synonyms, relations) from the corpus of noisy text (and what if there is a challenging setting where only the corpus-level co-occurrence statistics are available)?

- How can we learn better knowledge representations for both structured (e.g., graph node pairs) and unstructured ones (e.g., words) for better performance and interpretability?

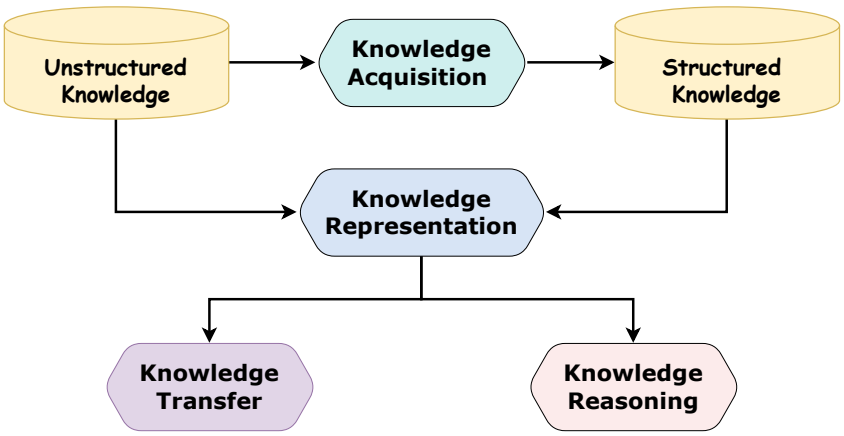

Figure 1.3: The flow of knowledge-centric NLP for the relationship between two types of knowledge and the manipulations.

- How can we efficiently and effectively transfer knowledge between systems to boost performance on downstream tasks or systems?
- How can we inject a self-interpretable reasoning process into the neural prediction with inspiration from cognitive science and verify the increased interpretability can earn the expert's trust?
- How can we let the machine practice the commonsense reasoning ability (e.g., via self-supervised training) over a predefined commonsense knowledge space for better learning and interpretability?

1.3 Contributions

This dissertation seeks to bridge the knowledge gap between AI and human-like learning by exploring the full life cycle of incorporating human knowledge into intelligent systems, i.e., acquisition, representation, transfer, and reasoning. The first two steps focus on extracting high-quality structured knowledge from the text corpus and learning better

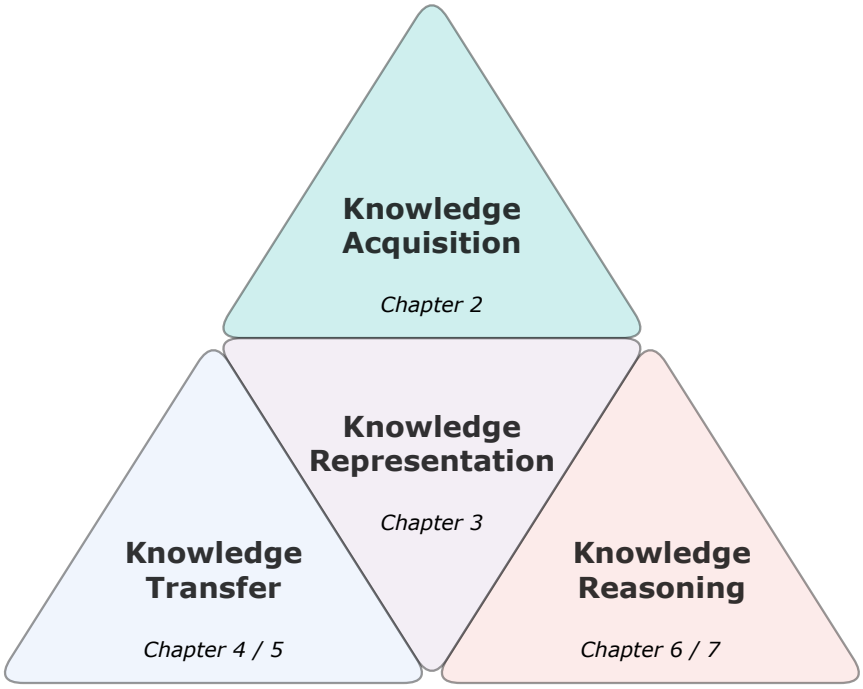

Figure 1.4: An overview of the dissertation outline.

neural structural representations. The last two steps study transferring existing knowledge between neural models or performing reasoning on structured knowledge to enable more transparent and generalizable inference in deep learning-based AI systems.

To solve these open research questions that emerged in the previous section, we now introduce our contributions toward bridging the knowledge gap of advanced machine learning and human learning processes. Figure 1.4 presents an overview of the contributions described in this dissertation. We can see that our contributions are distributed in all four directions. What is worth mentioning is that all four directions can be beneficial to others, e.g., the reasoning engine can be integrated into the acquisition or representation for greater interpretability.

More specifically, in the direction of knowledge acquisition, we make the following contributions:

- We study extracting synonyms, an essential structured knowledge, from noisy user data (i.e., clinical text) under a new setting where only the medical terms and their co-occurrence statistics are available (i.e., privacy-aware clinical data). It is a practical setting given the widespread concern about patient privacy for access to clinical text and also presents unique challenges to address for the task of synonym discovery.
- We propose a novel and effective framework named SURFCON that discovers synonyms for bother In-Vocabulary (InV) and Out-of-Vocabulary (OOV) terms. SURFCON jointly models two complementary types of information by neural models - surface form information and global context information, where the former works well for detecting synonyms that are similar in surface form, while the latter can help better find synonyms that do not look alike but are semantically similar.
- We conduct extensive experiments on publicly available privacy-aware clinical data and demonstrate the effectiveness of our framework in comparison with various types of baselines and our own model variants, especially in the challenging setting of finding synonyms for OOV terms.

In the direction of knowledge representation, we make the following contributions:

- We study pairwise predictions on graphs (e.g., link prediction) powered by a general framework CONPI of modeling the context interaction and a new type of pair embeddings that captures the semantics of any node pairs on homogeneous graphs.

- The proposed framework CONPI consists of two perspectives, node-centric and pair-centric for context interaction. It considers the mutual influence between node contexts and enhances the model interpretability by highlighting important context pairs.
- We conduct extensive experiments on two types of pairwise prediction tasks on graphs, link prediction, and relation prediction with a total of 6 datasets. In comparison with strong baselines from different categories, our framework can achieve very competitive performance and, more importantly, much better interpretability

In the direction of knowledge transfer, we make the following contributions:

- We propose a novel complex question answer (CQA) system, SimultQA to unify the knowledge-based question answering (KBQA) and Text-based question answering (TextQA) systems and study how the reasoning knowledge is transferred between these two heterogeneous knowledge sources.
- We study the efficient adaptation of large language models and transfer knowledge between tasks via prompt tuning. We propose multitask prompt tuning (MPT), which first learns a single transferable prompt by decomposing and distilling knowledge from multiple task-specific source prompts. We then learn multiplicative low-rank updates to this shared prompt to efficiently adapt it to each downstream target task.

In the direction of knowledge reasoning, we make the following contributions.

- We propose a self-interpretable neuro-symbolic framework CogStage that produces reasonable explanations for medical relation prediction based on corpus-level statistics. The interpretability is inspired by the theory of human memory (i.e., recall and recognition). The recall process retrieves strongly-associated medical terms, and

the recognition process establishes significant connections between recalled terms to make the final prediction.

- We compare CogStage with various competitive methods to show its predictive performance and conduct a human evaluation with the medical expert to demonstrate that the rationales generated by our framework can greatly help earn the expert's trust.

- We propose a self-supervised framework *CoRReL* that jointly utilizes commonsense knowledge and knowledge reasoning to enhance the generalization and interpretation of word representations. The framework consists of a self-supervised task and an explicit reasoning module as a Graph Neural Network. We empirically evaluate the effectiveness of *CoRReL* on the CommonsenseQA dataset and demonstrate its performance by comparing it with other basic word representations.

Finally, we release all our codes as well as newly collected or processed data for research purpose[2].

1.4 Dissertation Organization

Based on the core contribution of each work, we can divide this dissertation into four parts: Acquisition–mining structured knowledge from text data, Representation–learning neural structural representation, Transfer–transferring knowledge across systems and tasks, and Reasoning–explicit reasoning for interpretable machine learning.

The rest of this dissertation is organized as follows.

[2]https://github.com/zhenwang9102

In the first part, Chapter 2 focuses on knowledge acquisition where we extract synonyms from text corpus with noisy user data (Wang et al., 2019b). We first introduce the task of synonym discovery and the new setting of privacy-aware clinical text and the accompanying challenges for the task. We then present the proposed framework SURFCON that leverages both surface form and context information for accurate synonym discovery. We conduct extensive experiments and case studies on publicly available privacy-aware clinical data and show that SURFCON can outperform strong baseline methods by large margins under various settings.

In the second part, Chapter 3 presents the work on learning pair embeddings in graphs for context interaction (Wang et al., 2021c). We first introduce the notion of a general task as the pairwise prediction on graphs and emphasize the importance of context interaction. We present a unified framework with two general perspectives, node-centric and pair-centric, about modeling context pair interactions. We also propose a novel pair-centric context interaction model and a new pre-trained embedding, representing the pair semantics and showing many attractive properties. We test our models on two common pairwise prediction tasks: the link prediction task and relation prediction task, and compare them with graph feature-based, embedding-based, and Graph Neural Network (GNN)-based baselines. Our experimental results show the superior performance of the pre-trained pair embeddings and that the pair-centric interaction model outperforms all baselines by a large margin.

In the third part, Chapter 4 presents the first work on knowledge transfer, which is to transfer the multi-hop knowledge between two heterogeneous knowledge sources, the structured knowledge base, and unstructured text corpus (Mo et al., 2022). We first propose a unified system to bridge two distinct QA systems, KBQA and TextQA with a framework

of retrieval and reranking. We then leverage this system to study how knowledge is transferred between two KBQA and TextQA datasets. Chapter 5 presents the second knowledge transfer work, where we study how to efficiently adapt LLMs by transferring knowledge from a set of source tasks to a wide range of diverse downstream target tasks (Wang et al., 2023).

In the fourth part, Chapter 6 addresses the low interpretability of existing neural models for medical relation prediction (Wang et al., 2020). We first highlight the importance of interpretability and present a new interpretable framework inspired by existing theories on how human memory works, e.g., theories of recall and recognition. We conduct experiments on a real-world public clinical dataset. We show that our framework can not only achieve not only competitive predictive performance against a comprehensive list of neural baseline models but also present rationales to justify its prediction. We further collaborate with medical experts deeply to verify the usefulness of our model rationales for clinical decision-making.

Chapter 7 describes the work of learning interpretable word representations by commonsense reasoning. We first point out the necessity of interpretable word presentations and the advantages of commonsense knowledge and reasoning. Then, we propose *CoRReL* (COmmonsense knowledge Reasoning based word REpresentation Learning) that leverages commonsense knowledge and reasoning to enhance word representation learning. *CoRReL* includes pre-training and testing phases. In the pre-training phase, we propose a self-supervision task that guides *CoRReL* to learn competitive reasoning modules. In the testing phase, *CoRReL* is able to provide word pair representations and single word representations distilled from learned reasoning modules. Moreover, *CoRReL* offers reasoning paths to justify word closeness and correlation with minimal knowledge barriers. Empirical

results on public benchmark datasets demonstrate the effectiveness and interpretability of *CoRReL*.

At last, in Chapter 8, we conclude this dissertation by first summarizing the key findings and contributions, and discussing promising future research directions of pushing the research frontier forward toward knowledge-centric natural language processing.

Part II: Acquisition: Mining Structured Knowledge from Text Data

Chapter 2: Synonym Discovery on Privacy-Aware Clinical Data

Knowledge acquisition is the first stage in the life cycle of incorporating human knowledge in AI systems, which plays a similar role to how humans abstract structured knowledge by observing the real world. It aims to automatically collect high-quality structured human knowledge from the noisy environment, such as raw data, which will be reused in later applications. In this chapter, we focus on one of the most critical structured knowledge, synonym, and propose to extract synonyms from a corpus of clinical texts. But note that our methodology can be generalized to other types of structured knowledge with richer relational semantics.

Specifically, we consider unstructured clinical texts containing rich health-related information. To better utilize the knowledge buried in clinical texts, discovering synonyms for a medical query term has become an important task. Recent automatic synonym discovery methods leveraging raw text information have been developed. However, to preserve patient privacy and security, it is usually quite difficult to get access to large-scale raw clinical texts. In this paper, we study a new setting named *synonym discovery on privacy-aware clinical data* (i.e., medical terms extracted from the clinical texts and their aggregated co-occurrence counts, without raw clinical texts). To solve the problem, we propose a new framework SURFCON that leverages two important types of information in the privacy-aware clinical data, i.e., the *surface form information*, and the *global context information*

for synonym discovery. In particular, the surface form module enables us to detect synonyms that look similar while the global context module plays a complementary role to discover synonyms that are semantically similar but in different surface forms, and both allow us to deal with the OOV query issue (i.e., when the query is not found in the given data). We conduct extensive experiments and case studies on publicly available privacy-aware clinical data, and show that SURFCON can outperform strong baseline methods by large margins under various settings. We release the code as well as re-processed data on Github: https://github.com/zhenwang9102/SurfCon.

2.1 Introduction

Clinical texts in Electronic Medical Records (EMRs) are enriched with valuable information including patient-centered narratives, patient-clinician interactions and disease treatment outcomes, which can be especially helpful for future decision making. To extract knowledge from unstructured clinical texts, synonym discovery (Wang et al., 2015a) is an important task which can benefit many downstream applications. For example, when a physician issues a query term (e.g., "vitamin C") to find relevant clinical documents, automatically discovering its synonyms (e.g., "c vitamin", "vit c", "ascorbic acid") or even commonly misspelled variations (e.g. "viatmin c") can help to expand the query and thereby enhance the retrieval performance.

For the sake of patient privacy and security, it is usually quite difficult, if not impossible, for medical institutes to grant public access to large-scale raw or even de-identified clinical texts (Beam et al., 2018). Consequently, medical terms[3] and their aggregated co-occurrence

[3] A medical term is a single- or multi-word string (e.g., "Aspirin", "Acetylsalicylic Acid").

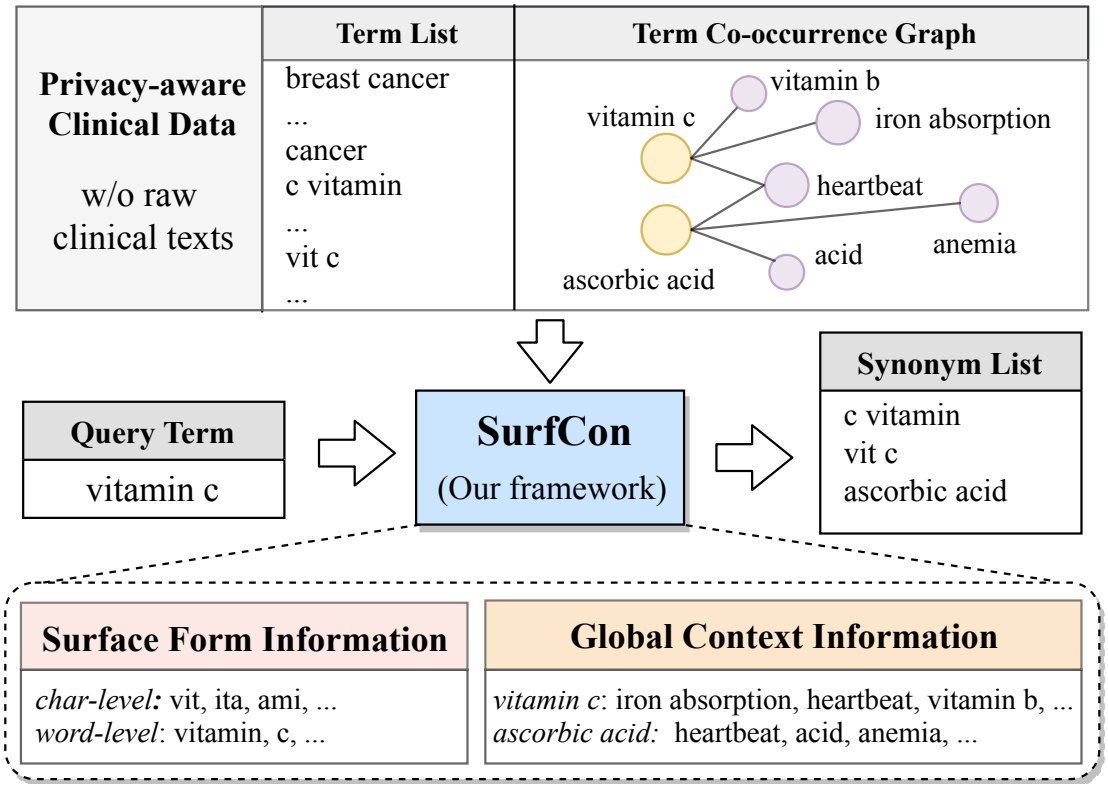

Figure 2.1: Task illustration: We aim to discover synonyms for a given query term from privacy-aware clinical data by effectively leveraging two important types of information: Surface form and global contexts.

counts extracted from raw clinical texts are becoming a popular (although not perfect) substitute for raw clinical texts for the research community to study EMR data (Finlayson et al., 2014; Ta et al., 2018; Beam et al., 2018). For example, (Finlayson et al., 2014) released millions of medical terms extracted from the clinical texts in Stanford Hospitals and Clinics as well as their global co-occurrence counts, rather than releasing raw sentences/paragraphs/documents from the clinical text corpus. In this work, we refer to the given set of medical terms and their co-occurrence statistics in a clinical text corpus as *privacy-aware* clinical data, and investigate synonym discovery task on such data (Figure 2.1): *Given a set of terms*

extracted from clinical texts as well as their global co-occurrence graph[4]*, recommend a list of synonyms for a query term*. Developing effective approaches under this setting is particularly meaningful, as they will suggest that one can utilize less sensitive information (i.e., co-occurrence statistics rather than raw sentences in clinical texts) to perform the task well.

A straightforward approach to obtain synonyms is to map the query term to a knowledge base (KB) entity and retrieve its synonyms or aliases stored in the KBs. However, it is widely known that KBs are incomplete and outdated, and their coverage of synonyms can be very limited (Wang et al., 2015c). In addition, the informal writing of clinical texts often contain variants of surface forms, layman terms, frequently misspelling words, and locally practiced abbreviations, which should be mined to enrich synonyms in KBs. Recent works (Wang et al., 2015a; Qu et al., 2017; Zhang et al., 2018a) have been focused on automatic synonym discovery from massive text corpora such as Wikipedia articles and PubMed paper abstracts. When predicting if two terms are synonyms or not, such approaches usually leverage the original sentences (a.k.a. *local* contexts) mentioning them, and hence do not apply or work well under our privacy-aware data setting where such sentences are unavailable.

Despite the lack of local contexts, we observe two important types of information carried in the privacy-aware data - surface form information and global context information (i.e., co-occurrence statistics). In this work, we aim to effectively leverage these two types of information for synonym discovery, as shown in Figure 2.1.

Some recent works (Neculoiu et al., 2016; Mueller and Thyagarajan, 2016) model the similarity between terms in the character-level. For example, (Mueller and Thyagarajan, 2016) learn the similarity between two sequences of characters, which can be applied for

[4]where each node is a medical term and each edge between two nodes is weighted by the number of times that two terms co-occur in a given context window.

discovering synonyms that look alike such as "vit c" and "vitamin c". However, we observe two common phenomena that such approaches cannot address well and would induce false positive and false negative predictions respectively: (1) Some terms are similar in surface form but do not have the same meaning (e.g., "hemostatic" and "homeostasis", where the former means a process stopping bleeding while the latter refers to a constant internal environment in the human body); (2) Some terms have the same meaning but are different in surface form (e.g., "ascorbic acid" and "vitamin c" are the same medicinal product but look different).

On the other hand, given a term co-occurrence graph, various distributional embedding methods such as (Pennington et al., 2014; Tang et al., 2015; Levy and Goldberg, 2014b) have been proposed to learn a distributional representation (a.k.a. embedding) for each term based on its *global* contexts (i.e., terms connected to it in the co-occurrence graph). The main idea behind such methods is that two terms should have similar embedding vectors if they share a lot of global contexts. However, we observe that the privacy-aware clinical data tends to be very *noisy* due to the original data processing procedure[5], which presents new challenges for utilizing global contexts to model semantic similarity between terms. For example, (Finlayson et al., 2014) prune the edges between two terms co-occurring less than 100 times, which can lead to missing edges between two related terms in the co-occurrence graph. (Ta et al., 2018) remove all concepts with singleton frequency counts below 10. Hence, the noisy nature of the co-occurrence graph makes it less accurate to embed a term based on their original contexts. Moreover, when performing the synonym discovery task, users are very likely to issue a query term that does not appear in the given co-occurrence

[5]This tends to be a common issue in many scenarios as raw data has to go through various pre-processing steps for privacy concerns.

data. We refer to such query terms as Out-of-Vocabulary (OOV). Unlike In-Vocabulary[6] query terms, OOV query terms do not have their global contexts readily available in the given graph, which makes synonym discovery even more challenging.

In this chapter, to address the above challenges and effectively utilize both the surface form and the global context information in the privacy-aware clinical data, we propose a novel framework named SURFCON which consists of a bi-level surface form encoding component and a context matching component, both based on neural models. The bi-level surface form encoding component exploits both character- and word-level information to encode a medical term into a vector. It enables us to compute a surface score of two terms based on their encoding vectors. As mentioned earlier, such surface score works well for detecting synonyms that look similar in surface form. However, it tends to miss synonymous terms that do not look alike. Therefore, we propose the context matching component to model the semantic similarity between terms, which plays a complementary role in synonymy discovery.

Our context matching component first utilizes the bi-level surface form encoding vector for a term to predict its potential global contexts. Using predicted contexts rather than the raw contexts in the given graph enables us to handle OOV query terms and also turns out to be effective for InV query terms. Then we generate a semantic vector for each term by aggregating the semantic features from predicted contexts using two mechanisms - static and dynamic representation mechanism. Specifically, given term a and term b, the dynamic mechanism aims to learn to weigh the importance of individual terms in a's contexts based on their semantic matching degree with b's contexts, while the static mechanism assigns

[6]Query terms that appear in the given co-occurrence graph are referred to as In-Vocabulary (InV).

equal weights to all terms in one's contexts. The former takes better advantage of individual terms within the contexts and empirically demonstrates superior performance.

Our contributions are summarized in three folds:

- We study the task of synonym discovery under a new setting, i.e., on privacy-aware clinical data, where only a set of medical terms and their co-occurrence statistics are given, and local contexts (e.g., sentences mentioning a term in a corpus) are not available. It is a practical setting given the wide concern about patient privacy for access to clinical texts and also presents unique challenges to address for effective synonym discovery.

- We propose a novel and effective framework named SURFCON that can discover synonyms for both In-Vocabulary (InV) and Out-of-Vocabulary (OOV) query terms. SURFCON considers two complementary types of information based on neural models - surface form information and global context information of a term, where the former works well for detecting synonyms that are similar in surface form while the latter can help better find synonyms that do not look alike but are semantically similar.

- We conduct extensive experiments on publicly available privacy-aware clinical data and demonstrate the effectiveness of our framework in comparison with various baselines and our own model variants.

2.2 Related Work

Character Sequence Encoding. To capture the character-level information of terms, neural network models such as Recurrent Neural Networks and Convolutional Neural Networks can be applied on character sequences (Ballesteros et al., 2015; Kim et al., 2016). Further, CHARAGRAM (Wieting et al., 2016), FastText (Bojanowski et al., 2016), and

CharNGram (Hashimoto et al., 2017) are proposed to represent terms and their morphological variants by capturing the shared subwords and n-grams information. However, modeling character-level sequence information only is less capable of discovering semantically similar synonyms, and our framework considers global context information to discover those synonyms.

Word and Graph/Network Embedding. Word embedding methods such as word2vec (Mikolov et al., 2013b) and Glove (Pennington et al., 2014) have been proposed and successfully applied to mining relations of medical phrases (Wang et al., 2015a; Pakhomov et al., 2016). More recently, there has been a surge of graph embedding methods that seek to encode structural graph information into low-dimensional dense vectors, such as Deepwalk (Perozzi et al., 2014), LINE (Tang et al., 2015). Most of the embedding methods can only learn embedding vectors for words in the corpus or nodes in the graph, and thus fail to address the OOV issue. On the other hand, some more recent inductive graph embedding works, such as Planetoid (Yang et al., 2016a), GraphSAGE (Hamilton et al., 2017), and SEANO (Liang et al., 2018), could generate embeddings for nodes that are unobserved in the training phase by utilizing their node features (e.g., text attributes). *However, most of them assume the neighborhood of those unseen nodes is known, which is not the case for our OOV issue as the real contexts of an OOV term are unknown.* Since Planetoid (Yang et al., 2016a) can generate node embeddings based on node features such as character sequence encoding vectors, it can handle the OOV issue and is chosen as a baseline model.

Synonym Discovery. A variety of methods have been proposed to detect synonyms of medical terms, ranging from utilizing lexical patterns (Weeds et al., 2004) and clustering (Matsuo et al., 2006) to the distributional semantics models (Hagiwara et al., 2009). There are some more recent works on automatic synonym discovery (Wang et al., 2015a;

Qu et al., 2017; Zhang et al., 2018a; Shen et al., 2019). For example, (Wang et al., 2015a) try to learn better embeddings for terms in medical corpora by incorporating their semantic types and then build a linear classifier to decide whether a pair of medical terms is synonyms or not. (Qu et al., 2017) combine distributional and pattern based methods for automatic synonym discovery. However, many aforementioned models focus on finding synonyms based on raw texts information, which is not suitable for our privacy-aware clinical data. In addition, nearly all methods could only find synonyms for terms that appear in the training corpus and, thus cannot address the OOV query terms.

2.3 Preliminaries

In this section, we clarify several terminologies used in this chapter as well as our problem definition:

Privacy-aware Clinical Data. Electronic medical records (EMRs) typically contain patient medical information such as discharge summary, treatment, and medical history. In EMRs, a significant amount of clinical information remains under-tapped in the unstructured clinical texts. However, due to privacy concerns, access to raw or even de-identified clinical texts in large quantities is quite limited. Also, traditional de-identification methods, e.g., removing the 18 HIPAA identifiers (Stubbs and Uzuner, 2015), require significant manual efforts for the annotation (Dorr et al., 2006). Moreover, there also exists the risk that de-identified data can be attacked and recovered by the re-identification in some cases (Garfinkel, 2015). Thus, to facilitate research on EMRs, an increasingly popular substitute strategy for releasing raw clinical texts is to extract medical terms and their aggregated co-occurrence counts from the corpus (Beam et al., 2018; Ta et al., 2018; Finlayson et al., 2014). We refer to such data as privacy-aware clinical data in this chapter. Converting

raw sentences to co-occurrence data protects privacy as original patient records are very unlikely to be recovered. However, the local context information contained in the raw sentences is also lost, which makes various tasks including synonym discovery more challenging under privacy-aware datasets.

Medical Term Co-occurrence Graph. A medical term-term co-occurrence graph is defined as G=(V, E), where V is the set of vertices, each representing a medical term extracted from clinical texts. Each vertex has a surface form string (e.g., "vitamin c", "cancer") which is the spelling of the medical term. E is the set of edges, each weighted by how many times two terms co-occur in a certain context window (e.g., notes from patient records within 1 day).

Medical Term Synonym. Synonyms of a medical term refer to other medical terms that can be used as its alternative names (Qu et al., 2017). For example, "vit c", "c vitamin" and "ascorbic acid" refer to the same medicinal product, while "Alzheimer's disease" and "senile dementia" represent the same disease. In our dataset, the extracted medical terms are mapped to the Unified Medical Language System (UMLS) (Bodenreider, 2004) Concept Unique Identifier (CUI) by (Finlayson et al., 2014). Different terms mapping to the same UMLS CUI are treated as synonyms for model training/development/testing.

Task Definition. We formally define our task of synonym discovery on privacy-aware clinical data as: *Given a medical term co-occurrence graph G, for a query term q (which can be either In-Vocabulary or Out-of-Vocabulary), recommend a list of medical terms from G that are likely to be synonyms of q.*

2.4 SURFCON Framework

In this section, we introduce our proposed framework SURFCON for synonym discovery on privacy-aware clinical data.

2.4.1 Overview

We observe two important types of information carried in the privacy-aware clinical data: surface form information of a medical term and the global contexts from the given co-occurrence graph. On the one hand, existing approaches (Neculoiu et al., 2016) using character-level features to detect synonyms could work well when synonyms share a high string similarity, but tend to produce false positive predictions (when two terms look similar but are not synonyms, e.g., "hemostatic" and "homeostasis") and false negative predictions (when two terms are synonyms but look very different, e.g., "ascorbic acid" and "vitamin c"). On the other hand, the global contexts of a term under the privacy-aware setting tend to be noisy partly due to the original data pre-processing procedure, which also presents challenges for using them to model the semantic similarity between terms. Thus, a framework that is able to effectively leverage these two types of information needs to be carefully designed.

Towards that end, we propose SURFCON (Figure 2.2) and summarize its high-level ideas as below:

(1) Given a query term (whether being InV or OOV), the bi-level surface form encoding component and the context matching component score a candidate term[7] respectively based on the surface form information and global context information. The former enables us to find synonyms that look similar to the query term by considering both character- and

[7]Every term in the given co-occurrence graph can be a candidate term.

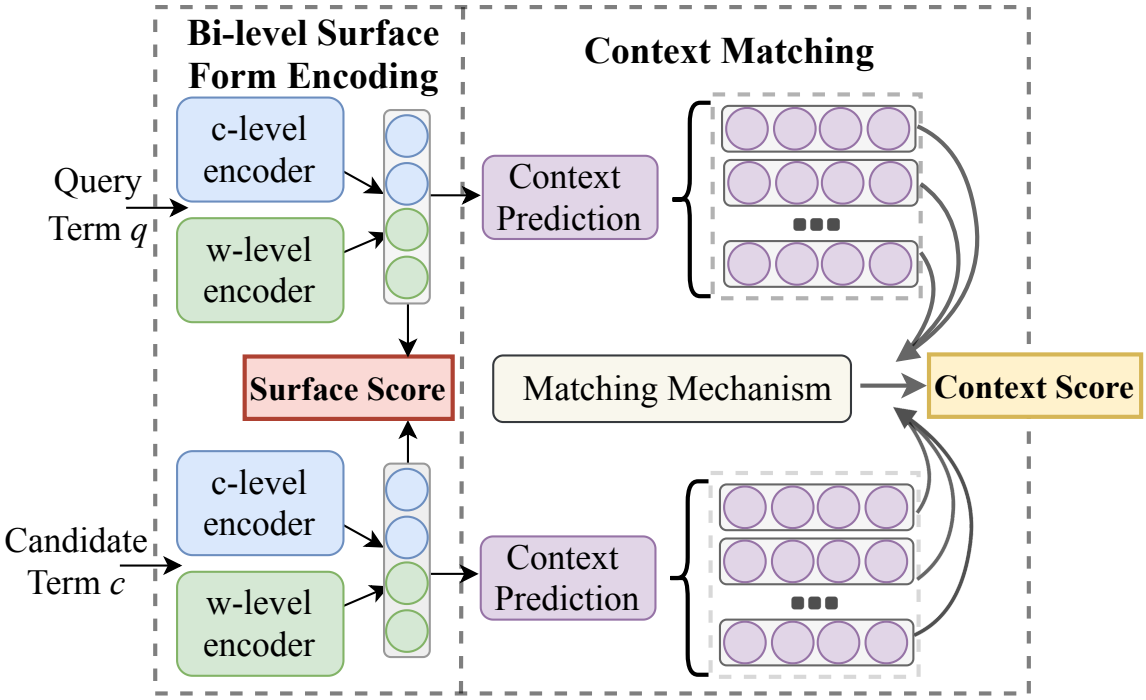

Figure 2.2: Framework overview. For each query term, a list of candidate terms will be ranked based on both the surface and context scores.

word-level information, and the latter complements it by capturing the semantic similarity between terms to better address the false positive and false negative problem mentioned earlier.

(2) Considering the original global contexts being noisy as well as the existence of OOV query terms, instead of directly leveraging the raw global contexts, the context matching component will first utilize the surface form encoding vector of a term to *predict* its potential global contexts[8]. We then investigate a novel dynamic context matching mechanism (see Section 2.4.2 for details) to evaluate if two terms are synonyms based on their predicted contexts.

[8]For terms in the co-occurrence graph, predicting contexts can be treated as denoising its original global contexts (or edges)

(3) The two components are combined by a weighted score function, in which parameters are jointly optimized with a widely used ranking algorithm ListNet (Cao et al., 2007). At testing time, given a query term, candidate terms are ranked based on the optimized score function.

2.4.2 Methodology

Now we describe the two components of SURFCON: Bi-level Surface Form Encoding and Context Matching in details.

Bi-level Surface Form Encoding

The bi-level surface form encoding of our framework aims to model the similarity between two terms at the surface form level, as we observe that two terms tend to be synonymous if they are very similar in surface forms. Such observation is intuitive but works surprisingly well in synonym discovery task. Driven by this observation, we design the bi-level surface form encoding component in a way that both of character- and word-level information of terms are captured. Then, a score function is defined to measure the surface form similarity for a pair of terms based on their bi-level encoding vectors. The bi-level encoders are able to encode surface form information of both InV terms and OOV terms.

Specifically, as shown in Figure 2.2, given a query term q and a candidate term c, we denote their character-level sequences as $x_q = \{x_{q,1}, ..., x_{q,m_q}\}, x_c = \{x_{c,1}, ..., x_{c,m_c}\}$, and their word-level sequences as $w_q = \{w_{q,1}, ..., w_{q,n_q}\}, w_c = \{w_{c,1}, ..., w_{c,n_c}\}$, where m_q, n_q, m_c, n_c are the length of the character-level sequence and word-level sequence of the query term and the candidate term respectively. Then we build two encoders ENC^{ch}

and ENC^{wd} to capture the surface form information at the character- and word-level respectively:

$$\begin{aligned} s_q^{ch} &= \text{ENC}^{ch}(x_{q,1}, ..., x_{q,m_q}), s_q^{wd} = \text{ENC}^{wd}(w_{q,1}, ..., w_{q,n_q}) \\ s_c^{ch} &= \text{ENC}^{ch}(x_{c,1}, ..., x_{c,m_c}), s_c^{wd} = \text{ENC}^{wd}(w_{c,1}, ..., w_{c,n_c}) \end{aligned} \tag{2.1}$$

where $s_q^{ch}, s_c^{ch} \in \mathbb{R}^{d_c}$ are the character-level embeddings for the query and candidate terms, and $s_q^{wd}, s_c^{wd} \in \mathbb{R}^{d_w}$ are the word-level embeddings for the query and candidate terms respectively.

Note that there has been a surge of effective encoders that model sequential information from character-level or word-level, ranging from simple look-up table (e.g., character n-gram (Hashimoto et al., 2017) and Skip-Gram (Mikolov et al., 2013b)) to complicated neural network architectures (e.g., CNN (Kim et al., 2016), LSTM (Ballesteros et al., 2015) and Transformer (Vaswani et al., 2017), etc.). For simplicity, here, we adopt simple look-up tables for both character-level embeddings and word-level embeddings. Instead of randomly initializing them, we borrow pre-trained character n-gram embeddings from (Hashimoto et al., 2017) and word embeddings from (Pennington et al., 2014). Our experiments also demonstrate that these simple encoders can well encode surface form information of medical terms for synonym discovery task. We leave evaluating more complicated encoders as our future work.

After we obtain the embeddings at both levels, we concatenate them and apply a non-linear function to get the surface vector s for the query and candidate term. Let us denote such encoding process as a function $h(\cdot)$ with the input as term q or c and the output as the surface vector s_q or s_c:

$$\begin{aligned} s_q &= h(q) = \tanh([s_q^{ch}, s_q^{wd}]W_s + b_s), \\ s_c &= h(c) = \tanh([s_c^{ch}, s_c^{wd}]W_s + b_s) \end{aligned} \tag{2.2}$$

where the surface vectors $s_q, s_c \in \mathbb{R}^{d_s}$, and $W_s \in \mathbb{R}^{(d_c+d_w)\times d_s}, b_s \in \mathbb{R}^{d_s}$ are weight matrix and bias for a fully-connected layer.

Next, we define the surface score for a query term q and a candidate term c to measure the surface form similarity based on their encoding vectors s_q and s_c:

$$\text{Surface Score}\,(q, c) = f_s(s_q, s_c) \tag{2.3}$$

Context Matching

In order to discover synonyms that are not similar in surface form, and also observing that two terms tend to be synonyms if their global contexts in the co-occurrence graph are semantically very relevant, we design the context matching component to capture the semantic similarity of two terms by carefully leveraging their global contexts. We first illustrate the intuition behind this component using a toy example:

Example 1. [Toy Example for Illustration.] *Assume we have a query term "vitamin c" and a candidate term "ascorbic acid". The former is connected with two terms "iron absorption" and "vitamin b" in the co-occurrence graph as global contexts, while the latter has "fatty acids" and "anemia" as global contexts.*

Our context matching component essentially aims to use a term's contexts to represent its semantic meaning and a novel *dynamic context matching mechanism* is developed to determine the importance of each individual term in one's contexts. For example, *"iron absorption"* is closely related to *"anemia"* since the disease "anemi" is most likely to be caused by the iron deficiency. Based on the observation, we aim to increase the relative importance of *"iron absorption"* and *"anemia"* in their respective context sets when representing the semantic meaning of *"vitamin c"* and *"ascorbic acid"*. Therefore, we develop a novel dynamic context matching mechanism to be introduced shortly.

In order to recover global contexts for OOV terms and also noticing the noisy nature of the co-occurrence graph mentioned earlier, we propose an *inductive context prediction module* to predict the global contexts for a term based on its surface form information instead of relying on the raw global contexts in the given co-occurrence graph.

Inductive Context Prediction Module. Let us first denote a general medical term as t. For a term-term co-occurrence graph, we treat all InV terms as possible context terms and denote them as $\{u_j\}_{j=1}^{|V|}$ where $|V|$ is the total number of terms in the graph. The inductive context prediction module aims to predict how likely term u_j appears in the context of t (denoted as the conditional probability $p\left(u_j|t\right)$). To learn a good context predictor, we utilize all existing terms in the graph as term t, i.e., $t \in \{u_i\}_{i=1}^{|V|}$ and the conditional probability becomes $p\left(u_j|u_i\right)$.

Formally, the probability of observing term u_j in the context of term u_i is denoted as:

$$p\left(u_j|u_i\right) = \frac{\exp\left(\nu_{u_j}^T \cdot s_{u_i}\right)}{\sum_{k=1}^{|V|} \exp\left(\nu_{u_k}^T \cdot s_{u_i}\right)} \tag{2.4}$$

where $s_{u_i} = h(u_i)$ and $h(\cdot)$ is the same encoder function defined in section 2.4.2. $\nu_{u_j} \in \mathbb{R}^{d_o}$ is the context embedding vector corresponding to term u_j and we let $d_o = d_s$. The predicted distribution $p\left(u_j|u_i\right)$ is optimized to be close to the empirical distribution $\hat{p}\left(u_j|u_i\right)$ defined as:

$$\hat{p}\left(u_j|u_i\right) = \frac{w_{ij}}{\sum_{(i,k)\in E} w_{ik}} \tag{2.5}$$

where E is the set of edges in the co-occurrence graph and w_{ij} is the weight between term u_i and term u_j. We adopt the cross entropy loss function for optimizing:

$$L_n = -\sum_{u_i,u_j \in V} \hat{p}(u_j|u_i) \log\left(p(u_j|u_i)\right) \tag{2.6}$$

When the number of terms in the graph $|V|$ is very large, it is computationally costly to calculate the conditional probability $p\left(u_j|u_i\right)$, and one can utilize the negative sampling

algorithm (Mikolov et al., 2013a) to train our inductive context predictor efficiently. The loss function Eqn. 2.6 can be modified as:

$$\log \sigma(\nu_{u_j}^T \cdot s_{u_i}) + \sum_{n=1}^{N_0} E_{u_n \sim P_n(u)}[\log \sigma(-\nu_{u_n}^T \cdot s_{u_i})] \tag{2.7}$$

where $\sigma(x) = 1/(1 + \exp(-x))$ and u_n is the negative sample drawn from the noise distribution $P_n(u) \propto d_u^{3/4}$. N_0 is the number of negative samples and d_u is the degree of term u in the co-occurrence graph.

Now, given a term t (either InV or OOV), we can select the top-K terms as its predicted contexts based on the predicted probability distribution $p(\cdot|t)$. Next, we describe the dynamic context matching mechanism to model the semantic similarity of two terms based on their predicted contexts.

Dynamic Context Matching Mechanism. Inspired by previous works on neighborhood aggregation based graph embedding methods (Hamilton et al., 2017; Velickovic et al., 2018), which generate an embedding vector for an InV node by aggregating features from its neighborhood (contexts), we introduce two semantic vectors respectively for the query term and the candidate term, $v_q, v_c \in \mathbb{R}^{d_e}$, and learn them by aggregating the feature vectors of their corresponding top-K predicted contexts from previous module.

Let us define $v_q^i \in \mathbb{R}^{d_e}$ as the feature vector of the i-th term in query term q's context while $v_c^j \in \mathbb{R}^{d_e}$ as the feature vector of the j-th term in candidate term c's context, and their context sets as $\Phi(q) = \{v_q^i\}_{i=1}^K$, $\Phi(c) = \{v_c^j\}_{j=1}^K$. Essentially, as we aim to capture the semantic meaning of terms, the feature vectors v_q^i's and v_c^j's are expected to contain semantic information. Also noticing that all predicted context terms are InV terms (i.e., in the co-occurrence graph), which allows us to adopt widely used graph embeddings, such as LINE(2nd) (Tang et al., 2015) as their feature vectors.

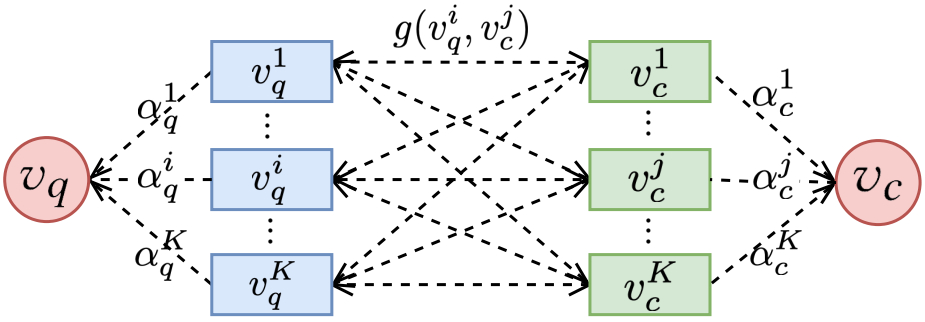

Figure 2.3: Dynamic Context Matching Mechanism.

One naive way to obtain the context semantic vectors, v_q and v_s is to average vectors in their respective context set. Since such v_q (or v_c) does not depend on the other one, we refer to such vectors as "static" representations for terms.

In contrast to the static approach, we propose the *dynamic context matching mechanism* (as shown in Figure 2.3), which weighs each term in the context of q (or c) based on its matching degree with terms in the context of c (or q) and hence the context semantic vector representation v_q (or v_c) is *dynamically* changing depending on which terms it is comparing with. More specifically, let us define $g(x, y) = \tanh(xW_m y^T)$ as a nonlinear function parameterized with weight matrix $W_m \in \mathbb{R}^{d_e \times d_e}$ to measure the similarity between two row vectors x and y. For each context vector v_q^i of the query term, we calculate its weight based on how it matches with c's contexts overall:

$$\mathsf{match}\left[v_q^i, \Phi(c)\right] = \mathsf{Pooling}\left[g(v_q^i, v_c^1), ..., g(v_q^i, v_c^K)\right] \tag{2.8}$$

For the pooling operation, we empirically choose the mean pooling strategy as it performs better than alternatives such as max pooling in our experiments. Then we normalize the

weight of v_q^i as:

$$\alpha_q^i = \frac{e^{\text{match}[v_q^i, \Phi(c)]}}{\sum_{k=1}^{K} e^{\text{match}[v_q^k, \Phi(c)]}} \tag{2.9}$$

Finally, the context semantic vector for the query term v_q is calculated through a weighted combination of q's contexts:

$$v_q = \sum_{i=1}^{K} \alpha_q^i \cdot v_q^i \tag{2.10}$$

Following the same procedure, we can obtain the context semantic vector v_c for the candidate term w.r.t. the query term. Then we define the context score for a query term q and a candidate term c to measure their semantic similarity based on v_q and v_c:

$$\text{Context Score}\,(q, c) = f_c(v_q, v_c) \tag{2.11}$$

2.4.3 Model Optimization and Inference

Objective Function. Given a query term q and a candidate term c, to capture their similarity based on surface forms and global contexts, we define the final score function as:

$$f(q, c) = (1 - \gamma) \cdot f_s(s_q, s_c) + \gamma \cdot f_c(v_q, v_c) \tag{2.12}$$

$f_s(\cdot)$ and $f_c(\cdot)$ are similarity functions between two vectors, e.g., cosine similarity or bilinear similarity. Now we obtain the recommendation probability of each candidate $t_i \in \{t_1, ..., t_N\}$ given a query q:

$$p(t_i|q) = \frac{e^{f(q,t_i)}}{\sum_{k=1}^{N} e^{f(q,t_k)}} \tag{2.13}$$

where N is the size of the candidate set. Finally, we adopt the ListNet (Cao et al., 2007) ranking framework which minimizes the cross entropy loss for query term q:

$$L_r = -\sum_{i=1}^{N} p^*(t_i|q) \log p(t_i|q) \tag{2.14}$$

where $p^*(t_i|q)$ is the normalized ground-truth distribution of a list of ranking scores as $\{r_i\}_{i=1}^N$ where r_i equals to 1 if q and t_i are synonyms and 0 otherwise.

Training. For efficiency concerns, we adopt a two-phase training strategy: We first train the inductive context prediction module by loss function L_n (Eqn. 2.6) in the term-term co-occurrence graph, and sample top-K contexts based on the predicted probability distribution and use them in the context matching component. Then, we train the ranking framework by minimizing the ranking loss L_r (Eqn. 2.14).

Inference. At the inference stage, we treat all InV terms as candidates for a given query. Since the dynamic representation mechanism involves pairwise term matching between the contexts of the query term and those of each candidate term and can have a high computational cost when the candidate set size is large, we adopt a two-step strategy: (1) For a given query term, select its top-N high potential candidates based on the surface form encoding vector and the context semantic vector obtained by the static representation mechanism; (2) Re-rank the selected candidates by applying our SURFCON framework with the dynamic representation mechanism.

2.5 Experiments

Now we evaluate our proposed framework SURFCON to show the effectiveness of leveraging both surface form information and global context information for synonym discovery.

2.5.1 Datasets

Medical Term Co-occurence Graph. We adopt publicly available sets of medical terms with their co-occurrence statistics which are extracted by (Finlayson et al., 2014) from 20 million clinical notes collected from Stanford Hospitals and Clinics(Lowe et al., 2009)

since 1995. Medical terms are extracted using an existing phrase mining tool (LePendu et al., 2012) by matching with 22 clinically relevant ontologies such as SNOMED-CT and MedDRA. And co-occurrence frequencies are counted based on how many times two terms co-occur in the same temporal *bin* (i.e., a certain timeframe in patient's records), e.g., 1, 7, 30, 90, 180, 365, and ∞-day *bins*.

Without loss of generality, we choose 1-day per-bin and ∞-day per-bin[9] graphs to evaluate different methods. We first convert the global counts between nodes to the PPMI values (Levy and Goldberg, 2014a) and adopt subsampling (Mikolov et al., 2013b) to filter very common terms, such as "medical history", "medication dose", etc. We choose these two datasets because they have very different connection density as shown in Table 2.1, and denote them as **1-day** and **All-day** datasets.

Synonym Label. In the released datasets, (Finlayson et al., 2014) provided a term-to-UMLS CUI mapping based on the same 22 ontologies as used when extracting terms. They reduced the ambiguity of a term by suppressing its least likely meaning so as to provide a high-quality mapping. We utilized such mapping to obtain the synonym labels: Terms mapped to the same UMLS CUI are treated as synonyms, e.g., terms like "c vitamin", "vit c", "ascorbic acid" are synonyms as they are all mapped to the concept "Ascorbic Acid" with ID C0003968.

Query Terms. Given a medical term-term co-occurrence graph, terms in the graph that can be mapped to UMLS CUIs are treated as potential query terms, and we split all such terms into training, development and testing sets. Here, since all terms appear in the given co-occurrence graph, this testing set is referred to as the **InV testing set**. We also create an **OOV testing set**: Under a UMLS CUI, terms not in the co-occurrence graph are treated

[9]Per-bin means each unique co-occurring term-term pair is counted at most once for each relevant bin of a patient. We refer readers to (Finlayson et al., 2014) for more information.

		1-day dataset	All-day dataset
# Nodes		52,804	43,406
# Edges		16,197,319	50,134,332
Average # Degrees		613.5	2310.0
# Train Terms		9,451	7,021
# Dev Terms		960	726
# InV Test Terms	All	960	726
	Dissim	175	152
# OOV Test Terms	All	2,000	2,000
	Dissim	809	841

Table 2.1: Statistics of our datasets.

Method Category	Methods	**1-day** Dataset					**All-day** Dataset				
		Dev	InV Test		OOV Test		Dev	InV Test		OOV Test	
			All	Dissim	All	Dissim		All	Dissim	All	Dissim
Surface form based methods	CharNgram (Hashimoto et al., 2017)	0.8755	0.8473	0.4657	0.7427	0.4131	0.8652	0.8553	0.4615	0.7675	0.4424
	CHARAGRAM (Wieting et al., 2016)	0.8705	0.8507	0.5504	0.7609	0.5142	0.8915	0.8805	0.5153	0.8119	0.5282
	SRN (Neculoiu et al., 2016)	0.8886	0.8565	0.5102	0.7241	0.4341	0.8460	0.8170	0.4523	0.7110	0.4176
Global context based methods	Word2vec (Mikolov et al., 2013b)	0.3838	0.3748	0.3188	-	-	0.4801	0.476	0.4180	-	-
	LINE(2nd) (Tang et al., 2015)	0.4279	0.4301	0.3494	-	-	0.5068	0.5043	0.4369	-	-
	DPE-NoP (Qu et al., 2017)	0.6222	0.6107	0.4855	-	-	0.5928	0.5949	0.4938	-	-
Hybrid methods (surface+context)	Concept Space (Wang et al., 2015a)	0.8094	0.8109	0.4690	-	-	0.8064	0.7924	0.5574	-	-
	Planetoid (Yang et al., 2016a)	0.8813	0.8514	0.5612	0.731	0.4714	0.8818	0.8765	0.6963	0.7403	0.4986
Our model and variants	SurfCon (Surf-Only)	0.9160	0.9053	0.6145	0.8228	0.5829	0.9034	0.8958	0.6006	0.8183	0.5622
	SurfCon (Static)	0.9242	0.9151	0.6542	0.8285	0.5933	0.9170	0.9019	0.6656	0.8203	0.5664
	SurfCon	**0.9348**	**0.9176**	**0.6821**	**0.8301**	**0.6009**	**0.9219**	**0.9199**	**0.7171**	**0.8232**	**0.5673**

Table 2.2: Model evaluation in MAP with random candidate selection.

as OOV query terms and are paired with their synonyms which are in the graph to form positive pairs. We sample 2,000 of such OOV query terms for experiments. In addition, since synonyms with different surface forms tend to be more challenging to discover (e.g., "vitamin c" vs. "ascorbic acid"), we also sample a subset named **Dissim** under both InV and OOV testing set, where query terms paired with their dissimilar synonyms[10] are selected. Statistics of our training/dev/testing sets are given in Table 2.1.

[10]Dissimilarity is measured by Levenshtein edit distance (Gomaa and Fahmy, 2013) with a threshold (0.8).

2.5.2 Experimental Setup

Baseline methods.

We compare SURFCON with the following 10 methods. The baselines can be categorized by three types: (i) Surface form based methods, which focus on capturing the surface form information of terms. (ii) Global context based methods, which try to learn embeddings of terms for synonym discovery; (iii) Hybrid methods, which combine surface form and global context information. The others are our model variants.

Surface form based methods. (1) *CharNgram* (Hashimoto et al., 2017): We borrow pre-trained character n-gram embeddings from (Hashimoto et al., 2017) and take the average of unique n-gram embeddings for each term as its feature, and then train a bilinear scoring function following previous works (Qu et al., 2017; Zhang et al., 2018a). (2) *CHARAGRAM* (Wieting et al., 2016): Similar as above, but we further fine-tune CharNgram embeddings using synonym supervision. (3) *SRN* (Neculoiu et al., 2016): A Siamese network structure is adopted with a bi-directional LSTM to encode character sequence of each term and cosine similarity is used as the scoring function.

Global context based methods. (4) *Word2vec* (Mikolov et al., 2013b): A popular distributional embedding method. We obtain word2vec embeddings by doing SVD decomposition over the Shifted PPMI co-occurrence matrix (Levy and Goldberg, 2014b). We treat the embeddings as features and use a bilinear score function for synonym discovery. (5) *LINE(2nd)* (Tang et al., 2015): A widely-adopted graph embedding approach. Similarly, embeddings are treated as features and a bilinear score function is trained to detect synonyms. (6) *DPE-NoP* (Qu et al., 2017): DPE is proposed for synonym discovery on text corpus, and consists of a distributional module and a pattern module, where the former utilizes global context information and the latter learns patterns from raw sentences. Since

raw texts are unavailable in our setting, we only deploy the distributional module (a.k.a. DPE-NoP in (Qu et al., 2017)).

Hybrid methods. (7) *Concept Space Model* (Wang et al., 2015a): A medical synonym extraction method that combines word embeddings and heuristic rule-based string features. (8) *Planetoid* (Yang et al., 2016a): An inductive graph embedding method that can generate embeddings for both observed and unseen nodes. We use the bi-level surface form encoding vectors as the input and take the intermediate hidden layer as embeddings. Similarly, a bilinear score function is used for synonym discovery.

Model variants. (9) SURFCON *(Surf-Only)*: A variant of our framework which only uses the surface score for ranking. (10) SURFCON *(Static)*: Our framework with static representation mechanism. By comparing these variants, we verify the performance gain brought by modeling global contexts using different matching mechanisms.

For baseline methods (1-3 and 8) and our models, we test them under both InV and OOV settings. For the others (4-7), because they rely on embeddings that are only available for InV terms, we only test them under InV setting.

Candidate Selection and Performance Evaluation.

For evaluating baseline methods and our model, we experiment with two strategies: (1) Random candidate selection. For each query term, we randomly sample 100 non-synonyms as negative samples and mix them with synonyms for testing. This strategy is widely adopted by previous work on synonym discovery for testing efficiency (Wang et al., 2015a; Zhang et al., 2018a). (2) Inference-stage candidate selection. As mentioned in section 2.4.3, at the inference stage, we first obtain high potential candidates in a lightweight way. Specifically, after the context predictor is pre-trained, for all terms in the given graph as well as the query term, we generate their surface form vector s and context semantic vector

v obtained by the static representation. Then we find top 50 nearest neighbors of the query term respectively based on s and v using cosine similarity. Finally, we apply our methods and baselines to re-rank the 100 high potential candidates. We refer to these two strategies as *random candidate selection* and *inference-stage candidate selection*.

For evaluation, we adopt a popular ranking metric Mean Average Precision defined as $\mathsf{MAP} = \frac{1}{|Q|}\sum_{i=1}^{|Q|}\frac{1}{m_i}\sum_{j=1}^{m_i}\mathsf{Precision}(R_{ij})$, where R_{ij} is the set of ranked terms from 1 to j, m_i is the length of i-th list, and $|Q|$ is the number of queries.

Implementation details

Our framework is implemented in Pytorch (Paszke et al., 2017) with Adam optimizer (Kingma and Ba, 2015). The dimensions of character embeddings (d_c), word embeddings (d_w), surface vectors (d_s), and sementic vectors (d_e) are set to be 100, 100, 128, 128. Early stopping is used when the performance in the dev sets does not increase continuously for 10 epochs. We directly optimize Eqn. 2.6 since the number of terms in our corpus is not very large, and set $f_s(\cdot)$ and $f_c(\cdot)$ to be cosine similarity and bilinear similarity function respectively, based on the model performance on the dev sets. When needed, string similarities are calculated by using the Distance package[11]. Pre-trained CharNgram (Hashimoto et al., 2017) embeddings are borrowed from the authors[12]. For CHARAGRAM (Wieting et al., 2016), we initialize the n-gram embeddings by using pre-trained CharNgram and fine-tune them on our dataset by the synonym supervision. We learn LINE(2nd) embeddings (Tang et al., 2015) by using OpenNE[13]. Heuristic rule-based matching features of

[11]https://github.com/doukremt/distance

[12]https://github.com/hassyGo/charNgram2vec

[13]https://github.com/thunlp/OpenNE

Concept Space model are implemented according to (Wang et al., 2015a). Code, datasets, and more implementation details are available online[14].

2.5.3 Results and Analysis

Evaluation with Random Candidate Selection

We compare all methods under random candidate selection strategy with the results shown in Table 2.2.

(1) Comparing SURFCON with surface form based methods.

Our model beats all surface form based methods, including strong baselines such as SRN that use complicated sequence models to capture character-level information. This is because: 1) Bi-level encoder of SURFCON could capture surface form information from both character- and word-level, while baselines only consider either of them; 2) SURFCON captures global context information, which could complement surface form information for synonym discovery. In addition, in comparison with CharNgram and CHARAGRAM, our model variant SURFCON (Surf-Only), which also only uses surface form information, obtains consistently better performance, especially in the OOV Test set. The results demonstrate that adding word-level surface form information is useful to discover synonyms.

(2) Comparing SURFCON with global context based methods.

SURFCON substantially outperforms all other global context based methods (Word2vec, LINE(2nd) and DPE-NoP). This is largely due to the usage of surface form information. In fact, as one can see, global context based methods are generally inferior to surface form based methods, partly due to the fact that a large part of synonyms are similar in surface form, while only a small portion of them are in very different surface form. Thus, detecting synonyms without leveraging surface information can hardly lead to good results.

[14] https://github.com/yzabc007/SurfCon

Besides, our context matching component conducts context prediction and matching strategies, which takes better advantage of global context information and thus lead to better performance on the synonym discovery task.

(3) Comparing SurfCon with hybrid methods. We also compare our model with baselines that combine both surface form and global context information. First, SURFCON is superior to the concept space model because the latter simply concatenates distributional embeddings with rule-based string features, e.g., the number of shared words as features and apply a logistic regression classifier for classification. Further, SURFCON also performs better than Planetoid, partly because our framework more explicitly leverages both surface form and global context information to formulate synonym scores, while Planetoid relies on one embedding vector for each term which only uses surface form information as input.

(4) Comparing SURFCON with its variants. To better understand why SURFCON works well, we compare it with several variants. Under both datasets, SURFCON (Surf-Only) already outperforms all baselines demonstrating the effectiveness of our bi-level surface form encoding component. With the context matching component in SURFCON (Static), the performance is further improved, especially under *InV Test Dissim* setting where synonyms tend to have different surface forms and we observe around 4% performance gain. Further, by using dynamic representation in context matching mechanism, SURFCON obtains better results, which demonstrates that the dynamic representation is more effective to utilize context information compared with the static strategy.

Evaluation at Inference Stage

To further evaluate the power of our model in real practice, we test its performance at the inference stage as mentioned in section 2.4.3. Due to space constraint, we only show

Methods	1-day		All-day	
	InV Test	OOV Test	InV Test	OOV Test
CHARAGRAM (Hashimoto et al., 2017)	0.3921	0.4044	0.3941	0.3913
DPE-NoP (Qu et al., 2017)	0.2396	-	0.2408	-
Planetoid (Yang et al., 2016a)	0.4563	0.4268	0.3765	0.3812
SURFCON	0.5525	0.5068	0.4686	0.4661

Table 2.3: Model evaluation at inference stage.

the comparison in Table 2.3 between SURFCON and several strong baselines revealed by Table 2.2. In general, the performance of all methods decreases at the inference stage compared with the random candidate selection setting, because the constructed list of candidates becomes harder to rank since surface form and context information are already used for the construction. For example, a lot of non-synonyms with similar surface form are often included in the candidate list. Even though the task becomes harder, we still observe our model outperforms the strong baselines by a large margin (e.g., around 8% at least) under all settings.

Parameter Sensitivity

Here we investigate the effect of two important hyper-parameters: The coefficient γ which balances the surface score and the context score, and the number of predicted contexts K used for context matching. As shown in Figure 2.5.3(a), the performance of SURFCON first is improved as γ increases, which is expected because as more semantic information is incorporated, SURFCON could detect more synonyms that are semantically similar. When we continue to increase γ, the performance begins to decrease and the reason is that

Query Term	”unable to vocalize” (InV)	”marijuana” (OOV)
SURFCON Top Ranked Candidates	”does not vocalize” ”aphonia” ”loss of voice” ”vocalization” **”unable to phonate”**	**”marijuana abuse”** **”cannabis”** ”cannabis use” ”marijuana smoking” ”narcotic”
Labeled Synonym Set	”unable to phonate”	”cannabis” ”marijuana abuse” ”marihuana abuse”

Table 2.4: Case studies on the 1-day dataset. Bold terms are synonyms in our labeled set while underlined terms are not but quite similar to the query term in semantics.

surface form is also an important source of information that needs to be considered. SURFCON achieves the best performance roughly at $\gamma = 0.3$ indicating surface form information is relatively more helpful for the task than global context information. This also aligns well with our observation that synonyms more often than not have similar surface forms. Next, we show the impact of K in Figure 2.5.3(b). In general, when K is small (e.g., $K = 10$), the performance is not as good since little global context information is considered. Once K increases to be large enough (e.g., ≥ 50), the performance is not sensitive to the variation under most settings showing that we can choose smaller K for computation efficiency but still with good performance.

2.5.4 Case Studies

We further conduct case studies to show the effectiveness of SURFCON. Two query terms “unable to vocalize” and “marijuana” are chosen respectively from the InV and OOV test set where the former is defined as the inability to produce voiced sound and the latter

is a psychoactive drug used for medical or recreational purposes. As shown in Table 2.4, for the InV query "unable to vocalize", our model can successfully detect its synonyms such as "unable to phonate", which already exists in the labeled synonym set collected based on term-to-UMLS CUI mapping as we discussed in Section 2.3. More impressively, our framework also discovers some highly semantically similar terms such as "does not vocalize" and "aphonia", even if some of them are quite different in surface form from the query term. For the OOV query "marijuana", SURFCON ranks its synonym "marijuana abuse" and "cannabis" at a higher place. Note that the other top-ranked terms are also very relevant to "marijuana".

2.6 Discussion and Conclusion

In this chapter, we study synonym discovery on privacy-aware clinical data, a new yet practical setting that consumes less sensitive information to discover synonyms. We propose a novel and effective framework named SURFCON that considers both the surface form information and the global context information, can handle both InV and OOV query terms, and substantially outperforms various baselines on real-world datasets. As future work, we will extend SURFCON to infer more semantic relationships (besides synonymity) between terms and test it on more real-life datasets.

SURFCON is a deep learning-based model built on top of feedforward neural networks under a ranking framework. It is characterized by the context prediction module containing many output dimensions with the softmax, the dynamic matching mechanism that injects the attention-based aggregation into our model design, and the ListNet-based (Cao et al.,

2007) ranking method that dynamically samples negative candidates for efficient training. Alternative architecture for context prediction may consider leveraging other approximation methods to predict large-scale contexts, such as hierarchical softmax or negative sampling (Mikolov et al., 2013b). Dynamic matching captures the relationships between contexts based on 1-hop relations, and alternative design may consider high-order contexts to capture more meaningful interactions. Last but not least, while listwise ranking methods, such as ListNet, enhance the training efficiency greatly, other ranking methods, such as pointwise and pairwise ranking, can also be explored, potentially improving the performance with more fine-grained ranking signals.

Part III: Representation: Learning Neural Structural Representation

Chapter 3: Modeling Context Pair Interaction for Pairwise Tasks on Graphs

Following knowledge acquisition, the next step in the life cycle of incorporating human knowledge into AI systems is knowledge representation to transform various forms of human knowledge into neural representations. This step is crucial because better neural representation could capture more advanced knowledge properties to be more effectively and efficiently fused into AI systems. In this dissertation, we focus on learning better neural representations for structured knowledge from the previous step of knowledge acquisition. Specifically, in this chapter, we focus on learning better graph representations for pairwise prediction tasks, such as link prediction, in homogeneous graphs and study how to represent node pairs for better predictions better.

Specifically, predicting pairwise relationships between nodes in graphs is a fundamental task in data mining with many real-world applications, such as link prediction on social networks, relation prediction on knowledge graphs, etc. A dominating methodology is to first use advanced graph representation methods to learn generic node representations and then build a pairwise prediction classifier with the target nodes' vectors concatenated as input. However, such methods suffer from low interpretability, as it is difficult to explain why certain relationships are predicted only based on their prediction scores. In this paper, we propose to model the pairwise interactions between neighboring nodes (i.e.,

contexts) of target pairs. The new formulation enables us to build more appropriate representations for node pairs and gain better model interpretability (by highlighting meaningful interactions). To this end, we introduce a unified framework with two general perspectives, node-centric and pair-centric, about how to model context pair interactions. We also propose a novel pair-centric context interaction model and a new pre-trained embedding, which represents the pair semantics and shows many attractive properties. We test our models on two common pairwise prediction tasks: link prediction task and relation prediction task, and compare them with graph feature-based, embedding-based, and Graph Neural Network (GNN)-based baselines. Our experimental results show the superior performance of the pre-trained pair embeddings and that the pair-centric interaction model outperforms all baselines by a large margin. The code and data can be found on Github: https://github.com/zhenwang9102/ConPI.

3.1 Introduction

Pairwise prediction, which aims to predict relationships between two nodes in a graph, is a fundamental problem in data mining and machine learning, and has a wide range of practical applications. For example, friend recommendation in social networks (Wang et al., 2015b) recommends potential friends to a user by predicting his/her links with other users. Other examples include multi-relational link prediction in knowledge graphs (Wang et al., 2018c), weakly-supervised relation extraction in entity co-occurrence graphs (Qu et al., 2018), etc.

Given a graph and a pair of target nodes (u, v), many powerful representation learning methods (Perozzi et al., 2014; Tang et al., 2015; Wang et al., 2016b; Grover and Leskovec, 2016a; Ribeiro et al., 2017; Tsitsulin et al., 2018; Kipf and Welling, 2017; Hamilton et al.,

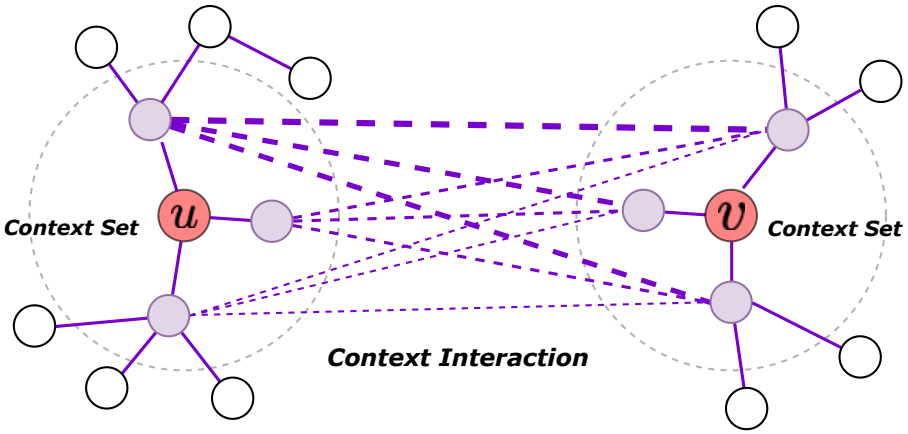

Figure 3.1: Intuition Illustration. To predict the potential relationships between target nodes u and v, we model the pairwise interactions between their context nodes (purple ones within the circle) and aggregate important interaction information (dashed lines) for the final prediction. The thickness of a line reflects how important the interaction is.

2017; Veličković et al., 2018) have been proposed to encode various graph information into node vectors, $\vec{\boldsymbol{\nu}}_u, \vec{\boldsymbol{\nu}}_v$, and predict their relationships based on these vectors. By preserving the graph structure and properties into a low-dimensional latent vector space, these representation learning methods have obtained state-of-the-art results in many pairwise prediction tasks, such as link prediction (Zhang and Chen, 2018), relation prediction (Qu et al., 2018), etc. However, such methods suffer from low transparency and interpretability and it is difficult for users to understand and trust the prediction decisions. This is because, various graph information, e.g., graph structure, context information, is all encoded into a latent vector space *implicitly*, and the pairwise prediction is usually made based on the similarity metrics of node vectors. For example, when predicting whether two users are friends in social networks, users may want to know how the decision is made and the model would be less likely to be convincing to users if the provided explanation is that two user vectors have high cosine similarity.

In this chapter, we propose to *explicitly* model the interactions between neighborhoods[15] of target nodes for the pairwise prediction. Taking a toy example for illustrating our intuition, as shown in Figure 4.1, the interaction links between target nodes could be indicative for predicting that node u is connected with v (e.g., some context nodes of node u are very similar or hold special relations with context nodes of node v). Learning such meaningful patterns explicitly among context interaction links in graphs is expected to shed some light on interpreting the model's decision (by highlighting important interaction links for the prediction). In addition, most of previous methods learn generic embeddings from the graph, which might be suboptimal for pairwise relationship predictions. In contrast, modeling context pair interactions enables us build pair-specific representations to further improve the model performance.

One straightforward way to manipulate the contexts of target nodes for the interaction is to compute heuristic features between their contexts, such as common neighbors, Jaccard similarity (Liben-Nowell and Kleinberg, 2007). Such features provide competitive performance in some pairwise prediction tasks, such as link prediction, and intuitive explanations for justifying the model prediction. However, they are also known for the lack of generalizability (Kovács et al., 2019). For example, though common neighbors feature works reasonably well in link prediction, (Kovács et al., 2019) find that linked proteins do not necessarily share many neighbors in the Protein-Protein Interaction (PPI) network.

Thus, in this chapter, we propose to combine the representation learning with context interaction and take the first step towards explicitly exploring Context Pair Interaction for pairwise prediction with a general deep learning framework, CONPI ("con-π"). Our framework essentially computes an interaction score for the pair of contexts (i.e., pairwise

[15] In this chapter, we use context and neighborhood interchangeably.

interaction) jointly with the pairwise prediction task, as shown in Figure 4.1. We provide two perspectives of aggregating the pair interaction information, node-centric and pair-centric. From the node-centric perspective (referred to as CONPI-node model), the model learns interaction-aware node representations for target nodes whose contexts dynamically influence each other. The final prediction is made based on the simple combination of new node representations as traditional embedding methods do (Grover and Leskovec, 2016a). On the other hand, from a relatively new perspective of pair-centric (referred to as CONPI-pair), we novelly propose to directly model pair representations for each interacting pair and aggregate them together for the final prediction. Such a new perspective further motivates us to propose a new type of pair embedding (which represents the semantics of each context pair) and inject it back into the CONPI-pair model for more efficient learning. Finally, our CONPI framework offers a certain amount of model interpretability that can generate instance-level explanations, i.e., meaningful and important context links, which could be easily understood by users and thus, obtain their trust better.

Our contributions are summarized as follows:

- We highlight the importance of context interactions for pairwise predictions. To the best of our knowledge, we are the first to systematically study how to model context pair interactions in pairwise tasks. Our study aims to inspire more graph-based models to study the interaction mechanisms.

- We propose a general framework CONPI with two perspectives, node-centric and pair-centric, and build a novel model for the pair-centric view that aggregates context pair representations directly for the final prediction. Our framework considers the mutual influence between nodes via context interactions and enhances the interpretability by

highlighting important context pairs that lead to the model decision.

- We also propose a new type of pair embedding on homogeneous graph to capture semantics of any node pairs and inject it back to CONPI-pair model to show its effectiveness.

- We conduct extensive experiments on two types of pairwise prediction tasks, link prediction and relation prediction with totally 6 datasets. In comparison with strong baselines from different categories, our framework can achieve very competitive performance and more importantly, much better interpretability.

3.2 Related Work

Pairwise Tasks on Graphs. Given a graph, there are generally three types of features that can be leveraged effectively for pairwise predictions, heuristic features (e.g., common neighbors), latent features (e.g., node embeddings) and explicit features (e.g., node attributes). In this chapter, as we mainly utilize the graph information for the interaction, we do not focus on modeling explicit features at this moment. For the heuristic features, they can be directly extracted from the graph, which is computationally efficient and could be competitive baselines for some tasks, such as link prediction. But they may lose the generalizability across different datasets (Kovács et al., 2019). On the other hand, latent features, i.e., low-dimensional embedding vectors learned from the graph, can be optimized via various approaches, e.g., matrix factorization (Belkin and Niyogi, 2002; Qiu et al., 2018), random walk (Perozzi et al., 2014; Grover and Leskovec, 2016a), neural networks (Tang et al., 2015; Wang et al., 2016b). However, modeling pairwise prediction in the latent space may suffer several disadvantages as we mentioned previously, such as

lacking a certain amount of interpretability or learning node representation independently from the target pairwise prediction task. Our framework CONPI deals with such issues by explicitly modeling context pair interactions between nodes.

Neighborhood Aggregation. Our framework takes as input neighborhoods from the graph and aggregates them for the pairwise prediction, which is closely related to the line of recent researches in neighborhood aggregation for graph learning. For example, (Hamilton et al., 2017) propose to aggregate features of neighboring nodes by different functions, e.g., Mean, LSTM, and Pooling. (Veličković et al., 2018) leverage the multi-head self-attention mechanism to aggregate neighborhood information for node representations. (Sun et al., 2020b) introduce gated multi-hop neighborhood aggregation to align nodes from knowledge graphs. All the aforementioned methods inspire us to thoroughly explore neighborhoods for pairwise predictions. Instead of aggregating neighborhoods independently, we propose that we need to consider the interactions between contexts for a better aggregation. The CONPI-node method models the context pair interactions with a similar mechanism as mutual attention module that has been applied in many other tasks (Santos et al., 2016; Tu et al., 2017; Li et al., 2019; Wang et al., 2019b) and our contribution for CONPI-node here is to unify it into a framework for systemically modeling the context pair interactions.

Graph Representation Learning. Representation learning on graphs has been extensively studied and mostly focuses on nodes (Perozzi et al., 2014; Cao et al., 2016; Grover and Leskovec, 2016a; Ribeiro et al., 2017; Wang et al., 2016b), i.e., projecting nodes to a low-dimensional vector space while preserving the graph structure (Wang et al., 2016b; Ribeiro et al., 2017) and properties (Zhang et al., 2018c). The learned embeddings can be reused for a variety of downstream tasks, such as the pairwise prediction tasks as mentioned above. More recently, while being far less mature compared with node embeddings, learning edge

semantics and representing edges beyond nodes have received increasing attention (Abu-El-Haija et al., 2017; Shi et al., 2018; Verma et al., 2019; Zhou et al., 2018; Bandyopadhyay et al., 2019; Park et al., 2019). First of all, there are some recent works trying to learn representations of connected edges in the graph (Zhou et al., 2018; Bandyopadhyay et al., 2019), which cannot be generalized to unobserved pairs of nodes, e.g., the context pairs in our interaction module. To learn embeddings for any pair of nodes, (Abu-El-Haija et al., 2017) model the asymmetry property between two nodes and define an edge function over node vectors to produce a score for the pair rather than a pair vector. In addition, edge embeddings (Shi et al., 2018; Verma et al., 2019) are studied in heterogeneous networks to learn task-relevant representations for specific pairs, e.g., author-paper pairs in citation networks. In this chapter, we aim to pre-train a general-purpose pair embedding on homogeneous graphs and incorporate it into our context pair interaction model for more general pairwise prediction tasks. More relevantly, (Joshi et al., 2019) propose to learn word pair embeddings from a large text corpus to incorporate text patterns into the pair embeddings while our CONPI-pair model (Section 3.4.3) focuses on learning node pair embeddings from the graph structure.

3.3 Preliminaries

In this section, we first introduce our problem definition and, then describe how we can obtain contexts for our framework.

3.3.1 Task Definition

Notations. We denote an undirected graph as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $\mathcal{V}$ is the set of vertices and $\mathcal{E}$ is the the set of existing edges. For any node $x \in \mathcal{V}$, we denote its neighbor[16] set as $\mathcal{N}(x)$.

Problem Definition. In this chapter, we consider the pairwise prediction tasks on graphs with a generalized definition in which the label to predict between nodes can by any pairwise supervisions, e.g., link label, relation label. Specifically, given the graph $\mathcal{G}$ and target pairs $\mathcal{S}$ consisting of a few positive training labels $\mathcal{S} = \{(u_i, v_i)\}_{i=1}^{N}, v, u \in \mathcal{V}$. We aim to leverage the training pairs to make prediction for unseen pairs, $P(y|(u'_j, v'_j))$ where y indicates whether the pair holding the target label. We formally define the pairwise prediction as follows:

Definition 3.3.1. (Pairwise Prediction). Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and a set of target pairs $\mathcal{S} = \{(u_i, v_i)\}_{i=1}^{N}$, we aim to estimate the possibility of a new node pair holding the target label, $P(y|(u'_j, v'_j))$ and make a binary prediction for the new pair[17].

3.3.2 Context Acquisition.

As our framework explicitly utilizes contexts for the pairwise prediction, we always assume the neighbor set of each node is given for the rest of the chapter. As the background, we give a brief introduction about how we can acquire contexts in order to apply our framework. For the simplest case, we can uniformly sample a fix-number of neighbors (Hamilton et al., 2017; Veličković et al., 2018). For example, for the neighbor set $\mathcal{N}(u)$, we sample from the set $\{v \in \mathcal{V}, (u, v) \in \mathcal{E}\}$.

[16] We consider 1-hop neighbor in this chapter.

[17] Note that our task can be easily extended to multi-class setting.

Moreover, in some graphs, the degree distribution has a long tail in which for some nodes, we may not able to sample enough contexts or even zero contexts (i.e., out-of-vocabulary issue). Under such circumstances, we can leverage some context prediction methods to recover sufficient high-potential contexts for our following context interaction framework (Hu* et al., 2020; Wang et al., 2019b). Instead of recovering the context graph by GNNs (Hu* et al., 2020), we need discrete contexts for our framework. That is, we can estimate a context conditional probability $p(c|u)$ representing the likelihood that context node c is connected with node u:

$$p(c|u) = \frac{\exp(\boldsymbol{\nu}_c^T \cdot \text{Encoder}(u))}{\sum_{i=1}^{|\mathcal{V}|} \exp(\boldsymbol{\nu}_{c_i}^T \cdot \text{Encoder}(u))} \tag{3.1}$$

where $\boldsymbol{\nu}_c$ is the context embedding for node c and $\text{Encoder}(\cdot)$ is used to encode the node features into a vector, for instance, we can adopt Recurrent Neural Networks (RNNs) to encode the article content from citation networks into a single vector. There are multiple options to optimize such probability (Pennington et al., 2014; Tang et al., 2015). For example, we can employ the cross-entropy loss to minimize the distance between the estimated probability with the empirical probability from the original graph. We refer to (Tang et al., 2015) for more theoretical details. After pre-training such a conditional probability on the graph, given a node v, we can select the top-L_v entities from $p(\cdot|v)$ as v's neighbors for the subsequent modeling.

3.4 CONPI Framework

In this section, we introduce our CONPI framework for modeling context pair interactions in pairwise tasks. We first give an overview and then introduce CONPI-node and CONPI-pair models as well as how to pre-train and utilize pair embeddings.

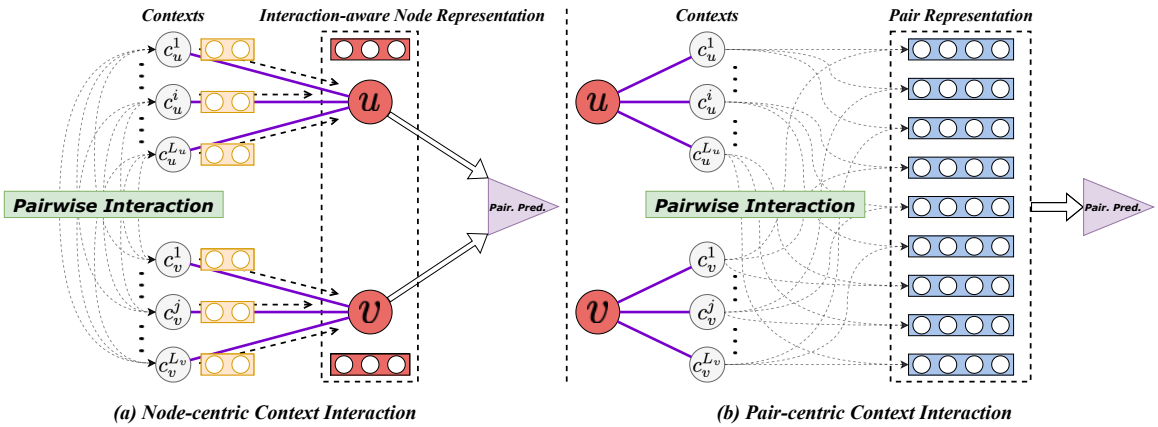

Figure 3.2: Framework Overview: (a) CONPI-node; (b) CONPI-pair.

3.4.1 Framework Overview

Current state-of-the-art pairwise prediction models usually focus on learning high-quality node representations with advanced node embedding methods (Grover and Leskovec, 2016a; Wang et al., 2016b; Ribeiro et al., 2017) or graph neural networks (Hamilton et al., 2017; Veličković et al., 2018; Kipf and Welling, 2017), which perform reasonably well across different tasks. However, there are two challenges that our framework tries to solve. Firstly, though preserving rich graph information, the learned latent representation is hard to be comprehended by humans, and thus, has a lack of interpretability. Secondly, when learning node representations, most models project each node to a latent vector space independently by preserving as much graph information as possible, but ignore the mutual influence between the pair when applying the learned vectors for the pairwise prediction task.

To solve the challenges as mentioned above, our proposed framework CONPI, as shown in Figure 4.2, presents two different perspectives for conducting context pair interactions

for pairwise prediction. For the node-centric context interaction (Figure 4.2 (a)), interaction-aware node representations are firstly learned by aggregating context information with the consideration of the pairwise interaction. By doing so, node representations of target nodes will mutually affect each other for a better pairwise prediction. Moreover, instead of learning node representations, we model the context pairs directly for the pairwise prediction (Figure 4.2 (b)). Specifically, for each context pair, we formulate a pair representation (either training a pair encoder function taking as input the node representations or applying pre-trained pair embeddings) and then aggregate them for the pairwise prediction. In next parts, we will introduce model details and show how to obtain interpretable results for pairwise prediction by our framework.

3.4.2 Node-centric Context Interaction

The goal of the node-centric context interaction model (CONPI-node) is to first form an interaction-aware representation for each target node and make the pairwise prediction based on the new representations. Inspired by previous graph neighborhood aggregation methods that aggregate contexts with a self-attention layer (Veličković et al., 2018), we propose to aggregate the context information for each node in such a way that each context node is weighed based on its interaction links with contexts on the other side. Thereby each node representations in the pairwise prediction task would influence each other via the interaction links between their contexts.

Formally, as shown in Figure 4.2 (a), given two target nodes in a graph, $(u, v) \in V$, we obtain their neighbor set, $\mathcal{N}(u) = \{c_u^1, ..., c_u^{L_u}\}$ and $\mathcal{N}(v) = \{c_v^1, ..., c_v^{L_v}\}$ as mentioned in Section 3.3.2. Without losing the generalizability, we assign a feature vector to each

context node and we get two sets of vectors, $\mathbf{\Psi}_u = \{\boldsymbol{\nu}_u^i\}_{i=1}^{L_u}, \mathbf{\Psi}_v = \{\boldsymbol{\nu}_v^j\}_{j=1}^{L_v}$ for target nodes u, v, as the input for CONPI-node model.

There are several neighborhood aggregation mechanisms that have been tried for mapping a set of context vectors to a node representation, e.g., Mean, LSTM, Pooling (Hamilton et al., 2017) or Multi-head Self-attention (Veličković et al., 2018). However, none of them consider the factor of context interactions with other nodes for the pairwise prediction when aggregating the contexts. Thus, we leverage the attention-based approach (Santos et al., 2016; Tu et al., 2017) to weigh each context with the consideration of its interactions with contexts on the other side and aggregate the weighed contexts for target node u as follows:

$$\boldsymbol{h}_u = \sum_{i=1}^{L_u} \alpha_u^i \cdot \boldsymbol{\nu}_u^i \tag{3.2}$$

$$\alpha_u^i = \frac{\exp(S_{\text{pool}}(\boldsymbol{\nu}_u^i, \mathbf{\Psi}_v))}{\sum_{k=1}^{L_u} \exp(S_{\text{pool}}(\boldsymbol{\nu}_u^k, \mathbf{\Psi}_v))} \tag{3.3}$$

$$S_{\text{pool}}(\boldsymbol{\nu}_u^i, \mathbf{\Psi}_v) = \text{Pool}\left(\left\{\eta(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j), \forall \boldsymbol{\nu}_v^j \in \mathbf{\Psi}_v\right\}\right) \tag{3.4}$$

$$\eta(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j) = \sigma\left({\boldsymbol{\nu}_u^i}^T \cdot \mathbf{W}_{\text{pool}} \cdot \boldsymbol{\nu}_v^j\right) \tag{3.5}$$

where $\mathbf{W}_{\text{pool}}$ is the weight matrix for the pooling function. S_{pool} is a pooling function (e.g., mean or max function) to measure the interaction score between one context vector with a set of context vectors in which the (pairwise) interaction between each pair of contexts is measured by a non-linearity function η (Eqn.3.5).

To this end, the representation, $\boldsymbol{h}_u$, for target node u incorporates the interaction information from the other side, and with a similar operation in the opposite direction, we can obtain the interaction-aware representation for node v as $\boldsymbol{h}_v$. Finally, the pairwise prediction can be made by applying a classifier on a binary operator for $\boldsymbol{h}_u, \boldsymbol{h}_v$ as what the previous works do (Grover and Leskovec, 2016a; Epasto and Perozzi, 2019).

Interpretability. CONPI-node model generates explanations for model decisions in a two-steps process as follows: after the prediction is made, we first trace back to the most important contexts for each target nodes by attention weights, $\alpha_u^i, i \in [1, L_u]$. Then, for each context node c_u^i, we further trace back to its pair interaction scores, $\eta(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j), j \in [1, L_v]$ to obtain the most influential interaction pairs, $\{(c_u^i, c_v^j), \eta(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j) > \epsilon\}$ as the explanations where ϵ is a pre-defined threshold.

3.4.3 Pair-centric Context Interaction

In contrast to the *node-centric context interaction*, as shown in Figure 3.4 (b), we also propose a new approach of the pair-centric perspective for modeling the pairwise prediction in graphs by the *pair-centric context interaction*. Instead of aggregating context features for each target node individually, CONPI-pair model directly works on context pairs for the final prediction. Our motivation is that to infer the pairwise relationships between two nodes, we are encouraged to model the pairwise context interactions by the nature of the task. In other words, the relationships between contexts could be more indicative for predicting the relationships for their target nodes, as shown in Figure 4.1 in the Introduction. Next, we introduce the details of our CONPI-pair model.

Same as CONPI-node model, we have two neighbor sets, $\boldsymbol{\Psi}_u$ and $\boldsymbol{\Psi}_v$. As shown in Figure 4.2 (b), the essential idea of the CONPI-pair model is to formulate a representation

for each pair of context nodes, and them aggregate all pair for the final prediction. For simplicity, we encode the pair representation as a compositional function for its given two node vectors, $(\boldsymbol{\nu}_i, \boldsymbol{\nu}_j)$ as:

$$g_p(\boldsymbol{\nu}_i, \boldsymbol{\nu}_j) = f_p([\boldsymbol{\nu}_i; \boldsymbol{\nu}_j; \boldsymbol{\nu}_i \oplus \boldsymbol{\nu}_j]) \tag{3.6}$$

where $\oplus$ is the element-wise product for two vectors, $[;]$ represents vector concatenation. $f_p(\cdot)$ is a fully-connected network that produces a low-dimensional vector for representing the pair. Such a pair representation will help capture the relationship for a pair of nodes during the pairwise prediction task. Next, we aggregate all pairs with an attention-based mechanism for the final prediction:

$$\boldsymbol{z} = \sum_{i}^{L_u} \sum_{j}^{L_v} \beta_{ij} \cdot g_p(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j) \tag{3.7}$$

$$\beta_{ij} = \frac{\exp(S_p(\boldsymbol{\nu}_u^i, \boldsymbol{\nu}_v^j))}{\sum_{m=1}^{L_u} \sum_{n=1}^{L_v} \exp(S_p(\boldsymbol{\nu}_u^m, \boldsymbol{\nu}_v^n))} \tag{3.8}$$

where S_p is a similarity function for each pair of context vectors (e.g., bilinear similarity). $\boldsymbol{z}$ will be used for making the final prediction with a classification layer.

Interpretability. The attention distribution, β_{ij}, in Eqn.3.8 is the normalized pairwise interaction scores and will be used to retrieve the most important pairs for explanations after the prediction. And, the explanation question that the CONPI-pair model can answer is which important context interactions lead to current prediction. Apparently, for interpretability, CONPI-pair selects context pairs in a one-step process, which is more straightforward and easier to be understood by users than CONPI-node model.

Though the model architecture for our CONPI-pair is relatively simple, the perspective of modeling pairs for pairwise prediction is new, and we also show the superior performance of CONPI-pair model with comprehensive experiments later. As one of our main contributions, shifting the perspective from node-centric to pair-centric model provides us more promising directions to go for modeling pairwise prediction tasks. Next, we show a new type of pair embeddings, which can be naturally injected into CONPI-pair.

3.4.4 Pretraining and Injecting Pair Embedding

To expand the idea of modeling pair interactions directly for pairwise prediction tasks, we further propose to leverage the graph structure to pre-train embeddings of node pairs, and inject them back to our CONPI-pair model with fewer parameters. Our intuition is that the pre-trained pair embedding would incorporate more prior knowledge for node pairs than combining their node vectors straightforwardly as in previous CONPI-pair model. In this part, we introduce learning general-purpose embeddings for node pairs based on the graph structure to facilitate context pair interactions.

Pair Representation Learning. Following the distributional hypothesis (Perozzi et al., 2014), we encourage the pair embeddings of two nodes to be similar if they are likely to co-occur with similar context nodes. Given a pair of node (u, v) and a context node c, we first embed them to vectors, $\boldsymbol{\nu}_u, \boldsymbol{\nu}_v, \boldsymbol{\nu}_c$ via two embedding matrix, $\boldsymbol{E}_p$ for u, v, $\boldsymbol{E}_c$ for c. Note that the node pairs are not necessarily to be connected in the graph, and it would be $O(|\mathcal{V}|^2)$ complexity to build embeddings for all pairs, which is computationally expensive. Thus, we define a deep compositional function taking the input as representations for nodes, (u, v) to generate a fixed-length vector for the pair:

$$\mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v) = f_{\text{MLP}}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v, \boldsymbol{\nu}_u \oplus \boldsymbol{\nu}_v) \tag{3.9}$$

where f_{MLP} is a fully-connected multi-layer neural networks with the same input as Eqn.3.6. With such a compositional function, we are able to generate embeddings for any pairs in the graph efficiently.

After encoding the node pair (u, v) and its context nodes c, we can define the following conditional probability representing how likely the context is observed around the pair (Tang et al., 2015) and optimize it with a negative log-likelihood objective function:

$$p(c|(u, v)) = \frac{\exp(\boldsymbol{\nu}_c^T \cdot \mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v))}{\sum_{c' \in \mathcal{V}} \exp({\boldsymbol{\nu}_{c'}}^T \cdot \mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v))} \tag{3.10}$$

Minimizing the negative log-likelihood for $p(c|(u, v))$ would be computationally costly, which leads to the adoption of popular negative sampling technique (Mikolov et al., 2013b) for efficient training. The goal of negative sampling is to encourage the similarity between the pair and context if they appear together (co-occur) in the graph, and the dissimilarity between the pair and randomly sampled contexts. That is, for a valid pair-context sample, we fix the pair (u, v) and randomly sample contexts c^N as the distractor.

However, pair embeddings are different from node embeddings in which a pair consists of two nodes that are both changeable. In other words, we are able to fix the left pair-node u and context c, sample the right pair-node v^N, and vice versa. By doing so, we can expose the pair embedding to noisier environments to make it more robust. Therefore, we introduce the objective function of the negative sampling for training our pair embeddings as follows:

$$\mathcal{L}_{\text{pair}} = \log \sigma(\boldsymbol{\nu}_c^T \cdot \mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v)) + \sum_{i=1}^{K_c} \log \sigma(-\boldsymbol{\nu}_{c_i}^{N^T} \cdot \mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_v)) + \sum_{j=1}^{K_u} \log \sigma(-\boldsymbol{\nu}_c^T \cdot \mathcal{P}(\boldsymbol{\nu}_{u_j}^N, \boldsymbol{\nu}_v)) + \sum_{k=1}^{K_v} \log \sigma(-\boldsymbol{\nu}_c^T \cdot \mathcal{P}(\boldsymbol{\nu}_u, \boldsymbol{\nu}_{v_k}^N)) \tag{3.11}$$

where K_c, K_u, K_v are the number of random samples for contexts, left pair-nodes, right pair-nodes, and σ here is the sigmoid function.

Our objective function is similar to the multivariate objective function in pair2vec (Joshi et al., 2019), but differs in that we optimize node pair embeddings with graph structure and represent contexts as nodes, while they try to encode a short span of words as the context to learn word pair embeddings from a text corpus.

Pair Representation Injection. Once the node pair embeddings are learned, we are ready to inject them back into our CONPI-pair model for the pairwise prediction task. We keep the injection as simple and generalizable as possible to show the effectiveness of our pair embeddings. We simply replace the pair encoder function $g_p(\boldsymbol{\nu}_i, \boldsymbol{\nu}_j)$ in Eqn.3.6 by pre-trained pair embedding vectors $\mathcal{P}(\boldsymbol{\nu}_i, \boldsymbol{\nu}_j)$ in Eqn.3.9 and keep the rest parts as the same in CONPI-pair model. With such a simple injection, we do not need the fully-connected network in the pair encoder (Eqn.3.6), and thus, reduce the number of parameters compared with CONPI-pair model. Note that although we can produce a pair embedding for the pair of target nodes and use it for the final prediction, directly making a prediction based on the pair embedding does not involve the context interaction, which is not the focus of this chapter.

3.4.5 Model Optimization

Pairwise Prediction Task. In this chapter, we consider a setting of binary pairwise prediction to test our framework for simplicity and define a binary cross entropy loss as follows:

$$\begin{aligned}\mathcal{L}_{pred} = -\sum_{i=1}^{M}(y_i \cdot \log(p(r|u_i, v_i)) \\ + (1 - y_i) \cdot \log(1 - p(r|u_i, v_i)))\end{aligned} \tag{3.12}$$

where M is the total number of samples, y_i is the ground-truth label indicating whether u_i, v_i holds a certain relation. $p(r|u_i, v_i)$ is a binary classifier with the sigmoid function with the input as interaction-aware node representations in CONPI-node or aggregated pair representations in CONPI-pair.

Complexity Analysis. Our pairwise context interaction takes $\mathcal{O}(L^2)$ computations (L is the maximal size of contexts), which could be costly when L is very large. In our preliminary experiments, we observe that the performance does not further improve when we set L at a reasonably large number (e.g., 100). Thus, we uniformly sample a limited-size set of contexts for each node to perform efficient interactions. For future work, we could select the contexts more smartly, e.g., adopting some graph sparsification techniques (Zheng et al., 2020), to further decrease the complexity without significantly sacrificing the performance.

Training Pair Embeddings. To optimize the objective function (Eqn.3.11) for pair embeddings with negative sampling, we need to obtain positive samples and randomly sample negatives ones. Our training algorithm adopts the random walks to provide contexts for pairs of nodes. Specifically, given a sequence of nodes generated by random walks (Perozzi et al., 2014; Grover and Leskovec, 2016a), we define a pair window to sample node pairs and then, for each pair, we define a context window to the left and right of the pair as well as all nodes in the middle of it to sample positive contexts. We randomly sample contexts, left pair-nodes, right pair-nodes for training the Eqn.3.11.

3.5 Experiments

In this section, to show the generalizability of our framework, we test CONPI on two types of pairwise prediction tasks , link prediction and (weakly-supervised) relation prediction.

3.5.1 Link Prediction Task

The link prediction task is a well-studied task in graph domain, which is to predict whether two nodes in a graph have a link. It has many meaningful applications in various kinds of graphs, e.g., predicting friendship links in social networks (Liben-Nowell and Kleinberg, 2007), predicting author identification in citation networks (Park et al., 2019), etc.

Datasets

We collect a number of commonly-used publicly-available real-world graphs for link prediction shown in Table 3.1.

- **PPI**: A Protein-Protein Interaction (PPI) graph for Home Sapiens that is used in (Grover and Leskovec, 2016a). Edges represent the interaction relationships between proteins.
- **Pubmed**: A citation network for papers from PubMed used in (Kipf and Welling, 2016). Edges represent the citation relationship between publications.
- **BlogCatalog**: A social network for bloggers from the BlogCatalog website that is used on (Grover and Leskovec, 2016a). Edges represent friendship links between bloggers.
- **DrugBank DDI**: A Drug-Drug Interaction graph crawled from DrugBank that is used in (Yue et al., 2019). Edges represent the interaction relationships between drugs.

Experimental Setup

To evaluate all methods fairly, we follow the experiment strategy that is commonly used by previous methods (Grover and Leskovec, 2016a; Zhang and Chen, 2018; Epasto and Perozzi, 2019). We first randomly split all links of the original graph into training, validation, and testing sets with the requirement of keeping the graph in training set connected. The training graph is used for extracting features, training embeddings, and obtaining context sets. For prediction tasks, we also randomly sample the equal number of non-existent links as negative samples. We split all edges to train/validation/test sets with a ratio of 70/15/15.

Compared Methods

We compare our models with the following three types of baselines for the link prediction tasks.

Graph feature-based methods. We test the traditional graph feature-based methods for link prediction tasks (Adamic and Adar, 2003; Liben-Nowell and Kleinberg, 2007; Zhang and Chen, 2018), which calculate some heuristics based on the neighborhood of nodes in the graph. We consider: *Common Neighbors* that calculates the number of shared neighborhoods, *Jaccard Coefficient* that measures the Jaccard similarity between two neighbor sets, and *Adamic Adar* for the number of shared links between two nodes.

Embedding-based methods. We consider several state-of-the-art embedding-based methods, which take the training graph as input and produce an embedding for every node in the graph: *Laplacian* eigenmaps (Belkin and Niyogi, 2002) is a matrix factorization method that factorizes the graph Laplacian matrix to the lowest eigenvectors as embeddings; *DeepWalk* (Perozzi et al., 2014) is a random walk-based method that learns node embeddings with the skip-gram algorithm on random walks generated from the graph. *LINE* (Tang et al.,

Dataset	# Nodes	# Edges	Avg. # Degrees
PPI	3,890	38,739	19.9172
Pubmed	19,717	44,327	4.4963
BlogCatalog	10,312	333,983	64.7756
DrugBank DDI	2,191	242,027	220.9283

Table 3.1: Statistics of datasets for link prediction task.

2015) utilizes the first and second order proximity to optimize node embeddings with edge sampling. *node2vec* (Grover and Leskovec, 2016a) learns the embedding by performing biased random walks on the training graph by minimizing the skip-gram objective.

Graph Neural Networks-based methods. We compare against a popular GNN-based method, *GAE* (Kipf and Welling, 2016), which uses graph auto-encoders model with graph convolutional network encoder to learn efficient node embeddings from the adjacent matrix.

In the testing phase, for all embedding and GNN based baselines, we follow the procedure in previous work (Grover and Leskovec, 2016a; Zhang and Chen, 2018) to learn a classifier based on positive training samples and an equal number of negative training samples, and test the classifier on the testing set. We train a binary logistic regression model on the Hadamard product of node embeddings, using the scikit-learn library (Pedregosa et al., 2011).

Variants of our CONPI framework. CONPI-*node* model makes the prediction based on the interaction-aware node representations of target nodes. CONPI-pair model makes the prediction based on the aggregated pair representations with an attention-based module. CONPI-*pair-emb* is one variant of CONPI-pair model that simply replaces the pair encoder function with pre-trained pair embedding and has fewer parameters than CONPI-pair.

Note that in this chapter, we focus on modeling context interaction based on graph structure, and thus, we mainly consider baseline methods that leverage graph structure for a fair comparison, e.g., we do not feed node attribute features in the GAE model.

Combination with more advanced GNN methods. We compare CONPI mainly with various embedding-based methods to show its effectiveness by adopting node embeddings (e.g., LINE) as feature vectors. In fact, our framework can be built on top of any graph representation methods, including those more complex GNN methods, such as GCN(Kipf and Welling, 2017), GAT(Veličković et al., 2018), HGCN(Chami et al., 2019). Specifically, we can replace current feature vectors by these advanced GNN methods as more powerful feature encoders for our CONPI framework. We do not include them as baselines in this chapter as they suffer from the same low interpretability problem as other baselines and it is suffice for us to demonstrate CONPI's interpretability using simpler embedding methods. We leave the exploration of combining our framework with more complex GNN methods to future work.

Implementation Details.

For the pairwise prediction task, we implement our framework in Pytorch with Adam optimizer. The dimension of node embeddings for all methods is set to 128. For model details, we use mean pooling in CONPI-node model (Eqn. 3.4). We adapt bilinear similarity for measuring the context pair interaction score in CONPI-pair model. The maximum number of contexts is set to 100. We adapt the embedding features (e.g., LINE embeddings) as the feature vector for each context node. Early stopping is conducted in validation set when its performance does not increase for 10 epochs. For CONPI-node model, we adopt the Hadarmard product of the interaction-aware node representations as the feature for a logistic regression classifier.

Method	PPI			Pubmed			BlogCatalog			DrugBank DDI		
	AUC	AP	F1	AUC	AP	F1	AUC	AP	F1	AUC	AP	F1
Common Neighbors	0.8290	0.8250	-	0.6317	0.6313	-	0.9396	0.9357	-	0.9440	0.9451	-
Jaccard Coefficient	0.8099	0.7835	-	0.6316	0.6292	-	0.7748	0.7130	-	0.9299	0.9043	-
Adamic Adar	0.8325	0.8391	-	0.6318	0.6317	-	0.9444	0.9456	-	0.9459	0.9478	-
Laplacian (Belkin and Niyogi, 2002)	0.5355	0.5742	0.3269	0.6976	0.7657	0.5616	0.7157	0.7711	0.6938	0.6828	0.7862	0.6590
DeepWalk (Perozzi et al., 2014)	0.7919	0.8079	0.4630	0.9131	0.9323	0.6643	0.8051	0.8007	0.6909	0.9235	0.9117	0.8377
LINE (Tang et al., 2015)	0.8153	0.8364	0.7205	0.8107	0.8318	0.6996	0.9153	0.9143	0.8382	0.9222	0.9169	0.8449
node2vec (Grover and Leskovec, 2016a)	0.7513	0.7625	0.4116	0.9230	0.9343	0.6572	0.6400	0.5873	0.5527	0.8949	0.8884	0.8030
GAE (Kipf and Welling, 2016)	0.7056	0.6029	0.7203	0.7904	0.8250	0.7162	0.7547	0.6496	0.6968	0.7516	0.7170	0.7500
ConPI-node	0.7970	0.7748	0.7332	0.8550	0.8140	0.7439	0.9419	0.9307	0.8745	0.9395	0.9334	0.8641
ConPI-pair-emb	0.8450	0.8291	0.7009	0.8736	0.8987	0.7542	0.9388	0.9177	0.8449	0.9663	0.9644	0.9003
ConPI-pair	**0.9004**	**0.8986**	**0.8208**	**0.9375**	**0.9362**	**0.8437**	**0.9684**	**0.9658**	**0.9117**	**0.9842**	**0.9823**	**0.9364**

Table 3.2: Results on Link Prediction Task

For baseline methods, we adopt existing tookits for the embedding methods. We use OpenNE[18] for training Laplacian, DeepWalk, LINE, node2vec, GAE[19] for training GAE model. For graph features, we employ NetworkX package to compute heuristic features.

For pre-training pair embeddings, we adopt the algorithm from DeepWalk (Perozzi et al., 2014) to generate random walks. And we define the random walks with the number of walks as 32, walk length as 64 for Pubmed, PPI, Drugbank DDI dataset, and the number of walks as 16, walk length as 32 for BlogCatalog and medical term-term co-occurrence graph as these two have a larger size of nodes and edges. For obtaining the positive pair-context pairs, we consider a pair window as 5, and the both of left and right context window to 3. We randomly generate 5 negative samples for the context, 5 negative samples for each node in the pair, which lead to a total 15 negative samples per positive sample. We train our pair embeddings on each dataset with 5 epochs. The dimension of the context nodes is set to 128.

[18]https://github.com/thunlp/OpenNE

[19]https://github.com/tkipf/gae

Experimental Results and Analysis

The main results for comparing all methods in link prediction task are shown in Table 3.2. For graph feature-based baselines, we report Area Under ROC curve (AUC) and Average Precision (AP) scores on their ranking of positive and negative samples in the testing set. For the rest of all classification methods, we report AUC, AP, and F1 scores in the testing set for their performance comparison.

We first compare with graph feature-based methods. Similar with our CONPI framework, those heuristic features also explicitly utilize neighborhood information but based on deterministic rules. As shown in Table 3.2, such heuristic features obtain very competitive performance for the link prediction task (e.g., Common Neighbors feature gets 0.9396 AUC score in BlogCatalog, 0.9440 AUC score in DrugBank DDI graph.), which shows the necessity of explicitly modeling neighborhood nodes for link prediction. However, heuristic feature baselines do not perform consistently in those four datasets indicating their bad generalizability. We observe that our CONPI-pair model outperforms them in a large margin. This is because CONPI incorporates the semantic representation of context pairs and models their interactions with more parameters.

Then, we compare with the embedding and GNN based methods. The CONPI-node model outperforms all baselines in BlogCatalog, and most of the baselines in other datasets, which indicates that making the pairwise prediction based on two node embeddings learned independently cannot fully capture the pairwise information. Furthermore, we can observe that our CONPI-pair model consistently outperforms all these methods in a relatively large margin, which is the core contribution in this chapter. These results confirm our hypothesis that directly modeling pairwise interactions fits more the nature of pairwise prediction task.

Finally, we also compare variants in the ConPI framework. All models that directly model pair representations outperform the node model that learns (interaction-aware) node representation by a large margin. Note that ConPI-pair-emb model with injecting pre-trained pair embeddings has the same number of parameters with ConPI-node, and it beats the ConPI-node model in most datasets. This shows the effectiveness of our pre-trained pair embeddings, which encourages us to further explore the pair representations in the future. All in all, these results indicate that for pairwise prediction tasks, it is necessary to model the pairwise interaction and directly design pair representations for the prediction.

3.5.2 Relation Prediction Task

Relation prediction is another well-known pairwise prediction task that aims to predict relation labels between entities, and can be leveraged to facilitate many downstream applications dealing with graphs, e.g., knowledge base completion (Wang et al., 2018c), hypernymy detection (Shwartz et al., 2016), synonyms discovery (Qu et al., 2017). In contrast to link prediction task, this task considers the setting where the labels are not connected links from the graph, and the graph is mainly used to offer the distributionally semantic information for the task. Same as the previous setting in the link prediction task, we do not consider other task-specific information (e.g., text patterns), but only take as input the graph structure for evaluating all methods fairly.

Datasets

We consider medical relation prediction tasks that infers relations between medical terms based on a medical term-term co-occurrence graph. We employ a publicly available dataset from (Finlayson et al., 2014) in which medical terms are extracted from 20 million

clinical notes, and the edges are weighted by the co-occurrence counts based on the frequency that two terms co-occur in a temporal bin. We select the 1-day per-bin graph that has the most number of terms/nodes (see more details of the original graph in (Finlayson et al., 2014)). For preprocessing, we convert the co-occurrence counts into PPMI value, subsample the graph to remove meaningless terms (Mikolov et al., 2013b), and filter edges with a PPMI value less than 2. Finally, we obtain a large medical term-term co-occurrence graph with 48,651 nodes and 1,659,249 edges.

Experimental Setup

For relations, we select CLINICALLY ASSOCIATED WITH (CAW) and ISA relations that are two of the most common relations in the dataset. The first one indicates a clinically salient relationship between medical terms, while the second one represents a hierarchical relationship meaning that the first term has a more specific meaning than the second one. To obtain the labels on a large scale, we follow the procedure of *weakly supervision* for relation extraction to automatically retrieve supervisions from knowledge bases (KBs) (Qu et al., 2017, 2018). Specifically, we first collect positive samples between concepts in the KB where each concept has a number of string mentions (terms) as $\mathcal{C}_A = \{t_i\}_{i=1}^m, \mathcal{C}_B = \{t_j\}_{j=1}^n$. Then we obtain the positive labels between terms as $\{(t_i, t_j), t_i \in \mathcal{C}_A, t_j \in \mathcal{C}_B\}$. The term-to-concept mapping is provided by the dataset (Finlayson et al., 2014), and we use UMLS (Unified Medical Language System) as the KB. Finally, we have 132,716 positive samples for CAW relation, 85,283 positive samples for ISA relation, and we sample an equal number of negative samples by randomly pairing one argument of the positive pair with a random term for the classification task. Then we

Method	CAW		ISA	
	AUC	F1	AUC	F1
Common Neighbors	0.5631	-	0.7084	-
Jaccard Coefficient	0.5632	-	0.7109	-
Adamic Adar	0.5634	-	0.7097	-
Laplacian (Belkin and Niyogi, 2002)	0.5547	0.5163	0.5452	0.4896
DeepWalk (Perozzi et al., 2014)	0.7309	0.6512	0.8303	0.7292
word2vec (Mikolov et al., 2013b)	0.6842	0.5855	0.8083	0.6845
LINE (Tang et al., 2015)	0.7426	0.6746	0.8209	0.7336
node2vec (Grover and Leskovec, 2016a)	0.7458	0.6653	0.8443	0.7477
CONPI-node	0.7852	0.7072	0.8434	0.7556
CONPI-pair-emb	0.8177	0.7401	0.8807	0.7917
CONPI-pair	**0.8807**	**0.8085**	**0.8945**	**0.8173**

Table 3.3: Results on Relation Prediction Task

split each dataset into train/validation/test sets with a ratio of 70/15/15. We use the full co-occurrence graph as the input for all methods, which is utilized for training all embedding methods and for extracting the graph features.

Compared Methods

We keep most of baseline methods the same as the link prediction task. We remove the GAE method as it cannot process a huge graph like our co-occurrence graph with the out-of-memory (OOM) issue. We also compare another representative embedding method in NLP domain, *word2vec* (Mikolov et al., 2013b) by conducting SVD over the shifted PPMI co-occurrence matrix (Levy and Goldberg, 2014b).

Results and Analysis

The results of relation prediction task are shown in Table 3.3. We observe similar performance comparison as what we have observed in link prediction task. Additionally,

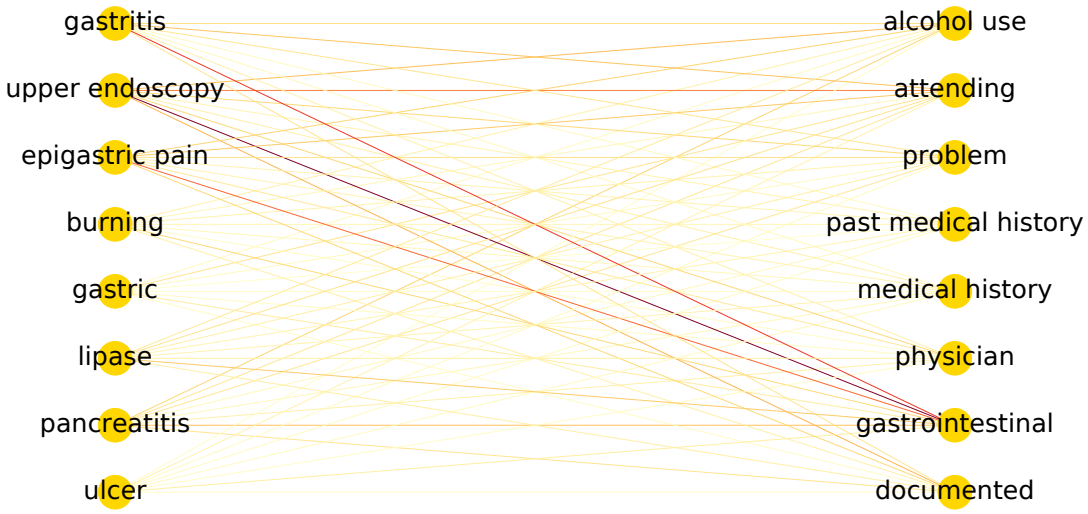

Figure 3.3: Interpretability Visualization for our CONPI-pair model (best view with colors). Contexts (left) of "*burning epigastric pain*" interact with contexts (right) of "*pain epigastric*" to make the pairwise prediction. The line color indicates interaction score (the redder, the larger).

compared with ISA relation, the CAW is relatively difficult to classify as it requires the understanding of the complex semantic of *clinical association*. Even though our CONPI framework beats all baseline methods, and CONPI-pair model obtains the best performance among all methods. Also, in ISA relation, we observe that the CONPI-pair-emb model gets very competitive performance with CONPI model with fewer parameters.

3.5.3 Interpretability Analysis

There are recently-proposed interpretability works on graphs, e.g., GraphLIME (Huang et al., 2020) and GNNExplainer (Ying et al., 2019), which can provide different types of explanations. Most of them try to analyze well-trained graph models post-hoc and instead, our model makes the prediction directly based on the explanations. To demonstrate

CONPI's interpretability for pairwise predictions, we conduct case study by visualizing the interactions scores, i.e., attention weights, calculated between context pairs.

We conduct the case study on the medical term-term co-occurrence graph and choose an easily-understood relation, ISA to interpret. As described in Section 3.4, the model decision is made based on interaction scores for context pairs and we recover such information to explore whether these scores can faithfully explain the model decision. Due to the limitation of the space, we show a testing-set example of our best model, CONPI-pair, for a correct prediction of "*burning epigastric pain*" is a type of "*pain epigastric*" in Figure 3.3. The figure visualizes the interactions between two sets of contexts for target nodes. The nodes on left (right) side are contexts for "*burning epigastric pain*" ("*pain epigastric*"). The color of the line indicates the strength of interaction (the redder, the stronger).

There are two major findings from Figure 3.3. Firstly, by observing the nodes, we can see that our CONPI-pair model nicely downweighs contexts that are irrelevant for the pairwise prediction, such as "problem", "medical history", etc. Secondly, by observing the pairs, the model successfully highlights three pairs, "gastritis", "upper endoscopy", "epigastric pain" with "gastrointestinal", which are strongly relevant with the pairwise prediction and can be treated as explanations. We also see that some interaction scores are not perfect yet, for example, pairs with a meaningless node, "attendig", get relatively high scores. To further enhance the interpretability of our model, we may adopt some post-hoc explainable techniques to further prune some untrustworthy explanations and we leave this to future work. Nevertheless, based on the case study, we can clearly see that our CONPI model can provide informative and faithful explanations for the pairwise prediction, and such interpretability would be helpful to convince users to better trust the model decision, especially on those high-stake domains (e.g., medicine, finance, etc).

3.6 Discussion and Conclusion

In this chapter, we study modeling context pair interactions for pairwise prediction tasks on graphs to better capture the pairwise relationships between nodes and provide a certain amount of interpretability by selecting influential interaction links. We propose a general framework CONPI with node-centric and pair-centric perspectives and a new pair-centric context interaction model, CONPI-pair, to formulate a pair representation and then attentively aggregate all pairs for the final prediction. To capture the pairwise node relationships in embedding space, we propose new pair embeddings in homogeneous graphs and show how to inject them back into the CONPI-pair model. We demonstrate the effectiveness of our framework in two pairwise prediction tasks across a variety of real-world datasets and the model interpretability by the case study.

CONPI is a graph embedding-based method focusing on pairwise predictions on general graphs. It consists of two major components, context interaction design, and pair embedding learning. The context interaction works on 1-hop neighbors for simplicity and inductively captures complex interactions in our node-centric and pair-centric structures. Alternative designs may consider more sophisticated interaction mechanisms to model high-order relationships among contexts. And combining such interaction mechanisms with existing embedding learning methods would also be an inspiring direction. Last but not least, the power of pair embedding can be further extended into a wide range of applications, and the interplay between it with node embedding would be interesting. Moreover, the pair embedding method can be developed to graph neural networks to model complex interactions for node pairs and encode their distance information (Li et al., 2020).

Part IV: Transfer: Transferring Knowledge across Neural Models

Chapter 4: Knowledge Transfer between Structured and Unstructured Knowledge Sources

After collecting and representing human knowledge in neural systems (knowledge acquisition and representation), we now consider more advanced operations of human knowledge, i.e., how to transfer existing knowledge among neural models, which can be seen as an analogy of how humans can generalize the accumulated knowledge to new situations. This step is also essential for AI systems to leverage previously learned knowledge and to become more generalizable and efficient to adapt new samples and save annotation efforts. In this chapter, we study the knowledge transfer between structured and unstructured knowledge sources with the application of complex question answering (CQA). Specifically, CQA or multi-hop question answering combines multiple pieces of evidence to search for the correct answer. Reasoning over a text corpus (TextQA) and/or a knowledge base (KBQA) has been extensively studied and led to distinct system architectures. However, knowledge transfer between such two QA systems has been under-explored. Research questions like what knowledge is transferred or whether the transferred knowledge can help answer one source using another one, are yet to be answered. In this chapter, therefore, we study the knowledge transfer of multi-hop reasoning between structured and unstructured sources. We first propose a unified QA framework named SIMULTQA to enable knowledge transfer and bridge the distinct supervisions from KB and text sources.

Then, we conduct extensive analyses to explore how knowledge is transferred by leveraging the pre-training and fine-tuning paradigm. We focus on the low-resource fine-tuning to show that pre-training SIMULTQA on one source can substantially improve its performance on the other source. More fine-grained analyses on transfer behaviors reveal the types of transferred knowledge and transfer patterns. We conclude with insights into how to construct better QA datasets and systems to exploit knowledge transfer for future work. Code and data are available at https://github.com/Stefan1220/SimultQA.

4.1 Introduction

Structured knowledge source, such as Knowledge Base (KB), and unstructured knowledge sources, such as text corpus, are arguably the most popular sources for complex question answering (CQA). Multi-hop KB-based question answering (KBQA) systems translate questions to logical forms to be executed over a KB for finding answers (Talmor and Berant, 2018; Maheshwari et al., 2019; Lan and Jiang, 2020; Gu et al., 2020; Das et al., 2021; Ye et al., 2021b), while text-based QA (TextQA) systems leverage large text corpora to retrieve paragraphs and extract answer spans for complex questions (Yang et al., 2018; Qi et al., 2019; Asai et al., 2020; Dhingra et al., 2020; Zhu et al., 2021).

However, despite the impressive performance of separate KBQA and TextQA systems, it is not quite clear to the community whether a system trained on one source can be transferred and beneficial to question answering over another source. Inspired by the general transfer learning in NLP by pre-trained language models (PLMs) (Radford et al., 2018; Devlin et al., 2019; Raffel et al., 2020), it is important to study this research problem systematically and thoroughly for the following reasons. First, given the heterogeneity of structured and unstructured sources, it is desirable to build a unified reasoning module to work on both

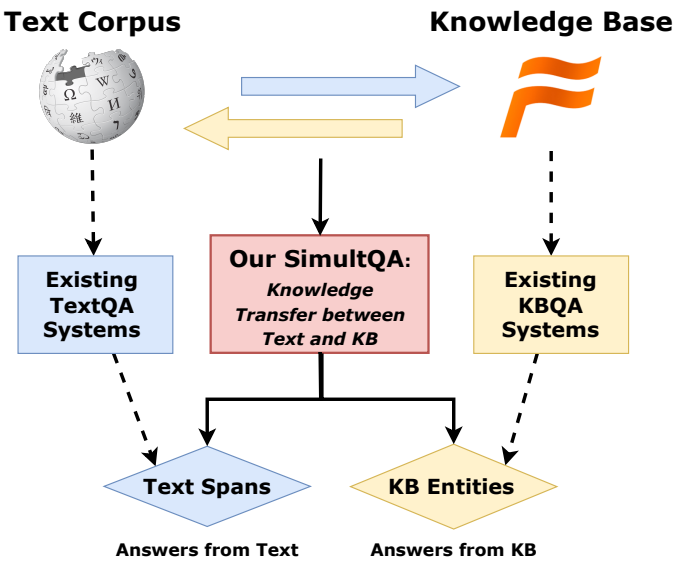

Figure 4.1: To facilitate knowledge transfer between structured and unstructured sources, we develop a unified framework SIMULTQA that can leverage supervision from both sources to answer complex questions.

text and KB and combine different source-specific supervisions. Second, transfer learning has been shown to boost performance on low-resource domains, and it would be practically useful to leverage annotated datasets on one source for CQA on the other source, especially when human annotations are expensive to create new multi-hop QA datasets. Third, it is also critical to investigate what kind of knowledge can be transferred, which can inspire future CQA dataset creation and system design.

One major obstacle in such an investigation for knowledge transfer between structured and unstructured sources is the disparity between them and their specifically designed QA systems as we mentioned earlier. For instance, KB is highly structured and curated where complex query functions can be executed, while text data is unstructured and noisier, leading to quite distinct QA systems. One relevant line of research is HybridQA which tries to leverage multiple sources for QA (Mihaylov and Frank, 2018; Sun et al., 2018, 2019a;

Min et al., 2019; Oguz et al., 2020; Shi et al., 2021). To operate their single model on both KB and text, these methods primarily convert distinct sources into similar data format, e.g., merge text and KB by entity linking, which sacrifices the unique characteristics of each source to some extent and makes it harder to investigate knowledge transfer as sources are entangled together. Thus, typical HybridQA methods are not suitable for studying knowledge transfer problems.

In this chapter, *our first contribution* is proposing a unified CQA framework to enable knowledge transfer between structured and unstructured sources, as shown in Figure 4.1. The proposed framework, SIMULTQA, could perform multi-hop reasoning over text and KB simultaneously by collecting reasoning paths from either text or KB, then reranking paths to select the best one for generating the answer. There are several new and desirable properties of SIMULTQA. First, SIMULTQA unifies the recent advanced KBQA (Luo et al., 2018; Lan and Jiang, 2020) and TextQA (Chen et al., 2017; Asai et al., 2020) systems seamlessly, which preserves their unique strengths maximally to handle various reasoning types. Second, SIMULTQA can utilize distinct supervisions from both sources, which has the potential to combine both KBQA and TextQA datasets for unified training. Last but not least, since SIMULTQA can be applied to any source, we can pre-train it on KB and fine-tune it on the text and vice versa, which makes it easier to quantify the transfer effect brought by the pre-training on a different source. In summary, despite the framework design looking straightforward, we are the first to unify two seemingly distinct CQA systems and study knowledge transfer between two sources for CQA.

With SIMULTQA that enables knowledge transfer, *our second contribution* is to systematically analyze the transfer behavior to help us deeply understand the nature of the

multi-hop reasoning process in KB and Text. We apply our methodology to CWQ (Talmor and Berant, 2018) and HotpotQA (Yang et al., 2018), which are arguably the most popular dataset in KB and text sources, and are representative enough to cover most of the reasoning types on KB and text. We first show that pre-training on one source can consistently improve the fine-tuning performance of the other one in the low-resource setting, indicating future data-hungry QA systems can be boosted by pre-training on another disparate source, especially when human annotation is expensive. More interestingly, further fine-grained analyses attempt to reveal sources of performance gain and find out what knowledge is transferred. We mainly investigate three aspects, reasoning types, reasoning hops, and question similarity. We find that despite KB and text sources being quite disparate, SIMULTQA still finds ways to transfer knowledge by learning a shared semantic space for reasoning and a high-level understanding beyond distinct surface forms of reasoning paths. In addition, we study a more challenging transfer setting where we seek to use text reasoning to answer KB-based questions[20] and vice versa. Promising results are obtained by using text knowledge to help KB questions highlight the expressiveness of the text corpus. We conclude that knowledge transfer between structured and unstructured sources is an intriguing direction to combine the strengths of KBQA and TextQA systems and to use data from one source to boost QA on the other. To the best of our knowledge, this chapter is the first to study knowledge transfer between KB- and text-based CQA in a quantitative and systematic manner.

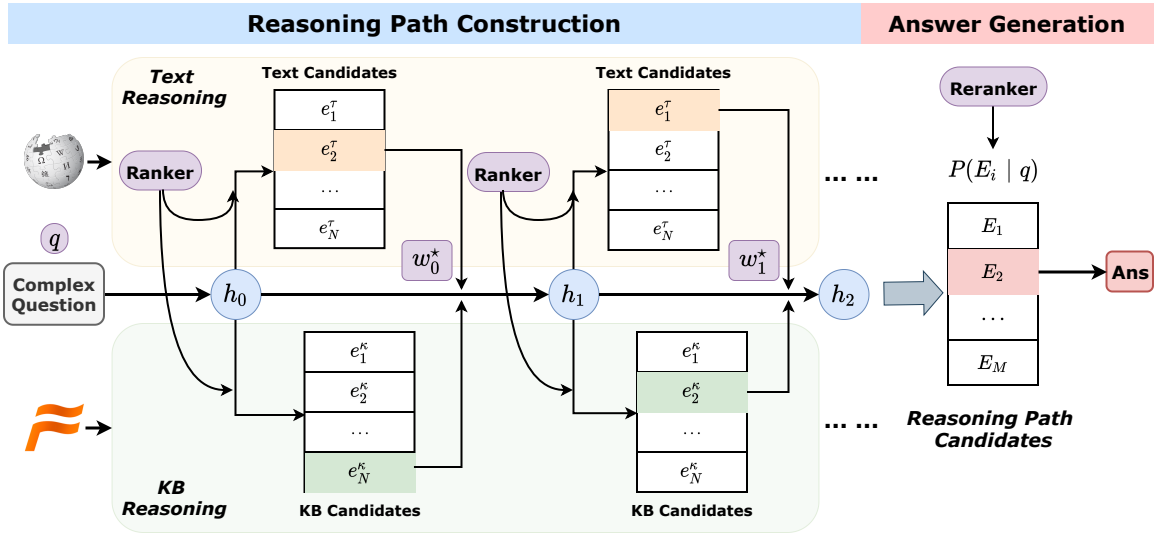

Figure 4.2: Overview of SIMULTQA Framework. There are two stages including constructing reasoning path from either text or KB, and path reranking for the answer generation. In the inference time, the reasoning can be performed simultaneously over text and KB source to find the final answer.

4.2 Related Work

Complex Question Answering. There has been a long history of QA models to answer simple questions (Berant et al., 2013; Rajpurkar et al., 2016; Chen et al., 2017; Wang et al., 2018b; Lee et al., 2018; Yang et al., 2019; Karpukhin et al., 2020). More recent attention has focused on answering complex questions, which requires a multi-hop reasoning process (Yang et al., 2018; Fang et al., 2020). For example, some of them target questions that can be answered using multiple text paragraphs as evidences (Das et al., 2018; Qi et al., 2019; Feldman and El-Yaniv, 2019; Asai et al., 2020), while some existing KBQA works (Bao et al., 2016; Luo et al., 2018; Chen et al., 2019; Lan et al., 2019; Lan and Jiang, 2020) studied how to answer questions by iteratively chaining multiple knowledge base

[20]We refer to questions originally from KBQA/TextQA datasets as KB-based/text-based questions in this chapter.

relations into the evidence path. Our proposed framework unifies these two recent trends of CQA frameworks in text and KB to study knowledge transfer between them.

Hybrid Question Answering. HybridQA is a line of QA research that also studies different knowledge sources (e.g., text articles, Web tables, knowledge bases) for answering questions (Mihaylov and Frank, 2018; Sun et al., 2018, 2019a; Xiong et al., 2019; Min et al., 2019; Oguz et al., 2020; Chen et al., 2020a,b). This line of work typically requires extra human efforts to merge hybrid data for later complex modeling, for example, linking text paragraphs to KB by entity linking or universal schema (Das et al., 2017c; Sun et al., 2018, 2019a) or converting KB edges to plain text (Oguz et al., 2020), which is not needed in SIMULTQA. Their major motivation is to unify data formats for text and KB and construct a more comprehensive knowledge space, which is orthogonal to our motivation of studying knowledge transfer between intact knowledge space of text and KB.

Transfer Learning in NLP. In the last few years, NLP has witnessed the emergence of several transfer learning techniques, and their effectiveness of constantly improving state-of-the-art on a wide range of NLP tasks. Traditional transfer learning techniques (Pan and Yang, 2009) include multi-task learning, domain adaptation, etc (Liu et al., 2019; Clark et al., 2019b; Ruder et al., 2019b). More recently, fine-tuning PLMs has become the de facto standard for transferring knowledge among NLP tasks (Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019; Raffel et al., 2020). In this chapter, we study knowledge transfer between structured and unstructured sources in CQA task and use BERT models as the backbone of our approach.

4.3 SIMULTQA Framework

SIMULTQA is a unified framework for multi-hop reasoning to incorporate both KB and text sources. It consists of two stages, iteratively reasoning and final reranking, which can be trained with supervisions from both sources.

4.3.1 Reasoning Path Construction

CQA requires a multi-hop reasoning process to derive the answer. For KBQA, the reasoning is to traverse the knowledge graph for multi-steps based on generated queries from the question, while for TextQA, it is to collect multiple documents from a text corpus. We consolidate both by iteratively searching for evidence from each source and construct the reasoning path at the end. The key formulation is we treat each step as a ranking problem and train the model to select the most appropriate document/ KB query graph from text corpus/ knowledge graph that can answer the complex question.

Formally, at time step $t, (t \geq 1)$, we are given the complex question q, a pool of candidate evidences, $e_i \in \{e_1, ..., e_N\}$, and the hidden state h_{t-1} from previous step. We first encode them by the BERT [CLS] token representation to get the contextual embedding $\mathbf{w}_i$ for each candidate e_i. Then, we calculate the probability of e_i to be selected in current step by feeding $\mathbf{w}_i$ to a fully-connected layer. We denote text evidence as e_i^τ which is a sequence of tokens from a document in the text corpus. For KB evidence, following previous work (Lan and Jiang, 2020), each candidate is "serialized" into a sequence of relation

tokens and denoted as e_i^κ. The scoring process at t-th step is defined as follows:

$$\mathbf{w}_i^\tau = \text{BERT}_{[\text{CLS}]}([q; e_i^\tau]), \tag{4.1}$$

$$\mathbf{w}_i^\kappa = \text{BERT}_{[\text{CLS}]}([q; e_i^\kappa]), \tag{4.2}$$

$$P_t^\tau(e_i^\tau|q) = \text{FC}(\mathbf{w}_i^\tau, \mathbf{h}_t) \in [0, 1], \tag{4.3}$$

$$P_t^\kappa(e_i^\kappa|q) = \text{FC}(\mathbf{w}_i^\kappa, \mathbf{h}_t) \in [0, 1], \tag{4.4}$$

where $[q; e_i]$ represents the concatenation of the question and evidence separated by [SEP] token. We simply choose a Recurrent Neural Network (RNN), and h_t is calculated to model the sequential multi-hop reasoning process as follows:

$$\mathbf{h}_t = \text{RNN}(\mathbf{h}_{t-1}, \mathbf{w}_{t-1}^*) \in \mathbb{R}^d \tag{4.5}$$

where $\mathbf{w}_{t-1}^*$ encodes the ground-truth evidence in previous step for $t > 1$ during training and $\mathbf{h}_0$ will be a free parameterized vector to be initialized randomly, when $t = 1$. During inference, evidences will be dynamically inferred based on the results from previous step. To encourage knowledge transfer, we share the parameters for the recurrent module and BERT model (as well as the answer generation module that will be introduced later) for KB and text source, which will be jointly optimized. We next introduce how to generate high-quality candidate pools for each step.

Generate Text Candidates. Following previous methods (Chen et al., 2017), for a given complex question and a large text corpus (e.g., Wikipedia), we leverage TF-IDF based methods to retrieve top-K documents with the tri-gram hashing techniques. For the iterative process, we reuse TF-IDF method to retrieve candidates in next step combining the complex question and the previous retrieved document. Moreover, since TF-IDF methods mainly consider the lexical matching, there are several advanced approaches that can be

explored to extend the reasoning path, such as meta-info based (e.g., entity links, hyperlinks (Nie et al., 2019; Asai et al., 2020)), search engine (Qi et al., 2019, 2020), dense retrieval (Xiong et al., 2021). We consider hyperlinks (Asai et al., 2020) in this work and leave more sophisticated methods to future work.

Generate KB Candidates. We follow recent advanced staged query generation methods (Yih et al., 2015; Luo et al., 2018; Lan and Jiang, 2020) to generate candidates and perform KB reasoning. As shown in Figure 4.2, the KB module starts from a grounded entity in the complex question and identifies core relation paths[21] as candidates with necessary constraints. We iteratively generate and rank candidate query graphs in each step based on the topic entity or the entity from the last step.

With the iterative ranking in each step, we can establish the reasoning chain as a sequence of documents, $E^{\tau} = [e_1^{\tau}, ..., e_k^{\tau}]$ for TextQA and a sequence of query graphs, $E^{\kappa} = [e_1^{\kappa}, ..., e_k^{\kappa}]$ for KBQA. We score each path by the multiplication of probability of each selected evidence as $P(e_1|q) \cdot ... \cdot P(e_k|q)$ and use beam search to produce top-M reasoning paths $\{E_1, ..., E_M\}$ for the final answer generation.

4.3.2 Reranking and Answer Generation

Given a complex question q and several reasoning paths $\{E_1, ...E_M\}$ from the previous component, we rerank the paths based on how likely they can answer the question. We use another BERT [CLS] token representation to encode the reasoning path E_i with a fully connected model to output the probability of choosing E_i as follows:

$$\mathbf{v}_i = \text{BERT}_{[\text{CLS}]}([q, \{e_{i1}, ..., e_{ik}\}]), \tag{4.6}$$

$$P(E_i|q) = \text{FC}(\mathbf{v}_i) \in [0, 1] \tag{4.7}$$

[21] As in (Lan and Jiang, 2020), we allow the relation to be a single predicate or two predicates connected through a CVT node designed for a multi-argument relation.

After the reasoning path reranking, our system allows the KB reasoning path and text reasoning path to be handled differently. This reflects the advantage of our system to combine the strength of both KBQA and textQA as discussed earlier. Since KB is structured, we can directly execute the complete query graph in the knowledge graph to get the answers. For question answering with textual evidence chains in particular, another *reader* component is employed to select the text spans that are the final answer based on the top-ranked path.

4.3.3 Training and Inference

We leverage the annotated document labels from HotpotQA dataset to train the reasoning path construction and reranking modules. For CWQ dataset, we split the golden complex logic form into sub-queries by defining the sub-query to be composed of head/tail entities along with one relation or two relations with CVT type node. Constraint relations are also added to the connected sub-queries. The sub-queries are treated as supervisions in each reasoning step as well as the path reranking module. Note that it is now the standard way to train robust CQA systems by leveraging full supervision in each hop. We leave utilizing distantly weak supervisions for training to future work. In each step of reasoning module, the loss functions for KB and text are defined as follows:

$$\begin{aligned} L_t^{\tau} &= -\log P(e_t^{\tau}|q) \\ &\quad - \textstyle\sum_{\widetilde{e^{\tau}} \in C_t^{\tau}} \log(1 - P(\widetilde{e^{\tau}}|q)) \end{aligned} \tag{4.8}$$

$$\begin{aligned} L_t^{\kappa} &= -\log P(e_t^{\kappa}|q) \\ &\quad - \textstyle\sum_{\widetilde{e^{\kappa}} \in C_t^{\kappa}} \log(1 - P(\widetilde{e^{\kappa}}|q)) \end{aligned} \tag{4.9}$$

where C_t^{τ} and C_t^{κ} are negative samples. For text, we follow previous work (Asai et al., 2020) to generate lexically and semantically similar negative samples based on TF-IDF as well as hyperlinks. For KB, we treat all query graphs other than the golden one in the same step as negative samples.

In terms of reranking reasoning paths for KB and text, we reuse the previous supervisions to train a ranker model (Eqn. 7) for selecting the correct path with the loss function as follows:

$$L^{\tau}_{\text{rank}} = -\sum_i y^{\tau}_i \cdot \log(P(E^{\tau}_i|q)) \tag{4.10}$$

$$L^{\kappa}_{\text{rank}} = -\sum_i y^{\kappa}_i \cdot \log(P(E^{\kappa}_i|q)) \tag{4.11}$$

where y^{τ}_i and y^{κ}_i are the assigned labels for the golden path of i-th sample from two sources. We also design negative samples for reasoning paths by replacing the golden evidence in one of k hops.

4.4 Knowledge Transfer Experiments

We focus on investigating knowledge transfer between structured and unstructured sources in this chapter, though the proposed SIMULTQA can be applied to any open-domain CQA datasets. We seek to answer three research questions (RQs):

- **RQ1**: Can the knowledge learned on one source help the QA performance on another one? (§4.4.2)
- **RQ2**: What kind of knowledge has been transferred between KB and text? (§4.4.3)
- **RQ3**: Can knowledge transfer help answer questions by both sources? (§4.4.4)

4.4.1 Experimental Setup

Choice of Datasets. Investigating knowledge transfer between text and KB requires at least one dataset from each source. Without losing the generality, we choose Wikipedia and Freebase as the source for text and KB respectively, and select their arguably the most representative CQA dataset to cover the majority of reasoning types. We leave applying SIMULTQA to other domain-specific sources and datasets as future work.

The selected large-scale KB dataset is Complex WebQuestions (CWQ) (Talmor and Berant, 2018) that consists of around 27K/3.5K/3.5K samples for train/dev/test. The text dataset is HotpotQA (Yang et al., 2018) that consists of around 90K/7.4K/7.4K samples for train/dev/test. For both datasets, we focus on the most practical setting, which is the open-domain QA, meaning that the model needs to reason over the entire knowledge space for answering the question.

Implementation Details. We adopt pre-trained BERT models (Devlin et al., 2019) using the uncased base configuration (768-hidden) for our reasoning path construction and reranking module. We follow Graph Retriever (Asai et al., 2020) and use their pre-trained whole word masking uncased large configuration (1024-hidden) for the reader. During the process of reasoning path construction, we set the number of negative examples along with the gold example as 30, set the number of hops as 2, and use beam search when doing the inference. Beam size is set as 5 for CWQ and 9 for HotpotQA.

4.4.2 RQ1: Quantitative measurement

Pre-training and Fine-tuning. A straightforward way to investigate the effect of knowledge transfer between text and KB is to leverage the pre-training and fine-tuning paradigm, where we first pre-train SIMULTQA on one source and fine-tune it on another one. The transfer effect then can be measured by the performance difference with and without the pre-training stage. Furthermore, to demonstrate the transfer effect carefully, we focus on the low-resource setting where we increasingly add more samples for the fine-tuning. Note that we only pre-train and fine-tune the first stage of SIMULTQA, which is the retriever, because this is the most important module for multi-hop reasoning.

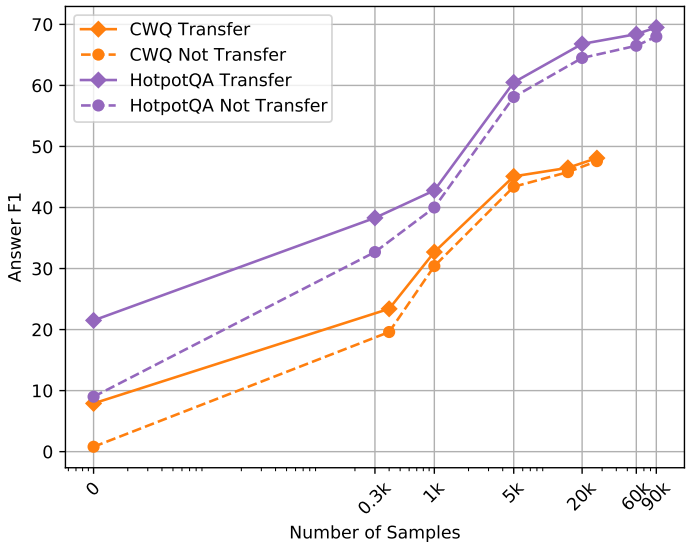

Figure 4.3: Pre-training and fine-tuning experiments on CWQ and HotpotQA datasets. We first pre-train SIMULTQA on one source with the full dataset, then fine-tune it on another one with various sizes of samples.

Transfer Text Knowledge to KB. We show the fine-tuning performance in Figure 4.3, where we can see that pre-training SIMULTQA on text dataset can consistently improve the performance on KB dataset, especially when the fine-tuning data is limited. Specifically, when there is no fine-tuning data for KB (zero-shot transfer), text pre-training achieves about 8 F1 score on CWQ already, meaning that text knowledge can greatly help the QA model on KB. We also notice that when a large number of KB samples are available, the transfer effect becomes less prominent, possibly due to the model begins overfitting KB-specific features.

To further demonstrate the transfer effect on low-resource setting, we conduct few-shot experiments by randomly sampling only a handful of samples for fine-tuning. We sample five times to reduce the randomness of few-shot samples and results are shown in Figure 4.4. We can see the transfer effect from text to KB more clearly, and this finding can be

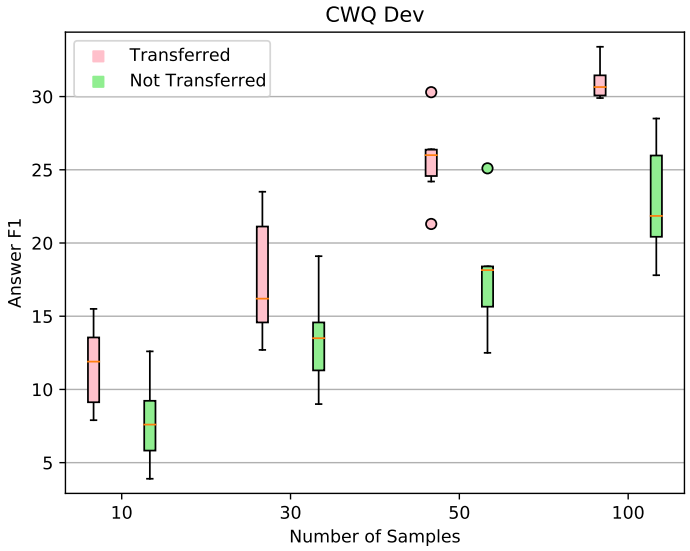

Figure 4.4: Few-shot experiments on CWQ dataset. Boxes extends from the first quartile to the third quartile of the samples, and lines inside boxes mark the medians.

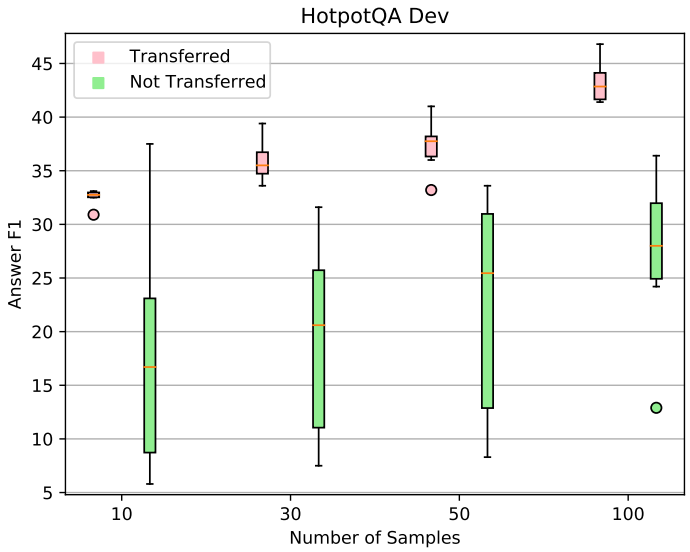

Figure 4.5: Few-shot experiments on HotpotQA dataset. Boxes extends from the first quartile to the third quartile of the samples, and lines inside boxes mark the medians.

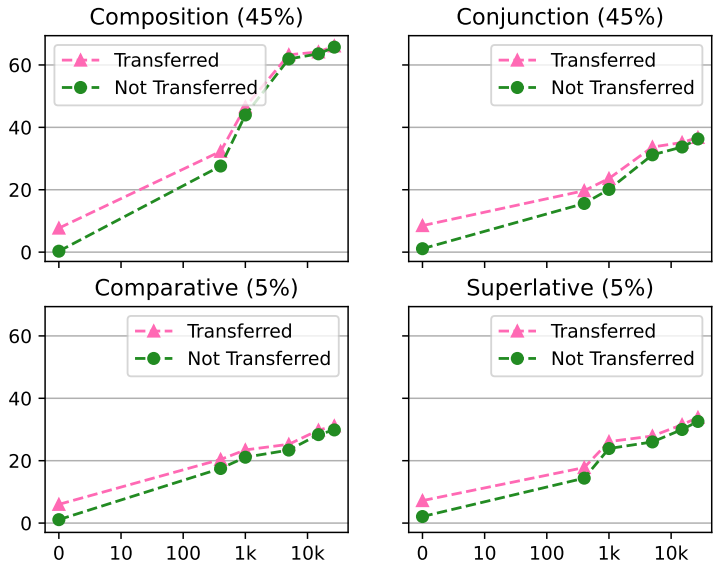

Figure 4.6: Analysis of reasoning types in CWQ. Numbers in parentheses are percentages of types.

leveraged to boost the performance of KBQA in low-data region when human annotations are expensive to collect over domain-specific KBs.

Transfer KB Knowledge to Text. Figure 4.3 shows the pre-training on KB also provides performance boost for fine-tuning on text domain in the low-resource setting. In zero-shot transfer, pre-training on KB brings about 12.5 F1 improvement, which verifies that KB knowledge can also help answer text-based questions. Moreover, few-shot experiments in Figure 4.5 demonstrate the transfer effect when $<$ 100 text-based samples are available. We notice that the variance of few-shot experiments is greatly reduced by the pre-training, indicating another potential useful transfer effect may be to help reduce the instability in the few-shot learning. Meanwhile, we conduct error analysis for both CWQ and HotpotQA respectively in Table 4.2.

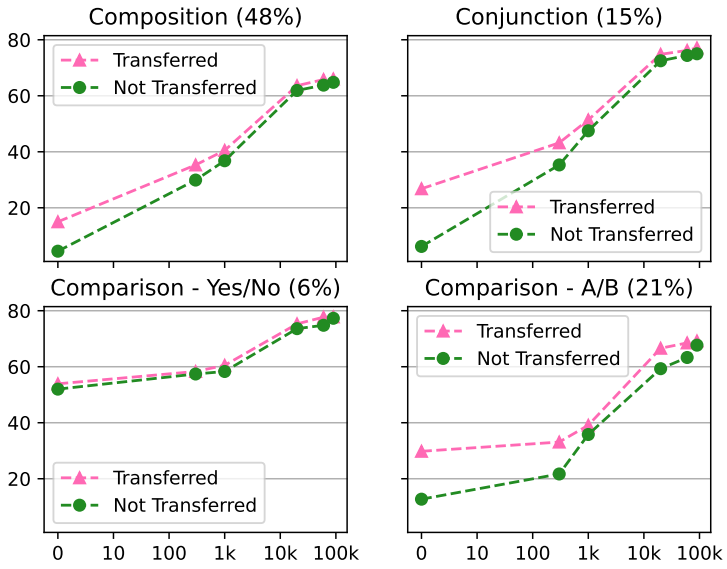

Figure 4.7: Analysis of reasoning types in HotpotQA. Numbers in parentheses are percentages of types.

4.4.3 RQ2: What has been transferred?

We further conduct fine-grained analyses under previous experiment settings trying to answer what knowledge is transferred between structured and unstructured sources. We hypothesize three major factors that may influence the transfer effect and test their correlations with performance changes.

Reasoning types play a central role in answering complex questions. SIMULTQA is expected to learn similar reasoning processes from structured/unstructured sources if the knowledge about certain reasoning types is transferred. We analyze the transfer effect w.r.t. various reasoning types defined in both datasets (we refer to the original papers (Talmor and Berant, 2018; Yang et al., 2018) for their detailed definitions). As shown in Figure 4.6 and 4.7, the most shared two types in both text and KB, composition (i.e., infer the bridge entity) and conjunction (i.e., checking multiple properties) questions are benefited

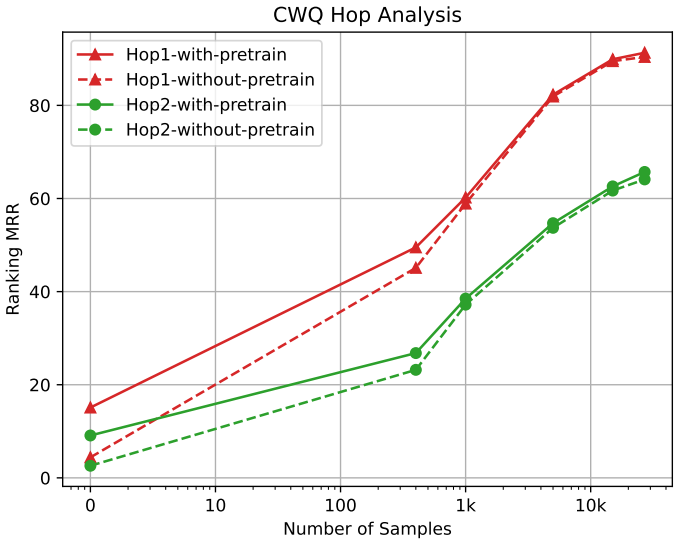

Figure 4.8: Hop Analysis on the CWQ dataset.

from knowledge transfer the most (especially in the zero-shot transfer), which suggests that SIMULTQA is able to transfer similar reasoning processes between disparate sources regardless of their distinct surface forms.

Another interesting observation is for the Comparison - A/B on HotpotQA (e.g., *Who is older, A or B?*) that has a larger F1 score gain under the zero-shot setting. This type asks a two-choice question which can be answered by locating an entity as the final answer through iteratively retrieving two evidences, which is similar to the chain reasoning in Composition and Conjunction. Although this specific reasoning type is not shared by both sources, the similarity between the reasoning processes makes it benefited from knowledge transfer.

Reasoning hops correspond to decomposed sub-questions from a complex question and we are interested in whether the transfer effect varies according to different hops. In both KBQA and TextQA, the first hop sub-question tends to closely connect with a topic entity

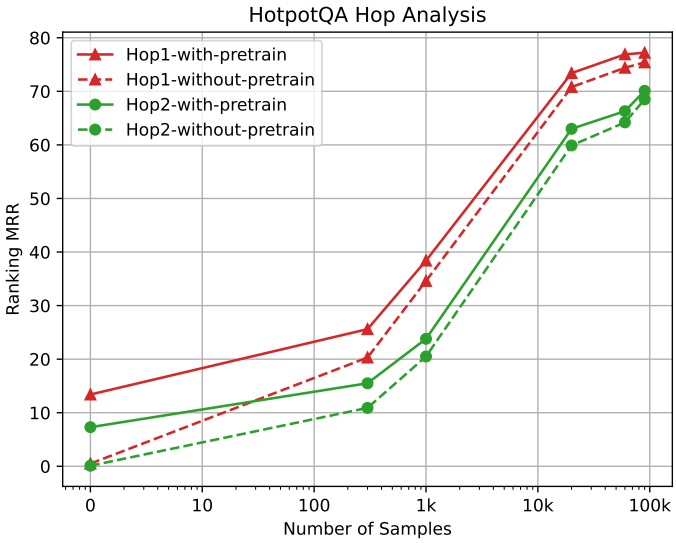

Figure 4.9: Hop Analysis on the HotpotQA dataset.

or phrase mentioned in the question, while the subsequent (second) hops require more semantic inference to answer the sub-question. As shown in Figure 4.8, the first hop in CWQ dataset usually gets higher retrieval performance and can be transferred from the other source, which indicates that the knowledge of finding the topic entity in the question is transferred. We also show the hop analysis for HotpotQA in Figure 4.9. Similar to the observation on CWQ, it shows that the first hop in HotpotQA gets higher retrieval performance and can be transferred from the other source, which further validate that the knowledge of finding the topic entity mentioned in the question is transferred.

Question similarity measures the semantic similarities between questions in testing and training. We hypothesize that the transfer might be easier for testing questions if some similar ones appear in the training. We investigate the zero-shot transfer to study the influence of pre-training questions more directly. Specifically, for a CWQ question in testing set, we calculate its semantic similarities with all HotpotQA questions in pre-training and

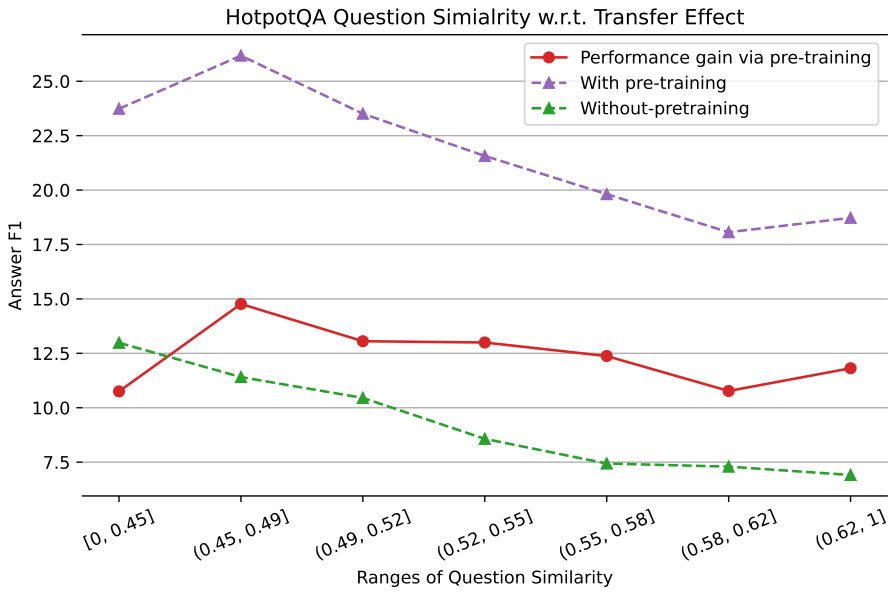

Figure 4.10: Relationship between question similarity and performance gain.

take the average of top 5 similarities. We then split CWQ testing questions into several chunks based on this averaged similarity and aggregate their QA performance before and after the pre-training. We do the same thing for the other direction of transfer. We present the relationship between question similarity and performance in HotpotQA on Figure 4.10. Interestingly, we observe that question similarity is not correlated with transfer effect, i.e., higher similar testing questions are not necessarily to obtain larger performance gain. This finding implies that SIMULTQA transfers the reasoning process in a high-level semantic space rather than through low-level lexical features. We show questions similarity for CWQ in Figure 4.11, where we also find question similarity is not correlated with the transfer effect.

Error analysis is conducted under the full dataset fine-tuning setting to further understand the transfer behaviors by manually checking errors and categorizing them. As is shown

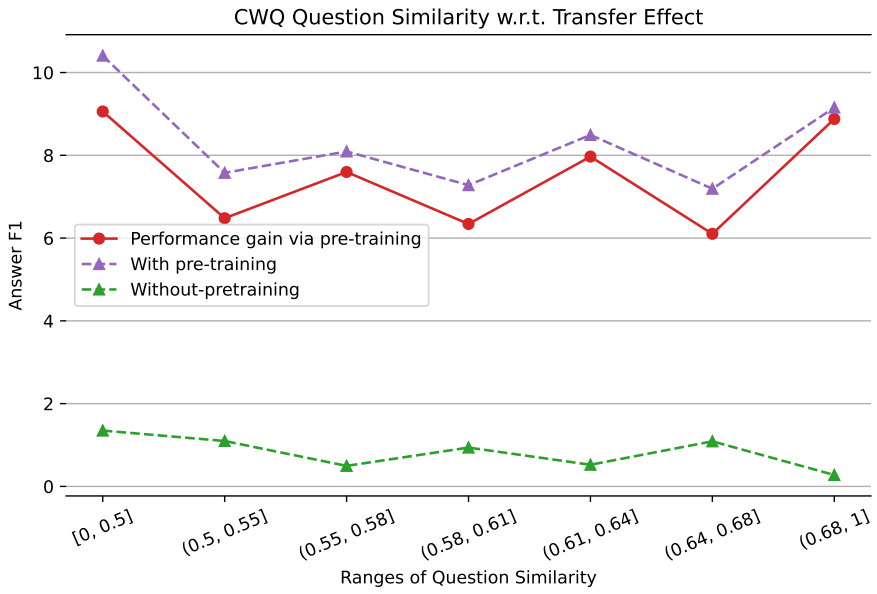

Figure 4.11: Relationship between question similarity and performance gain on CWQ.

in Table 4.2, 75% of wrongly answered questions sampled from CWQ contain additional constraints or arithmetic operations which are hard to be supported by text corpus. 45% questions sampled from HotpotQA contain semantic knowledge or relations which cannot be covered in Knowledge Base. 35% of them don't follow the chain reasoning process and are not suitable to be decomposed to answer step by step like KBQA. The other remaining questions are related to errors in retrieval, re-ranking or span extraction process. These unshared knowledge between CWQ and HotpotQA make it reasonable that those wrongly answered questions in one data source cannot be contributed from the other data source.

4.4.4 RQ3: Answering complex questions by both sources

To directly measure the transfer effect, in previous sections, the reasoning is always performed on the same knowledge source as where the question is from, e.g., a text-based question is answered by the text reasoning path. Now, we ask whether questions can be

Complex question: What is European Union country used the Hungarian forint as its main currency?
Gold KB reasoning path: European Union $\xrightarrow{members}$ y1(CVT) $\xrightarrow{member}$ Hungary $\xleftarrow{currency\_used}$ Forint
Reasoning paths from text source: 1. *(first passage)* The currency of Hungary is the Hungarian forint since 1 August 1946 ... *(second passage)* As a member of the European Union, Hungarian government ... replace the forint with the euro. 2. *(first passage)* The forint is the currency of Hungary. ... and the forint has been declared fully convertible. *(second passage)* As a member of the European Union, the long-term of aim of the Hungarian government ... 3. *(first passage)* The Gulden or forint was the currency ... and the Austro-Hungarian Monarchy ... *(second passage)* In Hungary, the forint was divided into ... for the unit and subunit.

Table 4.1: Case Study. The question comes from CWQ dataset and is originally answered by a KB reasoning path.

	Type	**%**
CWQ	Questions with constraints	50
	Questions with aggregation functions	25
	Others	25
HotpotQA	Relations not covered in KB	45
	Not satisfy chain reasoning	35
	Others	20

Table 4.2: We manually analyze 20 questions with wrong predicted answers respectively from CWQ and HotpotQA and categorize them.

better answered by considering both sources. Note that this is a more challenging setting because questions in both datasets only have supervisions from one source, which thus requires stronger transfer signal. Moreover, we can utilize this setting to test how complementary two knowledge sources are, regarding how much they can help each other. Specifically, in addition to the annotated reasoning paths, we collect candidate paths from

(HotpotQA) In the television series Green Hornet, which actor played the role of Kato?
Gold reasoning path from text source: *(first passage)* The Green Hornet is a television series on ABC ... starring Van Williams and Bruce Lee ... *(second passage)* Kato is a fictional character ... was portrayed by Bruce Lee.
Reasoning paths from KB source: 1. Green Hornet $\xleftarrow{series}$ y1(CVT) $\xleftarrow{starring\_roles}$ Bruce Lee $\xleftarrow{actor}$ y2(CVT) $\xleftarrow{appear\_in\_tv\_program}$ Kato 2. Green Hornet $\xleftarrow{film}$ y1(CVT) $\xleftarrow{character}$ AI Hodge $\xleftarrow{notable\_types}$ TV Actor 3. Green Hornet $\xleftarrow{film}$ y1(CVT) $\xleftarrow{character}$ Seth Rogen $\xleftarrow{appeared\_on}$ y2(CVT) $\xleftarrow{appearance\_type}$ Host

Table 4.3: Case study. The question comes from HotpotQA and is originally answered by a textual reasoning path.

CWQ	**F1**	**Hit@1**
SIMULTQA- KB	46.7	47.7
SIMULTQA- Hybrid	48.5	49.8
HotpotQA	**F1**	**EM**
SIMULTQA- Text	71.7	58.8
SIMULTQA- Hybrid	71.2	58.4

Table 4.4: Comparing single and hybrid evaluations.

the other source, i.e., KB paths for text-based questions and text paths for KB-based questions. The final reranking will select the best path from both KB and text paths for all questions. We refer to this setting as the hybrid evaluation.

Our preliminary experiments show that pre-training on one source and then fine-tuning on the other tends to forget the knowledge of the first source, leading to less satisfactory results. Therefore, we jointly train SIMULTQA by iteratively sampling batches from both sources to expose the model to both sources equally in the training time. We then compare the hybrid evaluation with the single-source evaluation in Table 4.4. For CWQ dataset,

SIMULTQA- Hybrid achieves 1.8 F1 score gains after incorporating text paths for the inference, while the performance of HotpotQA is not influenced in hybrid evaluation after incorporating KB paths. This shows that text knowledge is easier to be transferred to help KB-based questions.

We also conduct case studies by retrieving top-ranked reasoning paths in hybrid evaluation. Table 4.1 presents a CWQ question and shows that top-ranked text paths are closely related to the golden KB path, indicating that linguistics variants of text knowledge can greatly help KB reasoning. On the other hand, KB knowledge seems to be less helpful to answer text-based questions based on the overall QA performance in Table 4.4, partially due to the incompatibility between TextQA and KBQA dataset, e.g., entities and relations that cannot be mapped to KB, reasoning types that cannot be answered by KB (see Section 4.4.3), etc. However, we still find cases in HotpotQA in Table 4.3 to show KB can somehow contribute to textual reasoning as well.

4.5 Discussion and Conclusion

In this chapter, we study CQA over structured and unstructured knowledge sources (i.e., KB and text particularly), and focus on studying the knowledge transfer between different knowledge sources. To facilitate the transfer, we first propose a unified CQA framework, SIMULTQA to bridge KBQA and TextQA systems. Empirical results show that knowledge transfer enables substantial improvements in low-resource domains. More importantly, we conduct fine-grained analyses to shed more light on how knowledge is transferred to inspire future research on knowledge transfer between sources, and we conclude the chapter with insights for future CQA datasets and systems.

Based on our findings of knowledge transfer for CQA in this chapter, we discuss the following directions for future CQA datasets and systems.

Knowledge transfer for efficient CQA dataset annotations. When annotating new CQA datasets whether, on text or KB, it would be beneficial to leverage pre-trained SIMULTQA on other sources to discover high-quality reasoning paths for further annotating, which will save much annotation cost.

Diversity of reasoning types. Both text and KB sources are dominant by relatively easy reasoning types, e.g., composition and conjunction. Future CQA datasets should pursue more diverse and harder reasoning types, e.g., types with constraints and arithmetic operations (Dua et al., 2019).

A universal reasoning module. Investigating knowledge transfer between text and KB in this chapter suggests that despite the discrepancy of surface forms in different sources, their underlying reasoning processes could be shared. This points out the possibility of learning a universal reasoning process from multiple sources and it is strongly desired to modularize such a reasoning process, which can be injected into future QA systems.

Moreover, SIMULTQA is built on top of the recent popular retrieve-reranking paradigm in open-domain question answering. Our key intuition is to find the similarities between KBQA and TextQA and reformulate the pathfinding in KBQA and paragraph searching in TextQA as a retrieval problem, then rerank the resulting reasoning paths for further improving the performance. This framework is general and can incorporate different solutions for the retrieval and reranking components. More importantly, the alternative architecture can explore the recent generation-reranking paradigm inspired by the great generative power of LLMs, which replaces the retrieval module with a text generation module from LLMs and directly retrieves factual knowledge from the memory of LLMs (Yu et al., 2022). A more

intriguing design of the model needs to learn the multi-hop reasoning paths unsupervised from the data by transferring knowledge from exiting KBQA and TextQA datasets.

Chapter 5: Multitask Prompt Tuning Enables Parameter-Efficient Transfer Learning

The previous chapter presents how to transfer knowledge between two QA systems built on top of structured and unstructured knowledge sources individually. We now study a more generic transfer learning setting and consider most NLP tasks, such as text classification, natural language inference, and question answering. We explore how to transfer the knowledge pre-trained on generic NLP tasks to many diverse downstream tasks. Specifically, in this chapter, we study transferring knowledge between tasks via prompt tuning, where a base pretrained model is adapted to each task via conditioning on learned prompt vectors.

Prompt tuning has emerged as a promising approach for the efficient adaptation of large language models to multiple downstream tasks. However, existing methods typically learn soft prompt vectors from scratch, and it has not been clear how to exploit the rich cross-task knowledge in task-specific prompt vectors to improve performance on target downstream tasks. In this chapter, we propose multitask prompt tuning (MPT), which first learns a single transferable prompt by decomposing and distilling knowledge from multiple task-specific source prompts. We then learn multiplicative low rank updates to this shared prompt to efficiently adapt it to each downstream target task. Extensive experiments on 21 NLP datasets demonstrate that our proposed approach outperforms the state-of-the-art

methods, including the full finetuning baseline in some cases, despite only tuning 0.035% as many task-specific parameters.

5.1 Introduction

Finetuning Pretrained Language Models (PLMs) has led to significant improvements across various downstream NLP tasks (Devlin et al., 2019; Howard and Ruder, 2018; Raffel et al., 2020). However, the conventional paradigm of full task-specific finetuning is difficult to scale to multiple tasks given that contemporary PLMs can have hundreds of millions (or even billions) of parameters. There has thus been a growing interest in developing *parameter-efficient* methods for model tuning (Houlsby et al., 2019; Lester et al., 2021; Ding et al., 2022), where the goal is to learn only a small number of additional parameters per task while achieving performance comparable to full model finetuning.

Prompt tuning (PT), which prepends continuous prompt vectors to the input, has emerged as a promising approach for parameter-efficient transfer learning with PLMs (Liu et al., 2021a; Li and Liang, 2021; Lester et al., 2021; Liu et al., 2022b, 2021b). PT freezes the PLM parameters and only learns a small set of task-specific prompt vectors. Despite their impressive performance, there is still a large gap between prompt tuning and full finetuning for many models and tasks (Lester et al., 2021). Prompt vectors trained using task-specific training data only are more sensitive to initialization and require significantly more training time than finetuning (Su et al., 2022; Zhong et al., 2022).

Recent work has proposed to address these issues through *transferring* prompt vectors from different tasks (Su et al., 2022; Zhong et al., 2022). These methods first train soft prompts on multiple source tasks and then use these pretrained prompts to initialize the prompt for further finetuning on a target task based on a (potentially learned) similarity

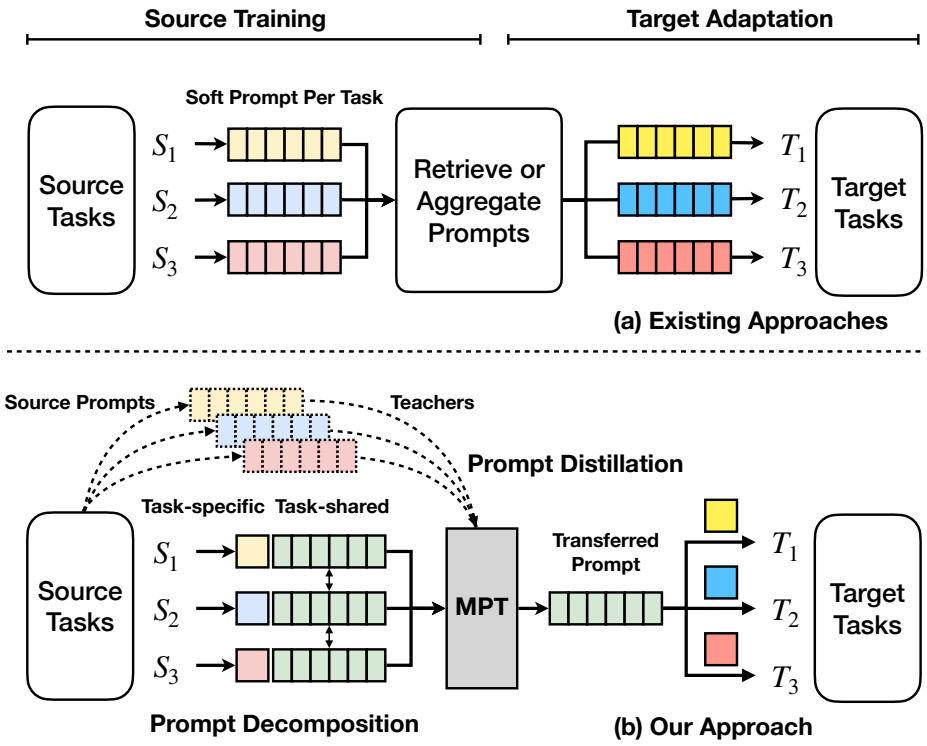

Figure 5.1: **A conceptual overview of our approach.** Instead of retrieving or aggregating source prompts, MPT learns a single transferable prompt exploiting rich cross-task shared knowledge. The transferable prompt is learned via prompt decomposition and distillation to enable parameter-efficient transfer learning with PLMs.

measure (Vu et al., 2022; Asai et al., 2022) (see Figure 5.1, top). In this chapter, we extend this line of work and introduce *multitask prompt tuning* (MPT), which uses multitask data to learn a *single* prompt that can be efficiently transferred to target tasks. While conceptually simple, learning a shared prompt space can be practically challenging as it requires learning commonalities across source tasks while minimizing interference. We decompose the soft prompt of each source task (which can be represented as a prompt matrix) as a multiplication of a shared matrix and a low-rank task-specific matrix, and find that this decomposition is more effective than simply sharing the prompt matrix across all tasks.

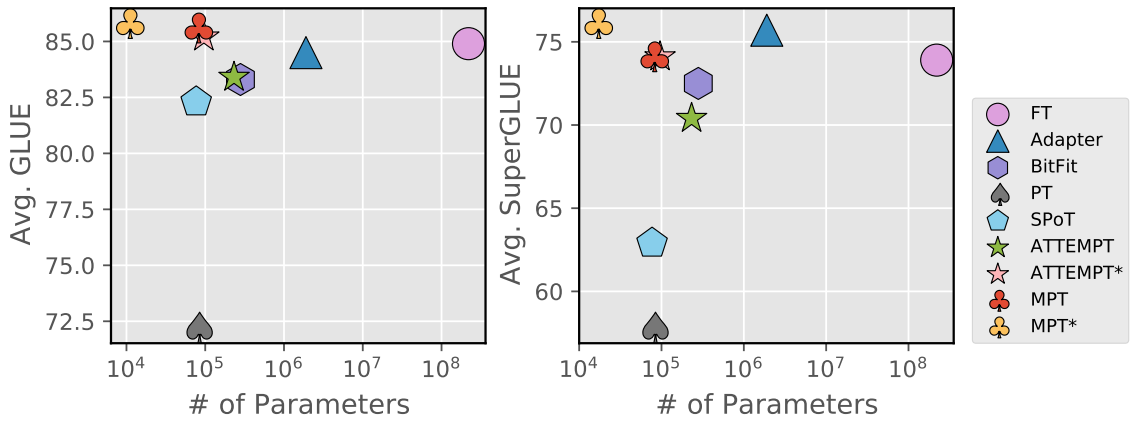

Figure 5.2: **Parameter efficiency on GLUE and SuperGLUE.** All results are based on T5-Base model (Raffel et al., 2020). Adapter (Houlsby et al., 2019), BitFit (Zaken et al., 2022), PT (Lester et al., 2021), SPoT (Vu et al., 2022), ATTEMPT (Asai et al., 2022). * indicates multitask training on target tasks. Our MPT approach—which transfers a single shared prompt learned from multiple source tasks using prompt decomposition and distillation—outperforms all the existing prompt tuning methods and full model finetuning (FT), despite updating much fewer task-specific parameters. Best viewed in color.

This decomposition is learned through knowledge distillation from soft prompts obtained from regular prompt tuning. To transfer to new tasks, we perform low-rank multiplicative updates to the shared prompt matrix. Figure 5.1 (bottom) illustrates our approach.

Extensive experiments on 21 NLP datasets across diverse tasks demonstrate the effectiveness of our proposed approach over state-of-the-art prompt transfer methods. On the SuperGLUE benchmark (Wang et al., 2019a), MPT with T5-Base (Raffel et al., 2020) yields a 16.3 improvement over the vanilla prompt tuning baseline (PT, Lester et al., 2021), and also outperforms the most competitive multitask prompt transfer baseline (ATTEMPT, Asai et al., 2022) despite tuning much fewer task-specific prompt parameters (77.6K vs 232K). On some benchmarks, MPT exceeds the performance of full finetuning while only requiring 0.035% tunable parameters per task (see Figure 5.2). We also discover that MPT

is very effective for few-shot learning with 4-32 labels. Finally, ablation studies further show that MPT matches the performance of full finetuning at different model scales ranging from 60M to 770M parameters. Our code and models will be made publicly available.

5.2 Related Work

Parameter-Efficient Transfer Learning. Parameter-efficient transfer learning for pre-trained language models is an active research area (Ding et al., 2022). Adapters (Houlsby et al., 2019; Mahabadi et al., 2021) and variants (Hu et al., 2021; Karimi Mahabadi et al., 2021) insert trainable layers, while BitFit (Zaken et al., 2022) only updates the bias parameters without changing any other model parameters. Diff pruning (Guo et al., 2021) and FISH (Sung et al., 2021) learn sparse updates to the original PLM. Another popular choice is prompt tuning (Lester et al., 2021) which only updates soft prompts prepended to the input. Prefix-tuning for optimizing continuous prompts for natural language generation tasks is presented in (Li and Liang, 2021). UNIPELT learns to combine different tuning methods via gating mechanism (Mao et al., 2022). HyperPrompt (He et al., 2022) introduces task-conditioned hyperprompts that conditions the model on task-specific information for constructing prompts. Discrete (i.e., hard) prompts have also been shown to be effective in many cases (Schick and Schütze, 2021a,b; Gao et al., 2021; Malkin et al., 2022). However, our approach is most related to the transferability of prompts (Wang et al., 2021a; Vu et al., 2022; Su et al., 2022), which focuses on boosting the performance of prompt tuning across many tasks. SPoT (Vu et al., 2022) selects one prompt using a similarity measure and ATTEMPT (Asai et al., 2022) adopts an attention mechanism over the source prompts to initialize the prompt for a target task. Unlike existing works, our proposed approach learns

a single shared prompt by decomposing and distilling knowledge from source prompts in a structured way for efficient adaptation to a diverse set of target tasks.

Multitask Learning. Multitask learning, which focuses on simultaneously solving multiple related tasks, has been studied from multiple perspectives (Zhang and Yang, 2021; Ruder, 2017). Transferring a model fine-tuned on multiple source tasks to another target task is a common approach to multitask learning (Vu et al., 2020; Raffel et al., 2020; Aghajanyan et al., 2021a; Zhong et al., 2021; Clark et al., 2019b). Few recent works show zero-shot and few-shot transfer capabilities of language models through massive multitask learning over a large number of tasks (Sanh et al., 2022; Wang et al., 2022; Liu et al., 2022a; Wei et al., 2021). Designing specific parameter-sharing strategies is also another recent trend in multitask learning (Ruder et al., 2019a; Sun et al., 2020a; Misra et al., 2016). While our proposed approach is inspired by these methods, in this chapter we focus on multitask prompt transfer for parameter-efficient adaptation of language models, which still remains as a challenging and largely under-addressed problem.

Knowledge Distillation. Knowledge distillation has been used to improve performance and efficiency across many tasks (Gou et al., 2021), including model compression (Hinton et al., 2015; Jiao et al., 2020; Sanh et al., 2019), transfer learning (Furlanello et al., 2018; Xu et al., 2020), machine translation (Zhou et al., 2019), question answering (Hu et al., 2018), and document retrieval (Shakeri et al., 2019). Concurrent to our work, PANDA (Zhong et al., 2022) uses knowledge distillation with a new metric to better predict the prompt transferability across different combinations of source-target tasks. This differs from MPT, which uses a prompt decomposition strategy to leverage commonalities across the source tasks while minimizing interference between them. In addition, PANDA focuses on transferring from one source task to another target task using a similarity measure (similar to

SPoT (Vu et al., 2022)), while our MPT approach leverages multitask learning to better exploit the rich cross-task knowledge in prompt transfer.

5.3 Methodology

Given a set of source tasks $\boldsymbol{S} = \{\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_\kappa\}$ and target tasks $\boldsymbol{T} = \{\mathcal{T}_1, \mathcal{T}_2, ..., \mathcal{T}_\tau\}$, our goal is to learn a single soft prompt over $\boldsymbol{S}$ that can be efficiently updated to enable better performance on $\boldsymbol{T}$. Simply training a single soft prompt on $\boldsymbol{S}$ is sub-optimal as it can fail to leverage commonalities across source tasks while minimizing interference. To this end, our multitask prompt tuning (MPT), aims to efficiently compress task-shared knowledge in $\boldsymbol{S}$ into a single prompt ϕ_s to improve performance on $\boldsymbol{T}$ while filtering out task-specific information that is less useful for transfer learning.

Prompt Tuning. Given a pre-trained language model with parameters Θ and one target task $\mathcal{T}$ with training data $(\boldsymbol{X}, \boldsymbol{Y}) = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$, directly finetuning all the parameters by maximizing the conditional probability $P(\boldsymbol{Y}|\boldsymbol{X}; \Theta)$ is expensive and often tends to overfit on small datasets. An alternative to finetuning that is more parameter-efficient is prompt tuning (PT), which randomly initializes a small number of learnable prompt vectors (i.e., soft prompts) to be prepended to the input embeddings of the PLM while freezing model parameters Θ (Lester et al., 2021; Liu et al., 2022b). Formally, for a sequence of input tokens with token embeddings as $\boldsymbol{x} = \{\boldsymbol{t}_1, \boldsymbol{t}_2, ..., \boldsymbol{t}_n\} \in \mathbb{R}^{n \times d}$, PT prepends the soft prompt $\boldsymbol{P} \in \mathbb{R}^{l \times d}$ with the same dimension as the token embedding d and vector length as l. Then PT optimizes the following loss function:

$$\mathcal{L}_{\text{PLM}} = -\sum_i \log P(\boldsymbol{y}_i \,|\, [\boldsymbol{P}; \boldsymbol{x}_i]\,; \Theta\,), \tag{5.1}$$

with respect to $\boldsymbol{P}$. While this approach has been successful on some tasks and models, researchers have observed that vanilla PT can sometimes lead to lower performance (especially on smaller PLMs), be slow to converge, and have high sensitivity to the initialization (Lester et al., 2021; Su et al., 2022; Zhong et al., 2022). Recent works address these issues by first training prompts on multiple source tasks, and then initializing the prompts for a target task via some similarity measure or learned attention (Asai et al., 2022; Vu et al., 2022). We extend this line of work and propose a novel framework for transferring multitask knowledge into a single soft prompt to enable more performant and parameter-efficient transfer learning to downstream target tasks $\boldsymbol{T}$.

5.3.1 Multitask Prompt Tuning

Our proposed framework mainly consists of two stages, *source training* and *target adaptation*. The proposed MPT framework first focuses on the source training to generate a single soft prompt to be reused in the second stage for target task adaptation. Specifically, task prompts for source tasks are decomposed into a task-shared component and a low-rank task-specific component (*prompt decomposition*), where the former is shared across all tasks, and the latter is task-specific. We also use prompt distillation to better transfer multitask knowledge to the shared component by distilling knowledge from multiple tasks-specific source prompts. Once learned, the shared prompt matrix is adapted to a downstream target task via low-rank multiplicative updates.

Prompt Decomposition. The goal of prompt decomposition is to enable efficient knowledge sharing across $\mathcal{S}$ while still allowing each task to maintain its own parameters to encode task-specific knowledge. Specifically, we decompose the soft prompt $\boldsymbol{P}_k$ for k-th task into two parts, as shown in Figure 5.3. Let $\boldsymbol{P}^* \in \mathbb{R}^{l \times d}$ denote the shared prompt

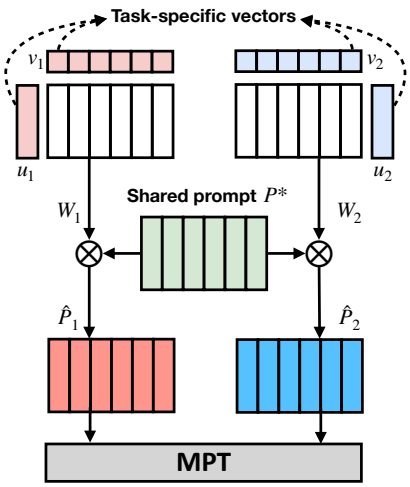

Figure 5.3: An illustration on prompt decomposition for two tasks.

across all tasks, and further let $\boldsymbol{u}_k \in \mathbb{R}^l$, $\boldsymbol{v}_k \in \mathbb{R}^d$ be the task-specific vectors for each task k. The task-specific vectors form a rank-one matrix $\boldsymbol{W}_k = \boldsymbol{u}_k \cdot \boldsymbol{v}_k^T$, which has the same dimensions as the shared prompt $\boldsymbol{P}^*$. The final task prompt $\widehat{\boldsymbol{P}}$ for k-th source task is then parameterized as:

$$\widehat{\boldsymbol{P}}_k = \boldsymbol{P}^* \circ \boldsymbol{W}_k = \boldsymbol{P}^* \circ (\boldsymbol{u}_k \cdot \boldsymbol{v}_k^T) \tag{5.2}$$

where $\circ$ denotes the Hadamard product between two matrices. Our parameterization of prompt decomposition is inspired by prior low-rank methods (Li et al., 2018; Aghajanyan et al., 2021b; Wen et al., 2020), such that general information of $\mathcal{S}$ can be captured by "slow" weights $\boldsymbol{P}^*$ shared across tasks and "fast" weights $\boldsymbol{W}_k$ could encode task-specific knowledge in a low-rank subspace.

Prompt Distillation. Simply using the mixed golden labels to guide the prompt decomposition directly can bias the shared component to overfit to larger tasks. We found knowledge distillation from separately-trained source prompts to be an effective strategy

for learning good decomposable prompts. Specifically, we first obtain a teacher prompt $\boldsymbol{P}_k^t$ for the k-th source task by conventional prompt tuning. We then randomly initialize a corresponding student prompt as $\widehat{\boldsymbol{P}_k^s} = \boldsymbol{P}^* \circ (\boldsymbol{u}_k \cdot \boldsymbol{v}_k^T)$, where all student prompts share $\boldsymbol{P}^*$ and have their own task-specific vectors as described above. Following Sanh et al. (2019), we design distillation losses to transfer cross-task knowledge into the shared prompt matrix. The first loss is to match the output probability distributions of students and teachers through minimizing their KL-Divergence,

$$\mathcal{L}_{\text{Logits}} = \sum_k \sum_{i \in \mathcal{S}_k} \text{KL}(P(\boldsymbol{y}_i|[\boldsymbol{P}_k^t; \boldsymbol{x}_i]), P(\boldsymbol{y}_i|[\widehat{\boldsymbol{P}_k^s}; \boldsymbol{x}_i])). \tag{5.3}$$

We further use a temperature T to control the smoothness of the output distribution for both teacher and student models as $p_j = \frac{1}{Z} \exp(z_j/T)$, where z_i is the logit score for class j and Z is the normalization factor. We also have an additional mean squared loss on teacher model hidden states,

$$\mathcal{L}_{\text{Hidden}} = \sum_k \sum_{i \in \mathcal{S}_k} (\boldsymbol{H}_{ki}^s - \boldsymbol{H}_{ki}^t)^2, \tag{5.4}$$

where $\boldsymbol{H}_{ki}^t, \boldsymbol{H}_{ki}^s$ denotes the hidden states of teacher and student networks, respectively, consisting of a sequence of hidden vectors for i-th input. Such additional distillation loss from intermediate states has been shown to improve results in distilling PLMs (Jiao et al., 2020; Shleifer and Rush, 2020). Finally, our total loss function to train student source prompts for obtaining a single shared prompt to be transferred to the target side is formulated as follows:

$$\mathcal{L}_{\text{Total}} = \mathcal{L}_{\text{PLM}} + \lambda(\mathcal{L}_{\text{Logits}} + \mathcal{L}_{\text{Hidden}}), \tag{5.5}$$

where $\mathcal{L}_{\text{PLM}} = \sum_k \mathcal{L}_{\text{PLM}}^k$ represents the aggregated task losses for all source tasks, and λ is a weight to balance the impact of distillation loss terms.

5.3.2 Source Training and Target Adaptation

Training the single source prompt to be transferred to target tasks requires two steps. First, the teacher prompts for all source tasks are pretrained individually through vanilla prompt tuning. Then, we conduct multitask training on $\mathcal{S}$ to jointly learn the single shared prompt via the knowledge distillation loss function in Equation 5.5. We also adopt a simple stochastic task sampling strategy, which dynamically changes the number of tasks per batch. In particular, for each batch of multitask samples, we randomly select a number k from $[2, \kappa]$ first, then randomly choose k tasks from $\mathcal{S}$ and their corresponding samples to constitute mini-batches. Such dynamic task sampling strategies are common in the PLM multitask learning literature (Raffel et al., 2020).

For target adaptation, initialize the target prompt to be the Hadamard product of the shared prompt and the low-rank target prompt matrix and optimize with the regular task loss in Equation 5.1. We remark that our proposed method can also be used for multitask learning on target tasks to enable more parameter-efficient adaption of pretrained language models.

Parameter-Efficiency. Our method is parameter-efficient during both source training and target adaptation. Each task contains the shared prompt $l \times d$ that has the same dimensions as a vanilla soft prompt and a smaller number of task-specific vectors $(l + d)$. Thus, the total number of tunable parameters for a single target task is $(l \times d) + (l + d)$. For a *group* of target tasks, the total number of tunable parameters is $(l \times d) + (l + d)\tau$, where τ is the number of target tasks. We list and compare different methods in terms of the number of trainable parameters in Table 5.1.

5.4 Experiments

We conduct extensive experiments on 21 diverse NLP datasets to show that MPT outperforms strong baselines in both full-dataset (Tables 5.1, 5.2) and few-shot adaptation settings (Table 5.3), while achieving more parameter-efficiency compared to existing methods (Figure 5.2). We also perform comprehensive ablation studies and analysis to better understand the effect of model sizes (Figure 5.4), prompt length (Figure 5.5) and different components in our approach (Table 5.4).

5.4.1 Experimental Setup

Datasets and Tasks. As in Asai et al. (2022) we evaluate performance of MPT using 6 datasets with more than 100k annotations as *source* tasks (MNLI (Williams et al., 2017), QNLI (Demszky et al., 2018), QQP (Wang et al., 2018a), SST-2 (Socher et al., 2013b), SQuAD (Rajpurkar et al., 2016), and ReCoRD (Zhang et al., 2018b)) and 21 datasets from four benchmarks as *target* tasks, namely MultiRC (Khashabi et al., 2018), BoolQ (Clark et al., 2019a), WiC (Pilehvar and Camacho-Collados, 2018), WSC (Levesque et al., 2012), and CB (De Marneffe et al., 2019) from SuperGLUE (Wang et al., 2019a); RTE (Giampiccolo et al., 2007), CoLA (Warstadt et al., 2019), STS-B (Cer et al., 2017), MRPC (Dolan and Brockett, 2005), MNLI, QQP, QNLI and SST-2 from GLUE (Wang et al., 2018a); Natural Questions (Kwiatkowski et al., 2019), HotpotQA (Yang et al., 2018), NewsQA (Trischler et al., 2017) and SearchQA (Dunn et al., 2017) from MRQA (Fisch et al., 2019); WinoGrande (Sakaguchi et al., 2021), Yelp-2 (Zhang et al., 2015), SciTail (Khot et al., 2018) and PAWS-Wiki (Zhang et al., 2019) from the Others benchmark in (Asai et al., 2022).

Models. Following the standard prompt tuning (Lester et al., 2021; Asai et al., 2022), we mainly experiment using the publicly available pretrained T5-Base model with 220M parameters (Raffel et al., 2020). In our ablation we also consider T5-Small, and T5-Large with 60M, and 770M parameters respectively, to empirically analyze the effect of model size on MPT performance in Figure5.4.

Baselines. We compare our approach with the following baselines. (1) Full finetuning (FT), where all the model parameters are tuned during adaptation on each downstream task, (2) vanilla prompt tuning (PT) (Lester et al., 2021), where target prompt vectors are initialized by randomly sampled top vocabularies, (3) existing prompt transfer methods, including SPoT (Vu et al., 2022) and ATTEMPT (Asai et al., 2022), that initialize target prompts by retrieving or aggregating source prompts, (4) popular parameter-efficient methods including Adapter (Houlsby et al., 2019) and BitFit (Zaken et al., 2022). On GLUE, we also compare with several state-of-the-art methods that adapt a pretrained model to all the target tasks using multitask learning, such as HyperFomer (Mahabadi et al., 2021), HyperDecoder (Ivison and Peters, 2022), multitask variants of FT, and Adapter. We directly quote numbers reported in published papers when possible or use publicly available source codes (Karimi Mahabadi et al., 2021; Mahabadi et al., 2021; Asai et al., 2022) under the same backbone and experimental settings for a fair comparison.

Implementation Details. Following (Karimi Mahabadi et al., 2021), for all datasets, we use the development set as the testing set if the original testing set is not publicly available. We split the original development set into the development and testing set if the training set is small; otherwise, we separate a development set from the training set and use the original development set for testing. We limit the number of training data for Yelp to 100k. For source training, we train MPT on the mixture of source tasks for 5 epochs

with the examples-proportional mixing strategy (Raffel et al., 2020) and stochastic task sampling described in Section 5.3.2. For prompt distillation, we calculate the hidden state loss for hidden states from both the encoder and decoder of T5.

For target adaptation, we reuse the shared prompt from MPT and take averaged source task-specific vectors to initialize the target task-specific vector. We train 20 epochs on small datasets, 10 epochs on large (more than 10k examples) datasets, and 5 epochs on the MRQA dataset. We run all the experiments three times with different random seeds and report the mean numbers. During source training, we set the default learning rate as 0.3 for both task-shared and task-specific components. However, during target adaptation, we use a strategy of two-speed learning rates for those two components, as in Ponti et al. (2022). We set the learning rate to 0.3 and 0.4 for task-shared and task-specific components during adaptation for each target task. Following Lester et al. (2021), we set the default number of tunable tokens per each prompt as 100 and initialize the teacher and student prompts by randomly sampling tokens from T5's vocabulary (Raffel et al., 2020). We set the default batch size for T5-Base as 32 and for model scaling experiments, the batch sizes for T5-Small and T5-Large are 100, and 12 respectively. The default input length for most tasks are set to 256, except MultiRC and MRQA benchmark have the input length as 348 and 512. We set the distillation loss coefficient λ in Equation 5.5 to 0.9 and keep it fixed for all our experiments. In few-shot experiments, for each number of shots k, we randomly sample 10 times from the training set with different random seeds and report the mean performance. Following (Asai et al., 2022), we report our performance on three target tasks, namely BoolQ, CB, and SciTail with the same validation and testing sets in the full-dataset setting. We use 1 NVIDIA Tesla V100 GPU (32GB) for training models on the small datasets and 6 GPUs for training models on the larger datasets.

Method	param/ task	GLUE									SuperGLUE					
		MNLI	QQP	QNLI	SST-2	STS-B	MRPC	RTE	CoLA	Avg.	Multi	BoolQ	WiC	WSC	CB	Avg.
Finetuning	220M	86.8	91.6	93.0	94.6	89.7	90.2	71.9	61.8	84.9	72.8	81.1	70.2	59.6	85.7	73.9
Adapter	1.9M	86.5	90.2	93.2	93.8	90.7	85.3	71.9	64.0	84.5	75.9	82.5	67.1	67.3	85.7	75.7
BitFit	280K	85.3	90.1	93.0	94.2	90.9	86.8	67.6	58.2	83.3	74.5	79.6	70.0	59.6	78.6	72.5
PT	76.8K	81.3	89.7	92.8	90.9	89.5	68.1	54.7	10.6	72.2	58.7	61.7	48.9	51.9	67.9	57.8
SPoT	76.8K	85.4	90.1	93.0	93.4	90.0	79.7	69.8	57.1	82.3	74.0	77.2	67.0	50.0	46.4	62.9
ATTEMPT	232K	84.3	90.3	93.0	93.2	89.7	85.7	73.4	57.4	83.4	74.4	78.8	66.8	53.8	78.6	70.5
MPT	77.6K	85.9	90.3	93.1	93.8	90.4	89.1	79.4	62.4	**85.6**	74.8	79.6	69.0	67.3	79.8	**74.1**
Finetuning*	28M	85.7	91.1	92.0	92.5	88.8	90.2	75.4	54.9	83.8	-	-	-	-	-	-
Adapter*	1.8M	86.3	90.5	93.2	93.0	89.9	90.2	70.3	61.5	84.4	-	-	-	-	-	-
HyperFomer*	638K	85.7	90.0	93.0	94.0	89.7	87.2	75.4	63.7	84.8	-	-	-	-	-	-
HyperDecoder*	1.8M	86.0	90.5	93.4	94.0	90.5	87.7	71.7	55.9	83.7	-	-	-	-	-	-
ATTEMPT*	96K	83.8	90.0	93.1	93.7	90.8	86.1	79.9	64.3	85.2	74.4	78.3	66.5	69.2	82.1	74.1
MPT*	10.5K	84.3	90.0	93.0	93.3	90.4	89.2	82.7	63.5	**85.8**	74.8	79.2	70.2	67.3	89.3	**76.1**

Table 5.1: **Results on GLUE and SuperGLUE.** We adopt Pearson Correlation for STS-B, F1 for MultiRC (Multi), and accuracy for other tasks as evaluation metrics. "param/task" represents number of trainable parameters for each task in GLUE. The top part of the table denotes model adaptation to each target task (so param/task for MPT is just $(l \times d) + (l + d)$). The bottom part (marked by *) denotes model adaptation to a *group* of tasks, where the param/task for MPT * is $(l \times d)/\tau + (l + d)$. See Section 5.3.2 for more details.

5.4.2 Results and Analysis

Full-Dataset Adaptation. Tables 5.1- 5.2 show the per-task performance of different methods on all four benchmarks. As seen from Table 5.1 (top part), MPT establishes new state-of-the-art results for parameter-efficient finetuning on both GLUE and SuperGLUE benchmarks. When compared to vanilla PT (Lester et al., 2021), MPT obtains more than $+10\%$ points improvement on average performance ($+13\%$ on GLUE, $+16\%$ on SuperGLUE) with the same number of task-specific parameters, which demonstrates that multitask prompt tuning provides an effective means of improving the performance of PT, especially on small datasets. MPT consistently outperforms SPoT (Vu et al., 2022) to obtain an average of 85.6% on GLUE and 74.1% on SuperGLUE, which is $+3.3\%$ and $+11.2\%$ point accuracy improvements respectively. Furthermore, our approach achieves 2.1% and 3.6% average accuracy improvements over the recent method, ATTEMPT (Asai

et al., 2022), despite updating $3\times$ fewer parameters on GLUE and SuperGLUE respectively. Similarly, when comparing against BitFit (Zaken et al., 2022) which only tunes the bias vectors, MPT outperforms it by $+2.3\%$ on GLUE and $+3.6\%$ on SuperGLUE, while tuning $2\times$ fewer parameters for each target task. Among the compared methods, MPT is the most competitive in terms of average accuracy on both benchmarks. MPT also outperforms Adapters (Houlsby et al., 2019) on GLUE with $4\times$ fewer task-specific parameters. More surprisingly, our MPT approach outperforms the full model finetuning baseline on both benchmarks, despite tuning 0.035% as many task-specific parameters (see Figures 5.2 for a comparison between different methods versus their number of updated parameters on GLUE and SuperGLUE benchmarks).

When compared with state-of-the-art multitask baselines which train a single model on different target tasks, including recent HyperFormer (Mahabadi et al., 2021) and HyperDecoder (Ivison and Peters, 2022), Table 5.1 (bottom part) shows that our MPT* (w/ prompt decomposition on target tasks) performs well and also further improves upon the single target task baseline. This reveals the potential of our method to further leverage multitask knowledge on the *target* side to enable even more parameter-efficient adaptation of pretrained language models.

Table 5.2 shows the performance of different methods on the MRQA and Others benchmark. Our approach significantly improves the average performance of PT by $+2.8\%$ on MRQA and $+13.5\%$ on the Others benchmark, while adding only 0.01% more task-specific parameters. Similarly, MPT obtains 85.5% average accuracy on WinoGrande, Yelp, SciTail, and PAWS, outperforming BitFit (84.7%), which updates $10\times$ more task-specific parameters. While the improvements achieved by our approach (being highly parameter-efficient) are encouraging on both GLUE and SuperGLUE, the accuracy gap between MPT

Method	param/task	MRQA					Others				
		NQ	HP	SQA	News	Avg.	WG	Yelp	SciTail	PAWS	Avg.
Finetuning	220M	75.1	77.5	81.1	65.2	**74.7**	61.9	96.7	95.8	94.1	**87.1**
Adapter	1.9M	74.2	77.6	81.4	65.6	74.7	59.2	96.9	94.5	94.3	86.2
BitFit	280K	70.7	75.5	77.7	64.1	72.0	57.2	94.7	94.7	92.0	84.7
PT	76.8K	67.9	72.9	75.7	61.1	69.4	49.6	95.1	87.9	55.8	72.1
SPoT	76.8K	68.2	74.8	75.3	58.2	69.1	50.4	95.4	91.2	91.1	82.0
ATTEMPT	232K	70.4	75.2	77.3	62.8	71.4	57.6	96.7	93.1	92.1	84.9
MPT	77.6K	72.0	75.8	77.2	63.7	72.2	56.5	96.4	95.5	93.5	85.5

Table 5.2: **Results on MRQA and Others.** We use F1 for MRQA tasks and accuracy for others as the evaluation metrics. MPT outperforms ATTEMPT on both benchmarks, while tuning 67% less parameters.

k-shot		FT (220M)	AD (1.9M)	PT (76.8K)	ST (76.8K)	HF (638K)	ATP (232K)	MPT (77.6K)
BoolQ	4	50.5	53.4	61.6	50.5	48.0	61.8	**62.2**
	16	56.5	51.4	61.9	50.6	50.2	60.0	**63.3**
	32	58.4	54.5	61.7	61.2	58.3	65.3	**68.9**
CB	4	57.7	51.1	53.5	71.4	60.7	**82.1**	73.6
	16	77.0	74.8	63.5	64.3	76.3	78.5	**78.6**
	32	80.0	74.8	67.8	64.3	81.4	**85.7**	82.1
SciTail	4	79.6	79.5	57.7	69.6	82.0	80.2	**80.2**
	16	80.0	83.2	60.8	71.9	86.5	79.5	**87.3**
	32	81.9	85.0	60.2	71.9	85.8	80.2	**86.3**

Table 5.3: **Few-Shot Results with k = {4, 16, 32}.** FT: Finetuning, AD: Adapter, PT: Prompt tuning, ST: SPoT, HF: HyperFormer, ATP: ATTEMPT. Numbers in bracket denote the number of parameters tuned for each task. Our proposed MPT consistently outperforms PT by a very large margin and competitive or even better than existing methods on majority of the cases, while tuning much fewer task-specific parameters.

and the full finetuning is still significant here (2.2% on MRQA and 1.6% on Others), which indicate opportunities for future work in multitask prompt tuning.

Few-Shot Adaptation. Following prior works (Mahabadi et al., 2021; Asai et al., 2022), in addition to the full dataset adaptation on four benchmarks, we conduct few-shot experiments on BoolQ, CB, and SciTail tasks to measure how pretrained MPT prompts can be

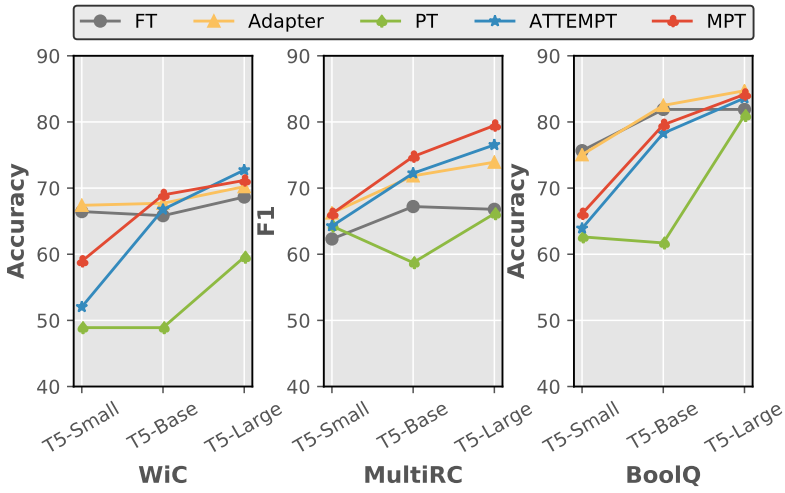

Figure 5.4: **Model Scaling.** With the increase of backbone PLM sizes (from T5-Small to T5-Large), the performance of our proposed MPT is improved consistently across tasks. Best viewed in color.

generalized to new tasks with only a few training examples available ($k = 4, 16, 32$). Table 5.3 shows the results of our approach and other baselines, including full-model finetuning, Adapter, and HyperFormer. As can be seen from Table 5.3, vanilla PT performs poorly in few-shot adaptation (esp., CB and SciTail), suggesting randomly initialized prompts are hard to generalize to new tasks with only a few shots. SPoT improves the performance of PT on CB and SciTail tasks, and MPT outperforms both PT and SPoT. We also observe that other methods in Table 5.3 (Finetuning, Adapter, HyperFormer, and ATTEMPT) have trouble in the few-shot setting. These results indicate that MPT can effectively use cross-task knowledge in source tasks to target tasks where there are only a few labeled examples.

Scaling. We conduct scaling experiments to analyze how MPT performs with increasing pretrained model sizes on three SuperGLUE tasks as in (Asai et al., 2022). Figure 5.4

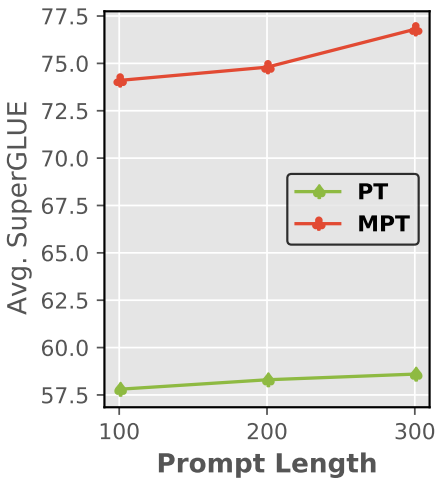

Figure 5.5: Prompt Scaling. Increasing prompt length can effectively boost MPT performance.

shows the performance of MPT as well as full model finetuning (FT), Adapter, prompt tuning (PT), and ATTEMPT with three different T5 models (T5-Small, T5-Base, T5-Large). Performance on WiC, MultiRC, and BoolQ shows that MPT can greatly benefit from scaling up the backbone model and outperforms PT and ATTEMPT consistently across all model sizes. These results show that our prompt decomposition strategy is not only able to achieve the best parameter efficiency but also effective across different model scales ranging from 60M to 770M parameters.

In addition to increasing model sizes, we also increase the length of the prompt l to add more parameters and compare it with vanilla PT on the SuperGLUE benchmark. Figure 5.5 compares PT and MPT over various prompt lengths $l = \{100, 200, 300\}$. From Figure 5.5, we observe that while increasing the prompt length for PT only produces marginal improvement, which is consistent with recent findings in (Lester et al., 2021; Li and Liang,

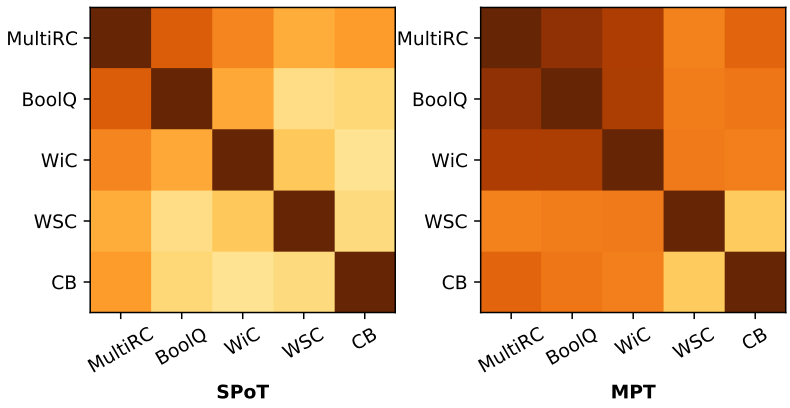

Figure 5.6: Analyzing task correlation using prompt similarities on SuperGLUE.

2021). On the other hand, our MPT approach can get consistent improvements over various lengths of prompts which indicates the potential of increasing prompt length in our multitask prompt tuning to encapsulate massive scale source datasets for learning better transferable prompts. We leave the in-depth exploration of the relationship between prompt capacity and knowledge transfer to future work.

Analyzing Learned Prompts. We conduct qualitative analysis on prompts learned using MPT to investigate whether cross-task knowledge is indeed encoded in the task-shared prompt which makes it easier for target tasks to effectively adapt and encode their own knowledge effectively.

Following (Vu et al., 2022), we leverage task embedding to compute cosine similarities between all target task pairs after adaptation, where each task is represented by the composition of task-shared and task-specific prompts (averaged to obtain a single vector). Figure 5.6 shows the visualization of cosine similarity matrices for SPoT and MPT on SuperGLUE tasks. We find that task embeddings can effectively cluster similar tasks together,

such as MultiRC is similar to BoolQ, and WiC is also close to both of them. Moreover, we also observe that MPT has clearer clusters than SPoT, which verifies our hypothesis that MPT helps target tasks to encode task-specific knowledge more effectively.

5.4.3 Ablation Studies

We conduct extensive ablation studies to measure the importance of various components of MPT and justify several important modeling strategies in our framework in the following subsections.

Effectiveness of Prompt Decomposition. Table 5.4 presents our ablation studeis on SuperGLUE for testing the effect of prompt decomposition and prompt distillation. We fix all the hyper-parameters across all settings and rerun MPT source training to get various ablated versions of the transferred prompt. First, we ablate both prompt decomposition and distillation and initialize a vanilla prompt to be shared across all source tasks (no task-specific vectors). We train it with the simple mixing of all datasets, then transfer the resulting prompt to target tasks for adaptation. Table 5.4 shows that simply training a single soft prompt only produces an average accuracy of 69.5% on SuperGLUE (top row), as it fails to leverage commonalities across source tasks while minimizing interference. To measure the effect of prompt decomposition, we replace the vanilla source prompt with our decomposable prompt with task-shared and task-specific components and train it without prompt distillation (third row in Table 5.4), which gives us 3.5% average performance improvement on SuperGLUE. This ablation clearly indicates the importance of our proposed prompt decomposition strategy in MPT and demonstrates that the shared component can effectively capture the rich cross-task knowledge to be beneficial to target downstream tasks.

Decomposition	Distillation	Avg. Score
✗	✗	69.5
✗	✓	70.6
✓	✗	73.0
✓	✓	74.1

Table 5.4: **Ablation studies on SuperGLUE.**

Effectiveness of Prompt Distillation. To test the effect of prompt distillation, we ablate prompt decomposition and train a vanilla prompt shared by all the source tasks with the same training loss of MPT in Equation 5.5. The teacher models are kept the same for this ablation and MPT. Compared with the simple baseline (first row in Table 5.4), adding prompt distillation (second row) produces 1.1% average performance improvement, which verifies the effectiveness of prompt distillation. With distillation, the vanilla shared prompt can be trained by more fine-grained learning signals from each task, but without prompt decomposition, knowledge of all the source tasks is entangled together, which may hurt transfer performance on target tasks. Finally, we observe that prompt distillation combined with prompt decomposition yields the best average performance of 74.1% on SuperGLUE benchmark. This confirms that distilling knowledge from separately-trained source prompts is an effective strategy for learning good decomposable prompts.

We further investigate the individual components of prompt distillation to see their influences on final performance. We remove the loss of hidden states from Equation 5.5 and find that it produces an average performance of 73.68% on SuperGLUE, verifying the effectiveness of regularizing hidden states in conjunction with logits to reach its full performance, which is consistent with findings in (Sanh et al., 2019). Lastly, we consider a variant of distillation loss to match the teacher and student prompts directly by adding an

MSE loss to minimize the distance between those two prompts. Replacing our proposed distillation losses with this prompt distance loss and jointly training it with prompt decomposition yields an average SuperGLUE performance of 73.6%, which performs worse than the distillation losses based on logits and hidden states.

Ablation on Target Adaptation Strategies. When transferring the shared prompt from source training to target tasks, we ablate MPT with the choices of how to tune the task-shared and task-specific components for target tasks. We find that only updating the task-shared component (i.e., removing target task-specific vectors) or only updating task-specific vectors (i.e., freezing task-shared component) produce unsatisfied results (62.5% and 71.3% SuperGLUE, respectively). This indicates the significance of keeping both components for prompt decomposition on target adaptation.

Effectiveness of Stochastic Task Sampling. Inspired by stochastic depth for training deep neural networks (Huang et al., 2016), we propose a multitask training strategy in Section 5.3.2, which is to stochastically sample a various number of tasks within each mini-batch to help the shared component of MPT be robust to task variances. Ablating this training strategy produces an average performance on SuperGLUE as 73.66%, which verifies the importance of this simple multitask training strategy.

5.4.4 MPT for NLG Tasks

We follow the standard evaluation protocol of prior works (Asai et al., 2022; Vu et al., 2022) to conduct our experiments on GLUE, SuperGLUE, MRQA and Other benchmarks. In this section, we conduct additional experiments on NLG tasks by applying T5-MPT source prompt on target NLG tasks. In particular, we transfer the T5-Large prompt trained using 6 diverse source tasks used in our current experiments for adaptation to two target

	E2E					WebNLG		
	BLEU	NIST	METEOR	Rouge-L	CIDEr	BLEU	METEOR	TER (↓)
PT	29.11	5.00	0.343	51.50	1.72	46.02	0.37	46.89
MPT	32.14	5.35	0.363	52.88	1.86	52.27	0.40	41.36

Table 5.5: Applying MPT-T5-Large prompts to NLG tasks. MPT consistently outperforms PT on both tasks.

	MRQA					Others				
	NQ	HP	SQA	News	Avg.	WG	Yelp	SciTail	PAWS	Avg.
MPT (w/ 6 Source Tasks)	72.0	75.8	77.2	63.7	72.2	56.5	96.4	95.5	93.5	85.5
MPT (w/ 12 Source Tasks)	72.1	76.4	77.9	64.0	72.6	56.6	96.8	95.9	92.9	85.6

Table 5.6: MPT performance on MRQA and Others with more number of source tasks.

tasks, namely E2E (Novikova et al., 2017) and WebNLG (Gardent et al., 2017). Table 5.5 shows that our proposed MPT significantly outperforms standard PT (Lester et al., 2021) on both NLG tasks across all the metrics. Our BLEU improvements over PT are 3.03% and 6.25% on E2E and WebNLG tasks respectively, showing the effectiveness of our approach on both NLU (e.g., classification, NLI, QA tasks) and NLG tasks. This is particularly an impressive result since the source tasks were all NLU tasks, i.e., MPT can transfer knowledge from NLU tasks to NLG tasks.

5.4.5 MPT with more number of source tasks

Following (Asai et al., 2022), we select 6 representative NLP tasks as source datsks, including 2 NLI, 1 paraphrase, 1 sentiment analysis and 2 large-scale QA tasks, which are general/diverse enough and can enable knowledge transfer to other tasks. In this section, we investigate the effect of adding more remotely relevant source tasks to our proposed

		MRQA					Others				
	param/task	NQ	HP	SQA	News	Avg.	WG	Yelp	SciTail	PAWS	Avg.
Fine-tuning	220M	75.1	77.5	81.1	65.2	74.7	61.9	96.7	95.8	94.1	87.1
Adapter	1.9M	74.2	77.6	81.4	65.6	74.7	59.2	96.9	94.5	94.3	86.2
MPT-100	77.6K	72.0	75.8	77.2	63.7	72.2	56.5	96.4	95.5	93.5	85.5
MPT-300	231.5K	72.6	76.4	78.4	64.3	73.0	57.0	97.0	96.8	93.8	86.1

Table 5.7: Performance on MRQA and Others benchmark by scaling prompt length. All results are based on T5-Base. MPT-300 is very competitive to Adapter on Others benchmaek while being highly parameter-efficient.

multitask prompt tuning. Specifically, we add 6 additional diverse source tasks, including one topic classification (AGNews (Zhang et al., 2015)), three multi-choice QA (CommonsenseQA (Talmor et al., 2018), OpenBookQA (Mihaylov et al., 2018), ARC (Clark et al., 2018)), one adversarial NLI (ANLI (Nie et al., 2020)) and one commonsense (winogrande (Sakaguchi et al., 2021)) datasets. Table 5.6 shows the results on MRQA and Others benchmarks. As can be seen, MPT with 12 tasks is still very effective for target adaptation on both benchmarks, slightly outperforming MPT trained using 6 tasks.

Furthermore, it would be compelling to use benchmarks like CrossFit (Ye et al., 2021a) consisting of 160 NLP tasks as source tasks for analyzing the performance of MPT on parameter-efficient transfer learning. While we currently do not possess the compute resources for this extreme large-scale study, we hope to cover this an interesting future work. Last but not least, we will release pretrained source task prompts and easily extendable code so that it can motivate further studies on task scaling and understanding transferabilty across a more diverse set of source and target tasks.

5.4.6 Prompt Scaling for MRQA and Others

Our proposed MPT outperforms both Finetuning and Adapter baslines on GLUE and SuperGLUE benchmarks (Table 5.1). However, they're still better than MPT on MRQA and Others benchmarks (Table 5.2) with 2832 and 24 times more parameters than MPT, respectively. While adding the same number of prompt parameters as the Adapter to close the performance gap on MRQA and Others benchmarks is an interesting suggestion, we note that it requires a prompt length of 2400 tokens on T5-Base, which can be computationally inefficient due to transformer's quadratic complexity with the input length. Table 5.7 shows that increasing prompt length from 100 to 300 yields an average improvement of 0.8% on MRQA and 0.6% on Others, further closing the gap between MPT and Adapters (e.g., only 0.1% difference in Others benchmark). We also test with a prompt length of 400 tokens but did not notice any significant improvements. We believe this is because the optimal prompt length in our current experiments is around 300 tokens as discussed in our prompt scaling analysis. Applying MPT for every layer of the pretrained model, instead of only input layer (like P-Tuning v2 (Liu et al., 2021b)) could be a promising direction to further improve the performance: we leave this as an interesting future work.

5.4.7 Few-Shot Results on GLUE and SuperGLUE

Following (Asai et al., 2022), we conduct few-shot experiments on BoolQ, CB, and SciTail tasks for a fair and direct comparison with other parameter-efficient methods, namely SpoT (Vu et al., 2022), ATTEMPT (Asai et al., 2022) and Hyperformer (Mahabadi et al., 2021). In this section, we conduct more comprehensive few-shot experiments on all the GLUE and SuperGLUE tasks by comparing vanilla PT (Lester et al., 2021) and MPT. As shown in Table 5.8, we can observe that not only MPT outperforms the vanilla PT by a

k-shot		MNLI	QQP	QNLI	SST-2	STS-B	MRPC	RTE	CoLA	Avg.	Multi	BoolQ	WiC	WSC	CB	Avg.
4	PT	40.1	63.2	40.4	53.0	88.8	68.1	56.3	27.4	54.7	61.8	61.6	51.2	60.4	53.5	57.7
	MPT	59.4	82.0	86.2	56.5	89.1	68.1	62.6	34.8	67.3	62.2	62.2	52.9	67.3	73.6	63.6
16	PT	41.5	62.3	59.9	50.9	87.8	68.1	54.7	28.5	56.7	60.3	61.9	48.9	44.2	63.5	55.8
	MPT	61.6	84.7	90.6	63.2	89.1	70.1	64.8	32.1	69.5	64.5	63.3	49.8	67.3	78.6	64.7
32	PT	37.0	62.3	56.7	50.9	87.5	68.1	54.7	23.2	55.1	59.2	61.7	52.6	67.3	67.8	61.7
	MPT	63.6	88.5	91.0	75.9	89.7	74.5	59.7	30.8	71.7	63.3	68.9	53.9	67.3	82.1	67.1

Table 5.8: Few-Shot results on GLUE and SuperGLUE with $k = \{4, 16, 32\}$. MPT consistently outperforms PT, demonstrating generalizability of MPT prompts to new tasks with only a few training examples.

large margin in most of the datasets, but also MPT can perform very well on many datasets to reach their full-dataset performance with 16 or 32 shots, such as QQP, QNLI, STS-B, and WSC. These results clearly indicate that MPT can effectively use cross-task knowledge in source tasks to target tasks where there are only a few labeled examples.

5.5 Discussion and Conclusion

We introduced and studied multitask prompt tuning (MPT), which learns a single transferable prompt by decomposing and distilling knowledge from multiple source tasks as well as their task-specific source prompts. MPT decomposes the task prompt by the Hadamard product of a shared prompt matrix and a rank-one task-specific matrix. The shared component is then transferred and adapted to target tasks to be further tuned. Empirically we found this approach enables parameter-efficient transfer learning to target downstream tasks across diverse NLP benchmarks, even outperforming the full finetuning baseline in some cases despite tuning much fewer task-specific parameters.

Specifically, Our current MPT is built on top of prompt tuning (Lester et al., 2021), which is mostly applied to T5 (Raffel et al., 2020). So, we follow prior works, such as

SPoT (Vu et al., 2022) and ATTEMPT (Asai et al., 2022), to conduct experiments on T5-variants. However, our proposed approach is quite generic and can be applied to both T5 and GPT models. This is primarily because MPT only prepends a prompt matrix (i.e., virtual tokens) to the input embedding layer and hence can be adapted to any transformer models (encoder-only, encoder-decoder, or decoder-only), not limited to T5. Specifically, MPT focuses on decomposing the prompt matrix into task-specific and task-shared components, which introduces minimal intrusion to the backbone model. Similarly, the distillation part of MPT is also model-agnostic and can be generalized to GPT models. Straightforward alternative architectures can extend this to BERT-based prompt tuning, such as P-Tuning-v2 (Liu et al., 2021b) and GPT-based prompt tuning, such as prefix-tuning (Li and Liang, 2021; Clive et al., 2021).

Moreover, our key intuition of learning rich cross-task shared knowledge across multiple generic NLP tasks can be extended to dictionary learning. We first group NLP tasks based on their task description and similarity, and then apply MPT to each group to learn a basis prompt. After learning basis prompts for all NLP tasks, including text classification, natural language inference, question answering, text generation, etc., we can ensemble them for more efficient fine-tuning on any NLP target tasks.

Part V: Reasoning: Explicit Reasoning for Interpretable Machine Learning

Chapter 6: Rationalizing Medical Relation Prediction from Corpus-level Statistics

Finally, we turn to the last stage of the life cycle of incorporating human knowledge into AI systems, knowledge reasoning. Knowledge reasoning aims to enable more transparent and generalizable prediction by performing human-like complex reasoning over structured knowledge. This stage mimics how humans can reason by understanding the relationships between different facts and drawing logical conclusions, which is necessary to solve complex problems, such as mathematical ones. It is challenging but especially important to increase the interpretability of existing AI systems that mostly turn out to be black-box models nowadays. It provides a reliable and human-understandable way to explain the internal decision-making process of AI systems, which is beneficial for developing trust and confidence in AI-based applications, especially in high-risk domains. This chapter proposes a self-explainable framework that can generate human-intuitive rationales for relation prediction tasks. The generated rationales are used to justify the model decision and increase user trust.

Specifically, the interpretability of machine learning models is becoming increasingly important, especially in the medical domain. Aiming to shed some light on how to rationalize medical relation prediction, we present a new framework inspired by existing theories

on how human memory works, e.g., theories of recall and recognition. Given the corpus-level statistics, i.e., a global co-occurrence graph of a clinical text corpus, to predict the relations between two entities, we first *recall* rich contexts associated with the target entities, and then *recognize* relational interactions between these contexts to form model rationales, which will contribute to the final prediction. We conduct experiments on a real-world public clinical dataset and show that our framework can not only achieve competitive predictive performance against a comprehensive list of neural baseline models but also present rationales to justify its predictions. We further collaborate with medical experts deeply to verify the usefulness of our model rationales for clinical decision-making. Our implementation is available on Github: https://github.com/zhenwang9102/X-MedRELA.

6.1 Introduction

Predicting relations between entities from a text corpus is a crucial task in order to extract structured knowledge, which can empower a broad range of downstream tasks, e.g., question answering (Xu et al., 2016), dialogue systems (Lowe et al., 2015), reasoning (Das et al., 2017b), etc. There has been a large amount of existing work focusing on predicting relations based on *raw texts* (e.g., sentences, paragraphs) mentioning two entities (Hendrickx et al., 2010; Zeng et al., 2014; Zhou et al., 2016; Mintz et al., 2009; Riedel et al., 2010; Lin et al., 2016; Verga et al., 2018; Yao et al., 2019).

In this chapter, we study a relatively new setting in which we predict relations between entities based on the *global co-occurrence statistics* aggregated from a text corpus, and focus on medical relations and clinical texts in Electronic Medical Records (EMRs). The corpus-level statistics present a *holistic graph view* of all entities in the corpus, which will greatly facilitate the relation inference, and can better preserve patient privacy than raw or

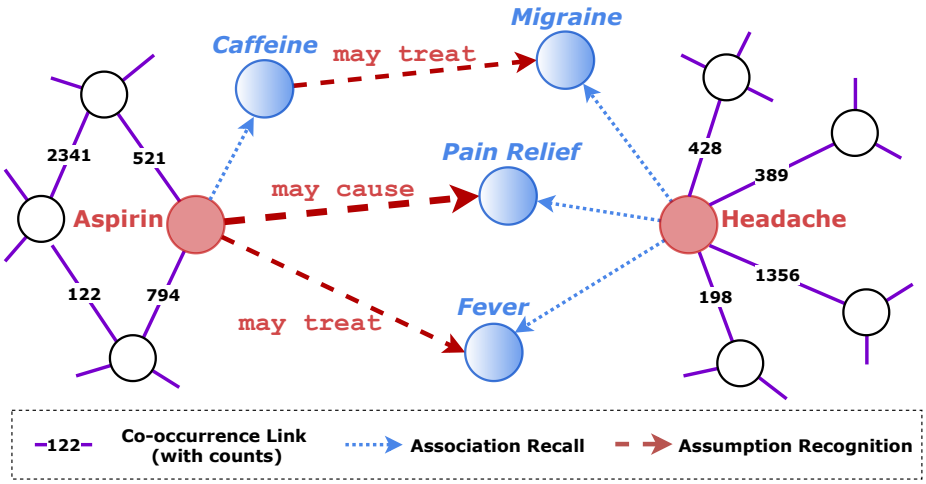

Figure 6.1: Our intuition for how to rationalize relation prediction based on the corpus-level statistics. To infer the relation between the target entities (red nodes), we recall (blue dashed line) their associated entities (blue nodes) and infer their relational interactions (red dashed line), which will serve as assumptions or model rationales to support the target relation prediction.

even de-identified textual content and are becoming a popular substitute for the latter in the research community for studying EMR data (Finlayson et al., 2014; Wang et al., 2019b).

To predict relations between entities based on a global co-occurrence graph, intuitively, one can first optimize the graph embedding or global word embedding (Pennington et al., 2014; Perozzi et al., 2014; Tang et al., 2015), and then develop a relation classifier (Nickel et al., 2011; Socher et al., 2013a; Yang et al., 2015; Wang et al., 2018c) based on the embedding vectors of the two entities. However, such kind of neural frameworks often lack the desired *interpretability*, which is especially important for the medical domain. In general, despite their superior predictive performance in many NLP tasks, the opaque decision-making process of neural models has concerned their adoption in high stakes domains like medicine, finance, and judiciary (Rudin, 2019; Murdoch et al., 2019). Building models that provide reasonable explanations and have increased transparency can remarkably enhance

user trust (Ribeiro et al., 2016; Miller, 2019). In this chapter, we aim to develop such a model for our medical relation prediction task.

To start with, we draw inspiration from the existing theories on cognitive processes about how human memory works, e.g., two types of memory retrieval (recall and recognition) (Gillund and Shiffrin, 1984). Basically, in the *recall* process, humans tend to retrieve contextual associations from long-term memory. For example, given the word "Paris", one may think of "Eiffel Tower" or "France", which are strongly associated with "Paris" (Nobel and Shiffrin, 2001; Kahana et al., 2008; Budiu, 2014). Besides, there is a strong correlation between the association strength and the co-occurrence graph (Spence and Owens, 1990; Lundberg and Lee, 2017). In the *recognition* process, humans typically recognize if they have seen a certain piece of information before. Figure 6.1 shows an example in the context of relation prediction. Assume a model is to predict whether *Aspirin* may treat *Headache* or not (That "*Aspirin* may treat *Headache*" is a known fact, and we choose this relation triple for illustration purposes). It is desirable if the model could perform the aforementioned two types of memory processes and produce rationales to base its prediction upon: (1) Recall. What entities are associated with *Aspirin*? What entities are associated with *Headache*? (2) Recognition. Do those associated entities hold certain relations, which can be leveraged as clues to predict the target relation? For instance, a model could first retrieve a relevant entity *Pain Relief* for the tail entity *Headache* as they co-occur frequently, and then recognize there is a chance that *Aspirin* can lead to *Pain Relief* (i.e., formulate model rationales or assumptions), based on which it could finally make a correct prediction (*Aspirin*, may treat, *Headache*).

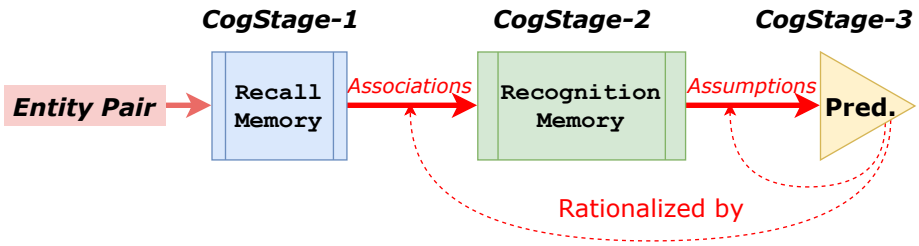

Figure 6.2: A high-level illustration of our framework.

Now we formalize such intuition to rationalize the relation prediction task. Our framework consists of three stages, *global association recall* (CogStage-1), *assumption formation and representation* (CogStage-2), and *prediction decision making* (CogStage-3), shown in Figure 6.2. CogStage-1 models the process of recalling diverse contextual entities associated with the target head and tail entities respectively, CogStage-2 models the process of recognizing possible interactions between those recalled entities, which serve as model rationales (or, assumptions[22]) and are represented as semantic vectors, and finally CogStage-3 aggregates all assumptions to infer the target relation. We jointly optimize all three stages using a training set of relation triples as well as the co-occurrence graph. Model rationales can be captured through this process *without any gold rationales* available as direct supervision. Overall, our framework rationalizes its relation prediction and is interpretable to users[23] by providing justifications for (i) why a particular prediction is made, (ii) how the assumptions of the prediction are developed, and (iii) how the particular assumptions are relied on.

[22]We use the two terms interchangeably in this chapter.

[23]Following (Murdoch et al., 2019), desired interpretability is supposed to provide insights to particular audiences, which in our case are medical experts.

On a real-life clinical text corpus, we compare our framework with various competitive methods to evaluate the predictive performance and interpretability. We show that our method obtains very competitive performance compared with a comprehensive list of various neural baseline models. Moreover, we follow recent work (Singh et al., 2019; Jin et al., 2020) to quantitatively evaluate model interpretability and demonstrate that rationales produced by our framework can greatly help earn expert trust. To summarize, we study the important problem of rationalizing medical relation prediction based on corpus-level statistics and propose a new framework inspired by cognitive theories, which outperforms competitive baselines in terms of both interpretability and predictive performance.

6.2 Related Work

Relation Extraction (RE) typically focuses on predicting relations between two entities based on their text mentions, and has been well studied in both open domain (Mintz et al., 2009; Zeng et al., 2015; Riedel et al., 2013; Lin et al., 2016; Song et al., 2019; Deng and Sun, 2019) and biomedical domain (Uzuner et al., 2011; Wang and Fan, 2014; Sahu et al., 2016; Lv et al., 2016; He et al., 2019). Among them, most state-of-the-art work develops various powerful neural models by leveraging human annotations, linguistic patterns, distance supervision, etc. More recently, an increasing amount of work has been proposed to improve model's transparency and interpretability. For example, (Lee et al., 2019) visualizes self-attention weights learned from BERT (Devlin et al., 2019) to explain relation prediction. However, such text-based interpretable models tend to provide explanations within a *local context* (e.g., words in a single sentence mentioning target entities), which may not capture a *holistic view* of all entities and their relations stored in a text corpus. We believe that such a holistic view is important for interpreting relations and can be provided

to some degree by the *global statistics* from a text corpus. Moreover, global statistics have been widely used in the clinical domain as they can better preserve patient privacy (Finlayson et al., 2014; Wang et al., 2019b).

On the other hand, in recent years, graph embedding techniques (Perozzi et al., 2014; Tang et al., 2015; Grover and Leskovec, 2016b; Yue et al., 2019) have been widely applied to learn node representations based on graph structure. Representation learning based on global statistics from a text corpus (i.e., co-occurrence graph) has also been studied (Levy and Goldberg, 2014b; Pennington et al., 2014). After employing such methods to learn entity embeddings, a number of relation classifiers (Nickel et al., 2011; Bordes et al., 2013; Socher et al., 2013a; Yang et al., 2015; Wang et al., 2018c) can be adopted for relation prediction. We compare our method with such frameworks to show its competitive predictive accuracy. However, such frameworks tend to be difficult to interpret as they provide little or no explanations on how decisions are made. In this chapter, we focus more on model interpretability than predictive accuracy, and draw inspirations from existing cognitive theories of recall and recognition to develop a new framework, which is our core contribution.

Another line of research related to interpreting relation prediction is path-based knowledge graph (KG) reasoning (Gardner et al., 2014; Neelakantan et al., 2015; Guu et al., 2015; Xiong et al., 2017; Stadelmaier and Padó, 2019). In particular, existing paths mined from millions of relational links in knowledge graphs can be used to provide justifications for relation predictions. For example, to explain *Microsoft* and *USA* may hold the relation *CountryOfHeadquarters*, by traversing a KG, one can extract the path *Microsoft* $\xrightarrow{\text{IsBasedIn}}$ *Seattle* $\xrightarrow{\text{CountryLocatedIn}}$ *USA* as one explanation. However, such path-finding methods typically require large-scale relational links to infer path patterns, and cannot be applied to our co-occurrence graph as the co-occurrence links are unlabeled.

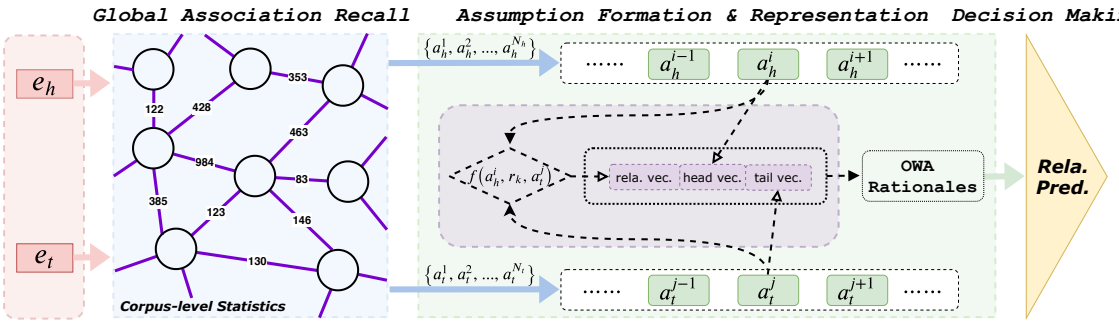

Figure 6.3: Framework Overview.

In addition, our work is closely related to the area of rationalizing machine decision by generating justifications/rationales accounting for model's prediction. In some scenarios, human rationales are provided as extra supervision for more explainable models (Zaidan et al., 2007; Bao et al., 2018). However, due to the high cost of manual annotation, model rationales are desired to be learned in an unsupervised manner(Lei et al., 2016; Bouchacourt and Denoyer, 2019; Zhao et al., 2019). For example, (Lei et al., 2016) select a subset of words as rationales and (Bouchacourt and Denoyer, 2019) provide an explanation based on the absence or presence of "concepts", where the selected words and "concepts" are learned unsupervisedly. Different from text-based tasks, in this chapter, we propose to rationalize relation prediction based on global co-occurrence statistics and similarly, model rationales in our work are captured without explicit manual annotation either, via a joint training framework.

6.3 Preliminaries

Different from existing work using raw texts for relation extraction, we assume a global co-occurrence graph (i.e., corpus-level statistics) is given, which was pre-constructed based

on a text corpus $\mathcal{D}$, and denote it as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each vertex $v \in \mathcal{V}$ represents an entity extracted from the corpus and each edge $e \in \mathcal{E}$ is associated with the global co-occurrence count for the connected nodes. Counts reflect how frequent two entities appear in the same context (e.g., co-occur in the same sentence, document, or a certain time frame). In this chapter, we focus on clinical co-occurrence graph in which vertices are medical terms extracted from clinical notes. Nevertheless, as we will see later, our framework is very general and can be applied to other relations with corpus-level statistics.

Our motivation for working under this setting lies in three folds: (1) Such graph data is stripped of raw textual contexts and thus, has a better preserving of patient privacy (Wang et al., 2019b), which makes itself easier to be constructed and shared under the HIPPA protected environments (Act, 1996) for medical institutes (Finlayson et al., 2014); (2) Compared with open-domain relation extraction, entities holding a medical relation oftentimes do not co-occur in a local context (e.g., a sentence or paragraph). For instance, we observe that in a widely used clinical co-occurrence graph (Finlayson et al., 2014), which is also employed for our experiments later, of all entity pairs holding the treatment relation according to UMLS (Unified Medical Language System), only about 11.4% have a co-occurrence link (i.e., co-occur in clinical notes within a time frame like 1 day or 7 days); (3) As suggested by cognitive theories (Spence and Owens, 1990), lexical co-occurrence is significantly correlated with association strength in the recall memory process, which further inspires us to utilize such statistics to find associations and form model rationales for relation prediction.

Finally, our relation prediction task is formulated as: Given the global statistics $\mathcal{G}$ and an entity pair, we predict whether they hold a relation r (e.g., MAY_TREAT), and moreover provide a set of model rationales $\mathcal{T}$ composed of relation triples for the prediction. For

the example in Figure 6.1, we aim to build a model that will not only accurately predict the MAY_TREAT relation, but also provide meaningful rationales on how the prediction is made, which are crucial for gaining trust from clinicians.

6.4 Methodology

Following a high-level framework illustration in Figure 6.2, we show a more detailed overview in Figure 6.3 and introduce each component as follows.

6.4.1 CogStage-1: Global Association Recall

Existing cognitive theories (Kahana et al., 2008) suggest that *recall* is an essential function of human memory to retrieve *associations* for later decision making. On the other hand, the association has been shown to significantly correlate with the lexical co-occurrence from the text corpus (Spence and Owens, 1990; Lund and Burgess, 1996). Inspired by such theories and correlation, we explicitly build up our model based on recalled associations stemming from corpus-level statistics and provide global highly-associated contexts as the source of interpretations.

Given an entity, we build an estimation module to globally infer associations based on the corpus-level statistics. Our module leverages distributional learning to fully explore the graph structure. One can also directly utilize the raw neighborhoods in the co-occurrence graph, but due to the noise introduced in the preprocessing of building the graph, it is a less optimal choice in real practice.

Specifically, for a selected node/entity $e_i \in \mathcal{E}$, our global association recall module estimates a conditional probability $p(e_j|e_i)$, representing how likely the entity $e_j \in \mathcal{E}$ is

associated with e_i[24]. We formally define such conditional probability as:

$$p\left(e_j|e_i\right) = \frac{\exp\left(\boldsymbol{v}'^{T}_{e_j} \cdot \boldsymbol{v}_{e_i}\right)}{\sum_{k=1}^{|\mathcal{V}|} \exp\left(\boldsymbol{v}'^{T}_{e_k} \cdot \boldsymbol{v}_{e_i}\right)} \tag{6.1}$$

where $\boldsymbol{v}_{e_i} \in \mathbb{R}^d$ is the embedding vector of node e_i and $\boldsymbol{v}'_{e_j} \in \mathbb{R}^d$ is the context embedding for e_j.

There are many ways to approximate $p\left(e_j|e_i\right)$ from the global statistics, e.g., using global log-bilinear regression (Pennington et al., 2014). To estimate such probabilities and update entity embeddings efficiently, we optimize the conditional distribution $p\left(e_j|e_i\right)$ to be close to the empirical distribution $\hat{p}\left(e_j|e_i\right)$ defined as:

$$\hat{p}\left(e_j|e_i\right) = \frac{p_{ij}}{\sum_{(i,k)\in\mathcal{E}} p_{ik}} \tag{6.2}$$

where $\mathcal{E}$ is the set of edges in the co-occurrence graph and p_{ij} is the PPMI value calculated by the co-occurrence counts between node e_i and e_j. We adopt the cross entropy loss for the optimization:

$$\mathcal{L}_n = -\sum_{(e_i,e_j)\in\mathcal{V}} \hat{p}(e_j|e_i) \log\left(p(e_j|e_i)\right) \tag{6.3}$$

This association recall module will be jointly trained with other objective functions to be introduced in the following sections. After that, given an entity e_i, we can select the top-N_c entities from $p(\cdot|e_i)$ as e_i's associative entities for subsequent assumption formation.

6.4.2 CogStage-2: Assumption Formation and Representation

As shown in Figure 6.3, with the associative entities from CogStage-1, we are ready to *formulate* and *represent* assumptions. In this chapter, we define model assumptions as *relational interactions between associations*, that is, as shown in Figure 6.1, the model

[24]We assume all existing entities can be possible associations for the given entity.

may identify (*Caffeine*, MAY_TREAT, *Migraine*) as an assumption, which could help predict *Aspirin* may treat *Headache* (*Caffeine* and *Migraine* are associations for *Aspirin* and *Headache* respectively). Such relational rationales are more concrete and much easier for humans to understand than the widely-adopted explanation strategy (Yang et al., 2016b; Mullenbach et al., 2018; Vashishth et al., 2019) in NLP that is based on pure attention weights on local contexts.

One straightway way to obtain such rationales is to query existing medical knowledge bases (KBs), e.g., (*Caffeine*, MAY_TREAT, *Migraine*) may exist in SNOMED CT[25] and can serve as a model rationale. We refer to rationales acquired in this way as the *Closed-World Assumption* (CWA) (Reiter, 1981) setting since only KB-stored facts are considered and trusted in a closed world. In contrast to the CWA rationales, considering the sparsity and incompleteness issues of KBs that are even more severe in the medical domain, we also propose the *Open-World Assumptions* (OWA) (Ceylan et al., 2016) setting to discover richer rationales by estimating all potential relations between associative entities based on a seed set of relation triples (which can be regarded as prior knowledge).

In general, the CWA rationales are relatively more accurate as each fact triple has been verified by the KB, but would have a low coverage of other possibly relevant rationales for the target prediction. On the other hand, the OWA rationales are more comprehensive but could be noisy and less accurate, due to the probabilistic estimation procedure and the limited amount of prior knowledge. However, as we will see, by aggregating all OWA rationales into the whole framework with an attention-based mechanism, we can select high-quality and most relevant rationales for prediction. For the rest of the chapter, by default we adopt the OWA setting in our framework and describe its details as follows.

[25] https://www.snomed.org/

Specifically, given a pair of head and tail entity, $e_h, e_t \in \mathcal{V}$, let us denote their association sets as $\mathcal{A}(e_h) = \{a_h^i\}_{i=1}^{N_h}$ and $\mathcal{A}(e_t) = \{a_t^j\}_{j=1}^{N_t}$, where N_h, N_t are the number of associative entities a_h, a_t to use. Each entity has been assigned an embedding vector by the previous association recall module. We first measure the probability of relations holding for the pair. Given $a_h^i \in \mathcal{A}(e_h), a_t^j \in \mathcal{A}(e_t)$ and a relation $r_k \in \mathcal{R}$, we define a scoring function as (Bordes et al., 2013) to estimate triple quality:

$$s_k^{ij} = f(a_h^i, r_k, a_t^j) = -||\boldsymbol{\upsilon}_{a_h^i} + \boldsymbol{\xi}_k - \boldsymbol{\upsilon}_{a_t^j}||_1 \tag{6.4}$$

where $\boldsymbol{\upsilon}_{a_h^i}$ and $\boldsymbol{\upsilon}_{a_t^j}$ are embedding vectors, relations are parameterized by a relation matrix $R \in \mathbb{R}^{N_r \times d}$ and $\boldsymbol{\xi}_k$ is its k-th row vector. Such a scoring function encourages larger value for correct triples. Additionally, in order to filter unreliable estimations, we define an NA relation to represent other trivial relations or no relation as the score, $s_{\text{NA}}^{ij} = f(a_h^i, \text{NA}, a_t^j)$, which can be seen as a dynamic threshold to produce reasonable rationales.

Now we *formulate* OWA rationales by calculating the conditional probability of a relation given a pair of associations as follows (we save the superscript ij for space):

$$p(r_k|a_h^i, a_t^j) = \begin{cases} \dfrac{\exp(s_k)}{\sum_{s_k \geq s_{\text{NA}}} \exp(s_k)}, & s_k > s_{\text{NA}} \\ 0, & s_k \leq s_{\text{NA}} \end{cases} \tag{6.5}$$

For each association pair, (a_h^i, a_t^j), we only form an assumption with a relation r_k^* if r_k^* is top ranked according to $p(r_k|a_h^i, a_t^j)$.[26]

To *represent* assumptions, we integrate all relation information per pair into a single vector representation. Concretely, we calculate the assumption representation by treating $p(r_k|a_h^i, a_t^j)$ as weights for all relations as follows:

$$\mathsf{a}_{ij} = \rho(a_h^i, a_t^j; \mathcal{R}) = \sum_{k'=1}^{N_r} p(r_{k'}|a_h^i, a_t^j) \cdot \boldsymbol{\xi}_{k'} \tag{6.6}$$

[26]We remove the target relation to predict if it exists in the assumption set.

Finally, we combine the entity vectors as well as the relation vector to get the final representation of assumptions for association pair (a_h^i, a_t^j), where $c_i \in \mathcal{A}(e_h)$ and $c_j \in \mathcal{A}(e_t)$:

$$\mathsf{e}_{ij} = \tanh([\boldsymbol{v}_{a_h^i}; \boldsymbol{v}_{a_t^j}; \mathsf{a}_{ij}]\boldsymbol{W}_p + \boldsymbol{b}_p) \tag{6.7}$$

where $[\cdot\,;\cdot]$ represents vector concatenation, $\boldsymbol{W}_p \in \mathbb{R}^{3d \times d_p}$, $\boldsymbol{b}_p \in \mathbb{R}^{d_p}$ are the weight matrix and bias in a fully-connected network.

6.4.3 CogStage-3: Prediction Decision Making

Analogical to human thinking, our decision making module aggregates all assumption representations and measures their accountability for the final prediction. It learns a distribution over all assumptions and we select the ones with highest probabilities as model rationales. More specifically, we define a scoring function $g(\mathsf{e}_{ij})$ to estimate the accountability based on the assumption representation e_{ij} and normalize $g(\mathsf{e}_{ij})$ as:

$$g(\mathsf{e}_{ij}) = \boldsymbol{v}^T \cdot \tanh(\boldsymbol{W}_a \mathsf{e}_{ij} + \boldsymbol{b}_a) \tag{6.8}$$

$$p_{ij} = \frac{\exp(g(\mathsf{e}_{ij}))}{\sum_{m=1}^{N_h} \sum_{n=1}^{N_t} \exp(g(\mathsf{e}_{mn}))} \tag{6.9}$$

where $\boldsymbol{W}_a, \boldsymbol{b}_a$ are the weight matrix and bias for the scoring function. Then we get the weighted rationale representation as:

$$\mathsf{r} = \psi(e_h, e_t) = \sum_{i=1}^{N_h} \sum_{j=1}^{N_t} p_{ij} \mathsf{e}_{ij} \tag{6.10}$$

With the representation of weighted assumption information for the target pair (e_h, e_t), we calculate the binary prediction probability for relation r as:

$$p(r|e_h, e_t) = \sigma(\boldsymbol{W}_r \mathsf{r} + \boldsymbol{b}_r) \tag{6.11}$$

where $\sigma(x) = 1/(1 + \exp(-x))$ and $\boldsymbol{W}_r, \boldsymbol{b}_r$ are model parameters.

Rationalizing relation prediction. After fully training the entire model, to recover the most contributing assumptions for predicting the relation between the given target entities (e_h, e_t), we compute the importance scores for all assumptions and select those most important ones as model rationales. In particular, we multiply p_{ij} (the weight for association pair (a_h^i, a_t^j) in Eqn. 6.9) with $p(r_k|a_h^i, a_t^j)$ (the probability of a relation given the pair (a_h^i, a_t^j) in Eqn. 6.5) to score the triple (a_h^i, r_k, a_t^j). We rank all such triples for $a_h^i \in \mathcal{A}(e_h), a_t^j \in \mathcal{A}(e_t), r_k \in \mathcal{R}$ and select the top-K triples as model rationales for the final relation prediction.

6.4.4 Training

We now describe how we train our model efficiently for multiple modules. For relational learning to estimate the conditional probability $p(r_k|a_h^i, a_t^j)$, we utilize training data as the seed set of triples for all relations as correct triples denoted as $(h, r, t) \in \mathcal{P}$. The scoring function in Eqn. 6.4 is expected to score higher for correct triples than the corrupted ones in which we denote $\mathcal{N}(?, r, t)$ ($\mathcal{N}(t, r, ?)$) as the set of corrupted triples by replacing the head (tail) entity randomly. Instead of using margin-based loss function, we adopt a more efficient training strategy from (Kadlec et al., 2017; Toutanova and Chen, 2015) with a negative log likelihood loss function as:

$$\begin{aligned} \mathcal{L}_r = &- \textstyle\sum_{(h,r,t)\in\mathcal{P}} \log p\left(h|t,r\right) \\ &- \textstyle\sum_{(h,r,t)\in\mathcal{P}} \log p\left(t|h,r\right) \end{aligned} \tag{6.12}$$

where the conditional probability $p(h|t, r)$ is defined as follows ($p(t|h, r)$ is defined similarly):

$$p\left(h|t,r\right) = \frac{\exp(f\left(h,r,t\right))}{\sum_{h'\in\mathcal{N}(?,r,t)} \exp(f\left(h',r,t\right))} \tag{6.13}$$

For our binary relation prediction task, we define a binary cross entropy loss function with Eqn. 6.11 as follows:

$$\begin{aligned}\mathcal{L}_p = -\sum_{i=1}^{M}(y_i \cdot \log(p(r|e_h^i, e_t^i)) \\ + (1-y_i) \cdot \log(1-p(r|e_h^i, e_t^i)))\end{aligned} \tag{6.14}$$

where M is the number of samples, y_i is the label showing whether e_h, e_t holds a certain relation.

The above three loss functions, i.e., $\mathcal{L}_n$ for global association recall, $\mathcal{L}_r$ for relational learning and $\mathcal{L}_p$ for relation prediction, are all jointly optimized. All three of them share the entity embeddings and $\mathcal{L}_p$ will reuse the relation matrix from $\mathcal{L}_r$ to conduct the rationale generation. Our training algorithm is displayed at Algorithm 1

Algorithm 1 CogStage Training Algorithm

INPUT: Corpus Statistics $\mathcal{G}$, Gold Triples $\mathcal{P}$, Binary Relation Data $\{(h_k, t_k), y_k\}_{k=1}^{M}$
OUTPUT: Model parameters

repeat
  Sample $\{e_i\}_{i=1}^{b_1}$ with gold contexts from $\mathcal{G}$
  for $i \leftarrow 1 : b_1$ **do**
    Calculate $p(e_j|e_i)$ and $\hat{p}(e_j|e_i)$
    Optimize $\mathcal{L}_n$ by Eqn. 6.3
  Sample $\{(h_i, r_i, t_i)\}_{i=1}^{b_2}$ from $\mathcal{P}$
  for $i \leftarrow 1 : b_2$ **do**
    Generate N_n corrupted triples
    Optimize $\mathcal{L}_r$ by Eqn. 6.12
  Sample $\{(h_i, t_i), y_i\}_{i=1}^{b_3}$
  for $i \leftarrow 1 : b_3$ **do**
    Calculate $p(e_j|h_i)$ and $p(e_j|t_i)$
    Get contexts $\{a_h^m\}_{m=1}^{N_c}$ and $\{a_t^n\}_{n=1}^{N_c}$
    Optimize $\mathcal{L}_p$ by Eqn. 6.14
until Convergence

6.5 Experiments

In this section, we first introduce our experimental setup, e.g, the corpus-level co-occurrence statistics and datasets used for our experiments, and then compare our model with a list of comprehensive competitive baselines in terms of predictive performance. Moreover, we conduct expert evaluations as well as case studies to demonstrate the usefulness of our model rationales.

6.5.1 Dataset

We directly adopt a publicly available medical co-occurrence graph for our experiments (Finlayson et al., 2014). The graph was constructed in the following way: (Finlayson et al., 2014) first used an efficient annotation tool (LePendu et al., 2012) to extract medical terms from 20 million clinical notes collected by Stanford Hospitals and Clinics, and then computed the co-occurrence counts of two terms based on their appearances in one patient's records within a certain time frame (e.g., 1 day, 7 days). We experiment with their biggest dataset with the largest number of nodes (i.e., the per-bin 1-day graph here[27]) so as to have sufficient training data. The co-occurrence graph contains 52,804 nodes and 16,197,319 edges.

To obtain training labels for relation prediction, we utilize the mapping between medical terms and concepts provided by (Finlayson et al., 2014). To be specific, they mapped extracted terms to UMLS concepts with a high mapping accuracy by suppressing the least possible meanings of each term (see (Finlayson et al., 2014) for more details). We utilize such mappings to automatically collect relation labels from UMLS. For term e_a and e_b that

[27] https://datadryad.org/stash/dataset/doi:10.5061/dryad.jp917

are respectively mapped to medical concept c_A and c_B, we find the relation between c_A and c_B in UMLS, which will be used as the label for e_a and e_b.

Relations	UMLS Relations
May_treat	may_treat
May_prevent	may_prevent
Contraindicates	has_contraindicated_drug
Causes	cause_of; induces; causative_agent_of
Symptom of	disease_has_finding; disease_may_have_finding; has_associated_finding; has_manifestation; associated_condition_of; defining_characteristic_of

Table 6.1: Relations in our dataset and their mapped UMLS semantic relations. (UMLS relation "Treats" does not exist in our dataset and hence is not mapped with the "May_treat" relation.)

Following (Wang and Fan, 2014) that studied distant supervision in medical text and identified several crucial relations for clinical decision making, we select 5 important medical relations with no less than 1,000 relation triples in our dataset. Each relation is mapped to UMLS semantic relations, e.g., relation CAUSES corresponds to *cause_of*, *induces*, *causative_agent_of* in UMLS. A full list of mapping is in Table 6.1 We sample an equal number of negative pairs by randomly pairing head and tail entities with the correct argument types (Wang et al., 2016a). We split all samples into train/dev/test sets with a ratio of 70/15/15. Only relation triples in the training set are used to optimize relational parameters. The statistics of the positive samples for relations are summarized in Table 6.2.

Med Relations	Train	Dev	Test
Symptom of	14,326	3,001	3,087
May treat	12,924	2,664	2,735
Contraindicates	10,593	2,237	2,197
May prevent	2,113	440	460
Causes	1,389	305	354
Total	41.3k	8.6k	8.8k

Table 6.2: Dataset Statistics.

6.5.2 Implementation Details.

We implemented our model in Pytorch (Paszke et al., 2017) and optimized it by the Adam optimizer (Kingma and Ba, 2015). The dimension of term/node embeddings is set at 128. The number of negative triples for the relational learning is set at 100. The number of association contexts to use for assumption formation, N_c is 32. Early stopping is used when the performance in the dev set does not increase continuously for 10 epochs. We augment the relation triples for optimizing $\mathcal{L}_r$ (Eqn. 6.12) by adding their reverse relations for better training. We obtain DeepWalk and LINE (2nd) embeddings by OpenNE[28] and word2vec embeddings by doing SVD decomposition over the shifted PPMI co-occurrence matrix (Levy and Goldberg, 2014b). Code, dataset and more implementation details are available online[29].

[28] https://github.com/thunlp/OpenNE

[29] https://github.com/zhenwang9102/X-MedRELA

Methods	MAY_TREAT	CONTRAIN.	SYMPTOM_OF	MAY_PREVENT	CAUSES	Avg.
Word2vec + DistMult	0.767 (±0.008)	0.777 (±0.013)	0.815 (±0.005)	0.649 (±0.018)	0.671 (±0.015)	0.736
Word2vec + RESCAL	0.743 (±0.010)	0.767 (±0.003)	0.808 (±0.009)	0.658 (±0.023)	0.659 (±0.039)	0.727
Word2vec + NTN	0.693 (±0.013)	0.758 (±0.005)	0.808 (±0.004)	0.605 (±0.022)	0.631 (±0.017)	0.699
DeepWalk + DistMult	0.740 (±0.003)	0.776 (±0.004)	0.805 (±0.003)	0.608 (±0.014)	0.650 (±0.018)	0.716
DeepWalk + RESCAL	0.671 (±0.010)	0.778 (±0.003)	0.800 (±0.003)	0.600 (±0.023)	**0.708 (±0.011)**	0.711
DeepWalk + NTN	0.696 (±0.006)	0.778 (±0.005)	0.787 (±0.005)	0.614 (±0.016)	0.674 (±0.024)	0.710
LINE + DistMult	0.767 (±0.003)	0.783 (±0.002)	0.795 (±0.003)	0.621 (±0.015)	0.641 (±0.024)	0.721
LINE + RESCAL	0.725 (±0.003)	0.771 (±0.002)	0.801 (±0.001)	0.613 (±0.013)	0.694 (±0.015)	0.721
LINE + NTN	0.733 (±0.002)	0.773 (±0.003)	0.800 (±0.001)	0.601 (±0.015)	0.706 (±0.013)	0.723
REPEL-D + DistMult	0.784 (±0.002)	0.797 (±0.002)	0.809 (±0.003)	0.681 (±0.010)	0.694 (±0.022)	0.751
REPEL-D + RESCAL	0.726 (±0.003)	0.780 (±0.002)	0.776 (±0.002)	**0.685 (±0.010)**	**0.708 (±0.003)**	0.737
REPEL-D + NTN	0.736 (±0.004)	0.780 (±0.002)	0.773 (±0.001)	0.667 (±0.015)	0.694 (±0.024)	0.731
Ours (w/ CWA)	0.709 (±0.005)	0.751 (±0.009)	0.744 (±0.007)	0.667 (±0.008)	0.661 (±0.032)	0.706
Ours	**0.805 (±0.017)**	**0.811 (±0.006)**	**0.816 (±0.004)**	0.676 (±0.020)	0.684 (±0.017)	**0.758**

Table 6.3: Comparison of model predictive performance. We run all methods for five times and report the averaged F1 scores with standard deviations.

6.5.3 Predictive Performance Evaluation

Compared Methods. There are a number of advanced neural methods (Tang et al., 2015; Qu et al., 2018; Wang et al., 2018c) that have been developed for the link prediction task, i.e., predicting the relation between two nodes in a co-occurrence graph. At the high level, their frameworks comprise of an entity encoder and a relation scoring function. We adapt various existing methods for both the encoder and the scoring functions for comprehensive comparison. Specifically, given the co-occurrence graph, we employ existing distributional representation learning methods to learn entity embeddings. With the entity embeddings as input features, we adapt various models from the knowledge base completion literature as a binary relation classifier. More specifically, for the encoder, we select one word embedding method, Word2vec (Mikolov et al., 2013b; Levy and Goldberg, 2014b), two graph embedding methods, random-walk based DeepWalk (Perozzi et al., 2014), edge-sampling based LINE (Tang et al., 2015), and one distributional approach REPEL-D (Qu et al., 2018)

for weakly-supervised relation extraction that leverages both the co-occurrence graph and training relation triples to learn entity representations. For the scoring functions, we choose DistMult (Yang et al., 2015), RESCAL (Nickel et al., 2011) and NTN (Socher et al., 2013a).

Note that one can apply more complex encoders or scoring functions to obtain higher predictive performance; however, in this work, we emphasize more on model interpretability than predictive performance, and unfortunately, all such frameworks are hard to interpret as they provide little or no explanations on how predictions are made.

We also show the predictive performance of our framework under the CWA setting in which the CWA rationales are existing triples in a "closed" knowledge base (i.e., UMLS). We first adopt the pre-trained association recall module to retrieve associative contexts for head and tail entities, then formulate the assumptions using top-ranked triples (that exist in our relation training data), where the rank is based on the product of their retrieval probabilities ($p_{ij} = p(e_i|e_h) \times p(e_j|e_t)$). We keep the rest of our model the same as the OWA setting.

Results. We compare the predictive performance of different models in terms of F1 score under each relation prediction task. As shown in Table 6.3, our model obtains very competitive performance compared with a comprehensive list of baseline methods. Specifically, on the prediction tasks of MAY_TREAT and CONTRAINDICATES, our model achieves a substantial improvement (1$\sim$2 F1 score) and a very competitive performance on the task of SYMPTOM_OF and MAY_PREVENT. The small amount of training data might partly explain why our model does not perform so well in the CAUSES tasks. Such comparison shows the effectiveness of predicting relations based on associations and their relational interactions. Moreover, compared with those baseline models which encode graph structure into latent vector representation, our model utilizes co-occurrence graph more explicitly by

leveraging the associative contexts symbolically to generate human-understandable rationales, which can assist medical experts as we will see shortly. In addition, we observe that our model consistently outperforms the CWA setting: Despite the CWA rationales are true statements on their own, they tend to have a low coverage of possible rationales, and thus, may be not so relevant for the target relation prediction, which leads to a poor predictive performance.

6.5.4 Model Rationale Evaluation

To measure the quality of our model rationales (i.e., OWA rationales), as well as to conduct an ablation study of our model, we conduct an expert evaluation for the OWA rationales and also compare them with the CWA rationales. We first collaborate with a physician to explore how much a model's rationales help them better trust the model's prediction following recent work for evaluating model interpretability (Singh et al., 2019; Mullenbach et al., 2018; Atutxa et al., 2019; Jin et al., 2020). Then, we present some case studies to show what kind of rationales our model has learnt. Note that compared with evaluation by human annotators for open-domain tasks (without expertise requirement), evaluation by medical experts is more challenging in general. The physician in our study (an M.D. with 9 years of clinical experience and currently a fellow trained in clinical informatics), who is able to understand the context of terms and the basics of the compared algorithms and can dedicate time, is qualified for our evaluation.

Expert Evaluation. We first explained to the physician about the recall and recognition process in our framework and how model rationales are developed. They endorsed such reasoning process as one possible way to gain their trust in the model. Next, for each target pair for which our model correctly makes the prediction, they were shown the top-5

	OWA Rationales	CWA Rationales
Ranking Score	17	5
Avg. Sum Score/Case	6.14	2.24
Avg. Max Score/Case	2.04	0.77

Table 6.4: Human evaluation on the quality of rationales.

rationales produced by our framework and were asked whether each rationale helps them better trust the model prediction. For each rationale, they were asked to score it from 0 to 3 in which 0 is *no helpful*, 1 is *a little helpful*, 2 is *helpful* and 3 is *very helpful*. In addition to the individual rationale evaluation, we further compare the overall quality of CWA and OWA rationales, by letting experts rank them based the helpfulness of each set of rationales (the rationale set ranked higher gets 1 ranking score and both get 0 if they have the same rank). The details of the evaluation protocol can be found in Figure 6.4. We randomly select 30 cases in the MAY_TREAT relation and the overall evaluation results are summarized in Table 6.4. Out of 30, OWA wins in 17 cases and gets higher scores on individual rationales per case on average. There are 8 cases where the two sets of rationales are ranked the same[30] and 5 cases where CWA is better. To get a better idea of how the OWA model obtains more trust, we calculate the average sum score per case, which shows the OWA model gets a higher overall score per case. Considering in some cases only a few rationales are able to get non-zero scores, we also calculate the average max score per case, which shows that our OWA model generally provides one *helpful* rationale (score>2) per case. Overall, as we can see, the OWA rationales are more helpful to gain expert trust.

Case Study. Table 6.5 shows two concrete examples demonstrating what kind of model rationales our framework bases its predictions on. We highlight the rationales that receive

[30]Of which, 7 cases are indicated equally unhelpful.

Case 1		
cephalosporins	may_treat	bacterial infection
cefuroxime	may_treat	viral syndrome
cefuroxime	may_treat	low grade fever
cefuroxime	**may_treat**	**infectious diseases**
cefuroxime	may_prevent	low grade fever
sulbactam	may_treat	low grade fever
Case 2		
azelastine	may_treat	perennial allergic rhinitis
astepro	**may_treat**	**perennial allergic rhinitis**
pseudoephedrine	**may_treat**	**perennial allergic rhinitis**
ciclesonide	**may_treat**	**perennial allergic rhinitis**
overbite	may_treat	perennial allergic rhinitis
diclofenac	may_treat	perennial allergic rhinitis

Table 6.5: Case studies for rationalizing medical relation prediction. For each case, the first panel is target pair and the second is top-5 rationales (**Bold** ones are useful rationales with high scores from the physician). The left (right) most column is the head (tail) term and their relational associations.

high scores from the physician for being especially useful for trusting the prediction. As we can see, our framework is able to make correct predictions based on reasonable rationales. For instance, to predict that "cephalosporine" may treat "bacterial infection", our model relies on the rationale that "cefuroxime" may treat "infectious diseases". We also note that not all rationales are clinically established facts or even make sense, due to the unsupervised rationale learning and the probabilistic assumption formation process, which leaves space for future work to further improve the quality of rationales. Nevertheless, such model rationales can provide valuable information or new insights for clinicians. For another example, as pointed out by the physician, different medications possibly having the

same treatment response, as shown in Case 2, could be clinically useful. That is, if three medications are predicted to possibly treat the same condition and a physician is only aware of two doing so, one might get insights into trying the third one. To summarize, our model is able to provide reasonable rationales and help users understand how model predictions are made in general.

6.6 Discussion and Conclusion

In this chapter, we propose an interpretable framework to rationalize medical relation prediction based on corpus-level statistics. Our framework is inspired by existing cognitive theories on human memory recall and recognition, and can be easily understood by users as well as provide reasonable explanations to justify its prediction. Essentially, it leverages corpus-level statistics to recall associative contexts and recognizes their relational connections as model rationales. Compared with a comprehensive list of baseline models, our model obtains competitive predictive performances. Moreover, we demonstrate its interpretability via expert evaluation and case studies.

Our proposed model is a deep learning-based model built on top of symbolic knowledge, including two graphs, corpus-level statistics or term-term co-occurrence graph, and the external knowledge graph. The key intuitions of our model design include how to formulate the representation of basic knowledge units, i.e., knowledge tuples in our paper, and how to apply interpretable modeling to aggregate them for greater transparency, i.e., the attention module in our paper. Alternative approaches can explore both directions to generalize our model to broader applications. Specifically, we choose knowledge tuples as the atomic units of explainable knowledge for straightforward interpretations, but alternative approaches can formulate more meaningful knowledge units and representations, such

as a subgraph or a graph path, to represent higher-order and richer semantic information that may serve as better explanations later. Secondly, attention explanations introduce a certain degree of ambiguity since how to choose top-k knowledge as explanations is non-trivial. Thus, the alternative model can enforce hard attention or binary selections over the candidate knowledge units to make the explanations more decisive and useful.

Evaluation Interface (Example)

All models predict the **may_treat** relation between t1 term unfractionated heparin ['unfractionated heparin [epc]', 'heparin'] and t2 term myocardial infarction (mi) ['myocardial infarction'] with the following rationales.

Please answer the following questions:

1. Are you familiar with t1 and t2 terms?

   ○ Yes ○ No ○ Kind of

2. Check each rationale and answer this question: Is which degree is rationale helpful for you to trust the prediction?

   (0: no helpful; 1: a little bit helpful; 2: helpful; 3: very helpful)

Model A's Rationale Set:

T1's contexts	Relational Interaction	T2's contexts	Score
metabolic alkalosis	may_prevent	myocardial infarction (mi)	
metabolic alkalosis	may_prevent	venous thrombosis	
rbbb	may_treat	myocardial infarction (mi)	
ards	symptom_of	myocardial infarction (mi)	
micronutrient	may_prevent	venous thrombosis	

Model B's Rationale Set:

T1's contexts	Relational Interaction	T2's contexts	Score
cardiac dysrhythmias	contraindicates	theophylline	
malignant neoplasm without specification of site	has_symptom	family history of cancer	
Iddm	contraindicates	glyburide	
morphine sulfate	contraindicated_by	respiratory depression	
insulin dependent diabetes	contraindicates	glyburide	

3. Please rank all sets of rationales based on overall how much they help you trust the model prediction (e.g., A > B). Note that it is ok to reject them if both models are unhelpful (A = B = 0).

Figure 6.4: Evaluation interface for expert evaluation.

Chapter 7: Commonsense Knowledge Reasoning for Learning Word Representations

The previous chapter presents a reasoning framework for relation prediction inspired by human cognitive theory. This chapter presents a more generalized reasoning model, also working on structured knowledge and performing multi-hop reasoning on a commonsense knowledge graph. We apply this framework to learn better word representations by practicing this reasoning ability with a self-supervised task.

Specifically, distributional word representations have brought significant progress in numerous natural language understanding tasks. Traditional word embedding systems (e.g., word2vec, GloVe) learn word representations by optimizing their distribution in the vector space with two technical concerns. First, they are unable to utilize prior knowledge in word structures to guide representation learning with higher overfitting risks. Second, it is difficult for experts from various domains to understand and interpret the learned representations due to the statistical and analytical knowledge barrier. In this paper, we propose *CoRReL* (COmmonsense knowledge Reasoning based word REpresentation Learning) that leverages commonsense knowledge and reasoning to enhance word representation learning. *CoRReL* includes pre-training and testing phases. In the pre-training phase, we propose a self-supervision task that guides *CoRReL* to learn competitive reasoning modules. In the testing phase, *CoRReL* is able to provide word pair representations and

single word representations distilled from learned reasoning modules. Moreover, *CoRReL* offers reasoning paths to justify word closeness and correlation with minimal knowledge barrier. Empirical results on public benchmark datasets demonstrate the effectiveness and interpretability of *CoRReL*. The code for the reasoning part can be found on Github: https://github.com/zhenwang9102/CoRReL.

7.1 Introduction

Representing words in a vector space has become the standard to capture word semantics, and has led to tremendous success in natural language understanding tasks. Inspired by the idea that words appearing in the same contexts tends to be more related (Harris, 1954), existing learning systems (Mikolov et al., 2013a,b; Pennington et al., 2014; Devlin et al., 2019; Brown et al., 2020) optimize word representations so that correlation between words observed in same contexts can be preserved. For instance, in word2vec, one may expect "queen" is closer to "woman" than "man".

In this chapter, we investigate how to jointly leverage commonsense knowledge and knowledge reasoning to benefit word representation learning. First, learning from commonsense knowledge could effectively mitigate overfitting risks with better generalization performance. Commonsense knowledge bases (e.g., ConceptNet) directly provide prior knowledge (LoBue and Yates, 2011; Mihaylov and Frank, 2018; Guan et al., 2019; Banerjee et al., 2019) in word structures (e.g, the hierarchy between words (Miller, 1995)) such that one may not need to re-discover such structures (implicitly formed by learned word representations) from scratch and select good ones from a large model space by using millions of observed samples. Second, when one uses knowledge reasoning as the main problem solver, machine decisions can be justified by the underlying reasoning processes with

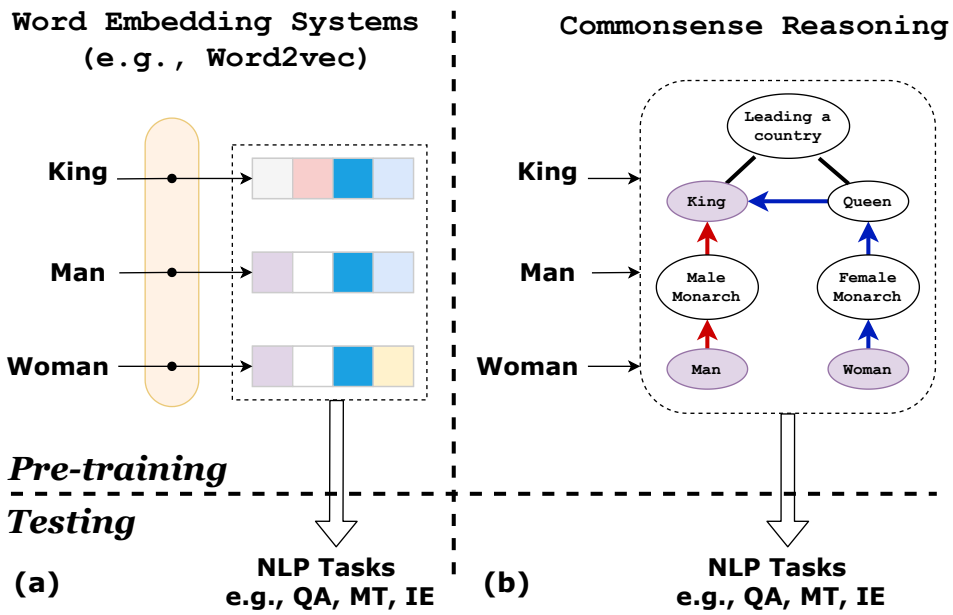

Figure 7.1: Two approaches for learning word representations. In contrast to traditional word embeddings (a) with shallow networks, our propose method (b) leverages commonsense knowledge reasoning with an explicit multi-hop reasoning process.

high transparency. Unlike existing methods (Ribeiro et al., 2016; Koh and Liang, 2017) that use statistical evidence in data or models to interpret decisions, reasoning based interpretations could be understood and accepted by domain experts (e.g., lawyer, salesperson, physicians, and so on) with minimized knowledge requirement in statistics or analytics. As shown in Figure 7.1, instead of explaining in a latent space the distance between "king" and "man" is smaller than the distance between "king" and "woman", reasoning paths from "man" to "king" and from "woman" to "king" could be easily recognized by ordinary users with little knowledge barrier.

Learning word representations from commonsense knowledge and knowledge reasoning is also challenging. First, although machines have access to facts stored in commonsense knowledge bases, their reasoning skills are missing. It is intuitive that we need self-supervision tasks that guide machines to master reasoning skills with the potential to benefit

downstream tasks, but it is unknown which tasks would be practical and cost-effective. Second, it is technically non-trivial to bridge the gap between knowledge reasoning and single word representations. Even if machines have mastered competitive reasoning skills, such skills generally deliver word kernels where knowledge reasoning captures interactions of word pairs. To support existing downstream modules that rely on word representations, it is desired to distill word representations from learned knowledge reasoning skills.

Existing techniques are generally unable to address the aforementioned challenges. There have been attempts to inject external knowledge into neural word representations by defining knowledge-award objective functions (Faruqui et al., 2014; Liu et al., 2016). A few recent works propose to learn representations from knowledge graphs (Bordes et al., 2013; Speer et al., 2017) without transparent justification provided by reasoning.

In this chapter, we propose a general framework *CoRReL* (*COmmonsense knowledge Reasoning based word REpresentation Learning*) that jointly utilize commonsense knowledge and knowledge reasoning to enhance generalization and interpretation of word representations. As shown in Figure 7.1, *CoRReL* includes pre-training and testing phases. In the pre-training phase, under a self-supervision task, a reasoning module in *CoRReL* is trained to discover reasoning paths between words that are correlated with co-occurrence statistics. In the testing phase, the learned reasoning module can directly serve downstream modules that rely on word pair representations; meanwhile, for the downstream modules that rely on single word representations, *CoRReL* provides word representations distilled from the learned reasoning module.

In particular, *CoRReL* addresses the aforementioned technical challenges as follows.

C1: How to guide machines to practice reasoning skills? We consider a reasoning procedure between a pair of words as a sequence of parameterized decision-making processes: Given the reasoning history, which word in the knowledge graph the reasoning procedure should proceed to next. In addition, we develop a self-supervision task where well-trained decision-making processes are expected to generate reasoning paths that are correlated with observed co-occurrence statistics. In this work, the parameterized decision making processes are implemented by deep neural networks.

C2: How to bridge the gap between reasoning and word representations? A well-trained reasoning module in *CoRReL* basically defines how likely a reasoning procedure would jump from one word to another. To this end, meaningful word representations from *CoRReL* should be able to preserve such transition probability. In this work, we address this problem from the perspective of knowledge distillation where word representations distilled from *CoRReL* preserve the transition knowledge in the learned reasoning module.

Note that our research goal in this chapter is to study the value and impact from commonsense knowledge and knowledge reasoning in word representation learning, instead of delivering state-of-the-art solutions to specific tasks. Therefore, in the self-supervision task design, we consider basic global co-occurrence statistics (against word2vec and GloVe), but do not include syntactic information used in the latest approaches (e.g., BERT). Due to the non-trivial technical challenge, the discussion on how to make *CoRReL* preserve syntactic information is out of this chapter's scope.

We empirically evaluate the effectiveness of *CoRReL* on a public benchmark dataset for CommonsenseQA (Talmor et al., 2018). Compared with competitive baseline methods, word pair representations delivered by *CoRReL* achieve up to 4% improvement in accuracy. For single word representations, by combining the word representations distilled

from *CoRReL*, a modified BERT is able to obtain around 1% significantly improvement compared with the original one, even with a few labels. In addition, a case study is provided to demonstrate reasoning paths learned by *CoRReL* and their potential to justify word co-occurrence.

7.2 Related Work

Pretrained Word Representations. Aside from traditional distributional word embeddings (Collobert et al., 2011; Mikolov et al., 2013a,b; Pennington et al., 2014), there are also some variants that try to inject external knowledge into the vector space (Faruqui et al., 2014; Bollegala et al., 2015; Liu et al., 2016; Lengerich et al., 2017; Speer et al., 2017), where most of them consider lexicon knowledge and define knowledge-aware objective functions constrained by the knowledge structure. However, little attempt from them was made to incorporate the explicit multi-hop reasoning process into the representation learning. More recently, pre-trained language models (PLMs) have been revolutionized the NLP field (Peters et al., 2018; Devlin et al., 2019; Radford et al., 2018), and one active line of research is to combine world knowledge, mainly knowledge graphs with PLMs (Sun et al., 2019b; Peters et al., 2019; Logan IV et al., 2019; Liu et al., 2020; Yu et al., 2020). Nonetheless, PLMs model both word semantics and sentence syntax, whereas traditional embeddings and our framework only consider semantics. Thus, we will not compete our framework with PLMs directly, but show the effectiveness of combining our representations with PLMs in experiments later.

Commonsense Knowledge and Reasoning. Recently, there has been renewed interest in teaching machines human-like commonsense knowledge and reasoning from NLP community. There are several fundamental open research problems, including how to represent,

probe, benchmark, incorporate commonsense knowledge. More relevant to our focus, lots of efforts have been made to construct high-quality and large-scale commonsense knowledge graphs, (Speer et al., 2017; Sap et al., 2019; Bosselut et al., 2019), which we are able to leverage to regularize our representation learning. Moreover, there is a growing body of literature that tries to integrate commonsense knowledge into NLP tasks by using structured knowledge bases (Lin et al., 2019) or PLMs (Banerjee et al., 2019). By contrast, in this chapter, we incorporate commonsense reasoning into representations which can also be applied to previous integration work.

Graph and Knowledge Reasoning. Extensive research has been conducted to do reasoning over knowledge graphs or general graphs. Based on the number of reasoning steps, there are generally two types of models for knowledge graph reasoning, one-hop and multi-hop reasoning. The former one mainly focuses on learning knowledge graph embeddings by various scoring functions (Bordes et al., 2013; Wang et al., 2018c; Yang et al., 2015), while the latter one aims to explicitly model paths for reasoning with path-sampling methods (Guu et al., 2015; Neelakantan et al., 2015; Toutanova et al., 2016) and Reinforcement Learning models (Xiong et al., 2017; Das et al., 2017a; Chen et al., 2018; Shen et al., 2018; Lin et al., 2018). Our framework also encourages multi-hop reasoning process, but these methods suffer from either scalability issue for sampling, or sparse rewards.

Interpretability on NLP Models. Interpretable Machine Learning has drawn extensive attention in recent years, especially in NLP field (Murdoch et al., 2019). There are two general approaches to improve the interpretability of NLP models (Belinkov et al., 2020; Wallace et al., 2020), the first is to conduct post-hoc analysis for a well-trained model, while the second is to intrinsically build interpretable models that typically vary case by case. The interpretability research usually has more focus in the first one for its generalizability, for

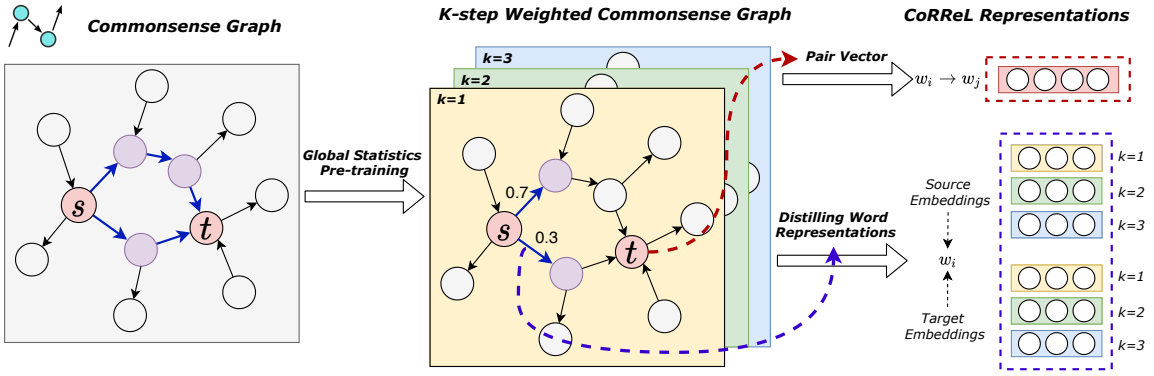

Figure 7.2: *CoRReL* Framework Overview.

example, by probing internal representations and model behaviors (Conneau et al., 2018; Tenney et al., 2019), checking input features by saliency maps or perturbations (Ribeiro et al., 2016; Li et al., 2016), finding decision rules or influential training examples (Guidotti et al., 2018; Han et al., 2020), etc. The goal of our framework is more towards to the second category which can help build an intrinsic interpretable model.

7.3 Proposed Method

In this section, we introduce *CoRReL* framework for learning neural commonsense representations. We first give an overview by formulating the problem and defining mathematical notations. Then, we introduce the self-supervision task used to pre-train the reasoning module, and finally, we illustrate how to distill word representations from *CoRReL*. The framework is illustrated in Figure 7.2.

7.3.1 Problem Formulation and Overview

In this chapter, we aim to learn a general representation for each word in a commonsense knowledge graph, which can be used to interpret the relationships between words

based on the knowledge reasoning process. . To interpret the fine-grained relationship between two nodes, we need a human-understandable structure for each pair, which we define as the paths in between. For example, we may find that "oven" and "survive" are related because of the path ("oven", *AtLocation*$^{-1}$, "cake"), ("cake" , *UsedFor*, "eat"), ("eat", *Causes*$^{-1}$, "survive"), where $^{-1}$ represents the reversed relation.

One straightforward way to obtain such paths from the commonsense graph could be to find all simple paths between nodes. However, running shortest path algorithms such as Dijkstra's algorithm for all pairs in the graph could cause severe scalability issue. Also, different path-funding algorithms induce various bias for paths to some degree. Thus, as the first technical contribution of this chapter, we propose a parametrized graph neural network model to select paths.

Formally, we denote the commonsense knowledge graph as $\mathcal{G} = (\mathcal{V}, \mathcal{R}, \mathcal{E})$, in which $\mathcal{V}$ is the set of nodes[31] and $\mathcal{R}$ is the set of commonsense relations, e.g., *Synonym* and *Antonym*. $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{R} \times \mathcal{V}$ is the set of edges representing the head-relation-tail triples. Our reasoning network essentially learns how to navigate the reasoning process from a source word s to a target word t within K hops and represent such multi-hop reasoning process as a vector. We denote such reasoning vector as $\mathbf{v}_{s \to t} = f(s, t|G, \theta)$.

7.3.2 Pre-training Commonsense Knowledge

To inject the commonsense knowledge into the reasoning process, we propose to pre-train the parameters by global statistics (Pennington et al., 2014).

Let the global co-occurrence matrix be denoted by X in which X_{ij} represents the number of times word j occurs in the context of word i and $X_i = \sum_k X_{ik}$ is the total number

[31] A node is a word or phrase in the natural language, which usually is a common word in its unambiguous form (Speer et al., 2017). We use "nodes" and "words" interchangeably in this chapter.

of times any words appears in the context of word i. Also, we denote the probability that word j appears in the context of word i as $P_{i,j} = p(j|i) = X_{i,j}/X_i$.

We train our reasoning network to fit the global statistics by applying a simple linear transformation on the reasoning vectors by defining the conditional probability of reaching j from i as

$$Q_{i,j} = \hat{p}(j|i) = \frac{\exp(\mathbf{w}^T \mathbf{v}_{i \to j})}{\sum_{k \in V} \exp(\mathbf{w}^T \mathbf{v}_{i \to k})} \tag{7.1}$$

To minimize the distance between the estimated probability $\hat{p}(j|i)$ and empirical probability $p(j|i)$, there are several possible distance measures, such as cross entropy. Due to the computational bottleneck of normalization operation for cross entropy loss, we could simply choose a least square loss between two unnormalized distributions, $\widetilde{P}_{i,j} = X_{i,j}$ and $\widetilde{Q}_{i,j} = \exp(\mathbf{w}^T \mathbf{v}_{i \to j})$. For numerical stability, we take their logarithms and our objective can be defined as:

$$\begin{aligned} \mathcal{L}_1 &= \sum_{i,j} g(X_i) \left(\log \widetilde{P}_{i,j} - \log \widetilde{Q}_{i,j} \right)^2 \\ &= \sum_{i,j} g\left(X_i\right) \left(\mathbf{w}^T \mathbf{v}_{i \to j} - \log X_{ij} \right)^2 \end{aligned} \tag{7.2}$$

where $g(\cdot)$ is a weighting function to discount the influence of frequent words (Mikolov et al., 2013b; Pennington et al., 2014).

7.3.3 Commonsense Reasoning Network

In this section, we describe details of our reasoning network that is pre-trained to learn commonsense knowledge by reasoning over the graph. For a given pair of nodes in the graph s, t, we leverage message passing mechanism (Gilmer et al., 2017) to propagate messages from the source node and aggregate them at the target node.

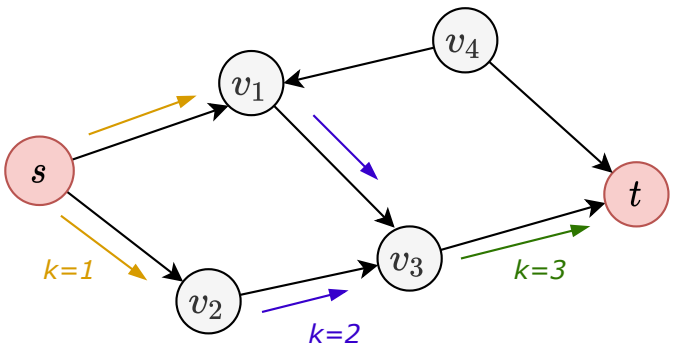

Figure 7.3: Illustrating the Multi-hop Reasoning Process.

For any intermediate nodes u between s and t, we calculate the conditional probability of transiting from u to its outgoing neighbors $v \in \mathcal{N}_u^{\text{out}}$ at k hop as follows:

$$p^{(k)}(v|u) = \frac{\exp\left(\psi\left(u, v, r_{u,v}\right)\right)}{\sum_{a \in \mathcal{N}_u^{\text{out}}} \exp\left(\psi\left(u, a, r_{u,a}\right)\right)} \tag{7.3}$$

where $\psi(u, v, r)$ is a MLP function that encodes the edge feature. Then, we update the hidden state of u at $k+1$ hop by neighborhood aggregation mechanism (Hamilton et al., 2017) as follows:

$$\mathbf{h}_u^{(k+1)} = \sum_{(r,b) \in \mathcal{N}_u^{\text{in}}} p^{(k)}(u|b) \cdot m_{s,b}^{(k)} \cdot \phi(u, b, r) + \mathbf{h}_u^{(k)} \tag{7.4}$$

where $\phi(u, b, r)$ is another MLP function to encode the edge. $m_{s,b}^{(k)}$ is a masking vector to ensure the messages are all from the source node s. $\mathbf{h}_u^{(k)}$ is the hidden state of u from the last hop. Lastly, for a maximum number of hops K, we take the last hidden state on the target node t to represent the whole reasoning process as well as the pair of nodes as $\mathbf{v}_{s,t} = \mathbf{h}_t^{(K)}$.

An illustration of a reasoning process is shown in Figure 7.3 with a 3-hop reasoning process in a toy graph. Starting from the source node s, the iteratively propagates source

messages to next available hop until we arrive at the target node. Essentially, the last hidden state of t has incorporated several paths, i.e., $s \rightarrow v_1 \rightarrow v_3 \rightarrow t$ and $s \rightarrow v_2 \rightarrow v_3 \rightarrow t$ in the given graph from Figure 7.3. Note that v_4 is not connected with s within K hops, thus, the target node does not receive its messages. This function is achieved by the masking vector $m_{s,\cdot}^{(k)}$ in Eqn. (4).

7.3.4 Distilling Word Representations.

Now that we have described how we can pre-train the reasoning network, we further introduce how we can derive the representation for each word.

As we have described in previous part, at each hop, for an intermediate word u between s and t, we have a transition probability, $p(v|u), v \in \mathcal{N}_u^{\text{out}}$. In other words, at each hop, the reasoning network provides a transition matrix, M^k, in which $M^k(u, v) = p(v|u)$. In order to learn meaningful word representations, we propose to reconstruct the reasoning process, that is, the transition matrix or a re-weighted graph.

From the perspective of matrix factorization, we can decompose the transition matrix at k hop as $M_k = U_k \cdot \Sigma_k \cdot V_k^T$, in which U_k (V_k) represent vectors for words when they are the source (target) word. However, it would be computationally prohibitive to explicitly calculate the exact value of transition matrix[32]. Thus, we approximate the decomposition by defining tractable functions for U_k and V_k. To be specific, for a given set of edges sampled from the re-weighted graph, $E_k = \{e_0, e_1, ..., e_m\}, e_i = e(v_{src_i}, v_{tgt_i}) \in V$, we have non-negative edge weights, $w(e_i)$ obtained from the reasoning network. We then define two neural networks for U_k and V_k to reconstruct the edge weights as

$$\hat{w}(e_i) = f_s(\mathbf{v}_{src_i}) \cdot W_\Sigma \cdot f_t(\mathbf{v}_{tgt_i})^T \tag{7.5}$$

[32]The complexity of explicitly calculating the transition matrix is $\mathcal{O}(|V| \cdot |E|)$.

where f_s and f_t are projection function to produce an embedding for source and target words at each hop. $\mathbf{v_{src_i}}$ and $\mathbf{v}_{tgt_i}$ are embedding parameters and W_Σ is a trainable parameter. We train the above parameterized model by a straightforward mean-square error (MSE) loss:

$$\mathcal{L}_2 = \frac{1}{N}\sum_{i=1}^{N}(w_i - \hat{w}_i)^2 \tag{7.6}$$

where N is the number of edges to sample.

Note that there are two conditions we need to consider, for $k = 1$ and $k > 1$. The difference is that when $k = 1$, edges' attention weights only depend on a single source word which is unambiguous to calculate, while when $k > 1$, these weights depend on multiple source words. For the second case, we will represent the weights by taking their mean and variance of all possibilities.

Finally, we could obtain the word representations by concatenating all source/target embeddings from each hop together.

7.4 Experiments

In the experiment section, we first introduce our choices of commonsense knowledge graph and benchmark dataset for evaluation. Then, we present the compared methods that adopt and evaluate various representations. We finally analyze the results and show the interpretability of our representations.

7.4.1 Dataset

To transfer commonsense knowledge from the symbolic world into neural representations, the choice of the source of commonsense knowledge is not trivial. Luckily, several

high-quality CSKGs have been constructed in recent years. Some popular choices include CONCEPTNET (Speer et al., 2017), Atomic (Sap et al., 2019), COMET (Bosselut et al., 2019), OpenCyc (Lenat, 1995), Webchild (Tandon et al., 2017), etc. Among them, we select CONCEPTNET[33] as the backbone of our neural representation learning mainly for two reasons: (1) CONCEPTNET is one of the largest and the most general CSKG that has good coverage of general commonsense knowledge. (2) CONCEPTNET represents the semantic knowledge in natural language form whose vocabulary size is very closed to the commonly-used vocabulary in NLP, such as GloVe.

After selecting the source of commonsense knowledge, we need a benchmark to evaluate the quality of our learned representations. There are many language-related commonsense benchmarks proposed recently ranging from Visual Commonsense Reasoning (Zellers et al., 2019) to Machine Reading Comprehension (Zhang et al., 2018b). To show the effectiveness of our representations, we select the CommonsenseQA dataset (Talmor et al., 2018), which contains 12,102 multi-choice questions collected by human annotators and has a wide range of coverage over commonsense, such as social, physical, spatial, causal and temporal, etc. We leave evaluating our representations on more general tasks, e.g., natural language inference, test classification, as future work.

7.4.2 Experiment Setup

We adopt CONCEPTNET 5.6.0 with 21 million edges and over 8 million nodes covering 85 languages. We extract the English triples covering over 2.4 million edges and 0.8 million nodes. However, most of the nodes in CONCEPTNET are directly extracted from a crowdsourced text corpus, Open Mind Commonsense, which contains some noise and thus, are less likely to be used in normal text. Therefore, we define a set of heuristic rules to filter

[33] http://conceptnet.io/

	10%		30%		50%		70%		100%	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
Random Guess	0.200	0.200	0.200	0.200	0.200	0.200	0.200	0.200	0.200	0.200
Bert-Base	0.426 ±0.016	0.395 ±0.006	0.471 ±0.015	0.455 ±0.006	0.490 ±0.011	0.463 ±0.005	0.511 ±0.006	0.484 ±0.009	0.527 ±0.014	0.492 ±0.008
Bert+*CoRReL*	**0.434** ±0.010	**0.411** ±0.005	**0.479** ±0.005	0.444 ±0.009	**0.501** ±0.000	**0.468** ±0.008	0.510 ±0.006	**0.492** ±0.005	**0.538** ±0.004	**0.503** ±0.009

Table 7.1: Performance comparison by combining our word representations with Bert model. We report the testing accuracy under different ratio of training data with the standard deviations with at least 3 times running.

undesired nodes, e.g., abandon nodes containing more than four words, ensure the nodes are alphabetic, etc. We also follow a commonly-used graph pruning procedure (Speer et al., 2017) by discarding nodes with fewer than 10 edges. To ensure the connectivity of the graph for multi-hop reasoning, we select the largest connected component. Finally, we have a clean graph with 813,394 edges and 59,302 nodes. For relations, we select and merge 17 common relations by following previous work (Lin et al., 2019) and also add their reversed relations.

For the benchmark dataset, CommonsenseQA, we adopt the in-house setting with the splitting of IHtrain/IHdev/IHtest as 8,500/1,221/1,241 for efficiently testing re-implemented baselines and our framework. We use the random-slit setting in CommonsenseQA which is considered as the hard mode because questions with the same concept could appear in both training and testing, which could confuse the model.

7.4.3 Compared Methods

CoRReL produces two kinds of embeddings, word pair representation and single word representation, and evaluating them in the exact same setting is not straightforward. For instance, pair representations suit tasks that have two inputs, such as question answering,

Model	Commonsense QA	
	Dev Accuracy	Test Accuracy
Random Guess	0.20000	0.20000
QABilinear + Glove	0.29044	0.27273
QABilinear + NB	0.29359	0.26133
QACompare + Glove	0.30206	0.27807
QACompare + NB	0.33452	0.30489
CoRReL + Glove	0.26363	0.23886
CoRReL + NB	0.29448	0.24622
CoRReL	0.36744	0.33956

Table 7.2: Performance comparison with our pair representations with other word representations.

to form word pairs, which requires a specific model to accommodate such kinds of inputs. On the other hand, single word representations are more generalizable that can be merged into more advanced base models, such as Bert. Thus, we will evaluate our pair and word representations in two different baseline settings.

Baselines for Pair Representations. We adopt two supervised learning models from the original paper of CommonsenseQA (Talmor et al., 2018), QABilinear and QACompare, which can equipped with different embeddings. For baselines of embeddings, we compare with **GloVe** (Pennington et al., 2014), a general word embeddings also trained by global statistics, and **Numberbatch**[34] (Speer et al., 2017), an assembled word embeddings combining general embeddings, such as word2vec, GloVe and retrofitting on CONCEPTNET..

[34] https://github.com/commonsense/conceptnet-numberbatch

To apply our pair representations to the multi-choice QA problem of CommonsenseQA dataset, we build a simple classification model that takes as the input the pair representations with a self-attention layer to weigh each pair representations. Specifically, each QA pair is represented as $h = \sum_{ij} \alpha_{ij} \mathbf{v}_{i \to j}, \alpha_{ij} = f(\mathbf{v}_{i \to j}), i \in q, j \in a$, in which $f(\cdot)$ s a MLP function, i and j are the concepts extracted from the question and answer, respectively. Then the model predicts an answer score using a linear layer as $h \cdot W_1 + b_1$. We also adjust the previous two embeddings by concatenating words from the question and answer to form pairs as the input for QAPair model.

Baselines for Word Representations. Because of the generalizability of word representations that can be combined with more advanced base models, we are interested in the performance by combining them straightforwardly with the powerful Bert model. Intuitively, we can replace the word embeddings of the Bert model with our word representations, but such adoption requires to re-train the pre-trained Bert model, which is computationally and financially costly. Thus, we integrate word representations into Bert model by simply concatenating them with [CLS] vectors with one additional linear layer for the final classification. The rest of the settings are the same as the Bert base model.

7.4.4 Implementation Details

We implement our framework with Pytorch with Adam optimizer. We implement a sparse graph neural network to deal with the large-scale graph. The maximum number of hops for the reasoning, K is set to 3. The dimension of hidden states is set to 100. We pre-train the graph model by the global statistics by at least 5 epochs. Each epoch takes about 3 hours on one NVIDIA Quadro RTX 6000 GPU. For word representation distillation, the dimension for each source and target embeddings is set to be 256 and we sample 10,000

nodes to approximate the reasoning module for the word reconstruction. We adopt the large-scale Wikipedia corpus[35] to calculate the global statistics. A lemma-based concept linker is adopted to map concepts from the text to CSKG (Lin et al., 2018).

7.4.5 Experiment Results

Evaluating Pair Representations. As the results are shown in Table 7.2, our *CoRReL* model outperforms the baseline methods by a large margin. Firstly, we find that Numberbatch embeddings generally perform better than GloVe, because they are also pre-trained in CONCEPTNET structure. Even though, compared with the best baseline combination, QACompare+NB, our model improves almost 4% improvement in testing accuracy. More importantly, with the same classification model, our model improves the testing accuracy by 15% compared with Numberbatch embeddings. These results indicate that the multi-reasoning process for word pairs encoded as vectors can provide useful information for downstream tasks, which could further promote the research on studying the pairwise vectors.

Evaluating Word Representations. The results of combining our word representations with the Bert model are shown in Table 7.1. Bert model has been considered as a powerful text encoder to model the semantic and syntax information and our results show that our word representations can further improve it with a straightforward integration, e.g., by simply adding the averaged word representations with the [CLS] vectors of Bert model. Moreover, we find the improvement in the low-resource areas is relatively stable, which partially justifies that our reasoning-based vectors could improve the generalizability of the Bert model.

[35]Downloaded here: https://hotpotqa.github.io/wiki-readme.html

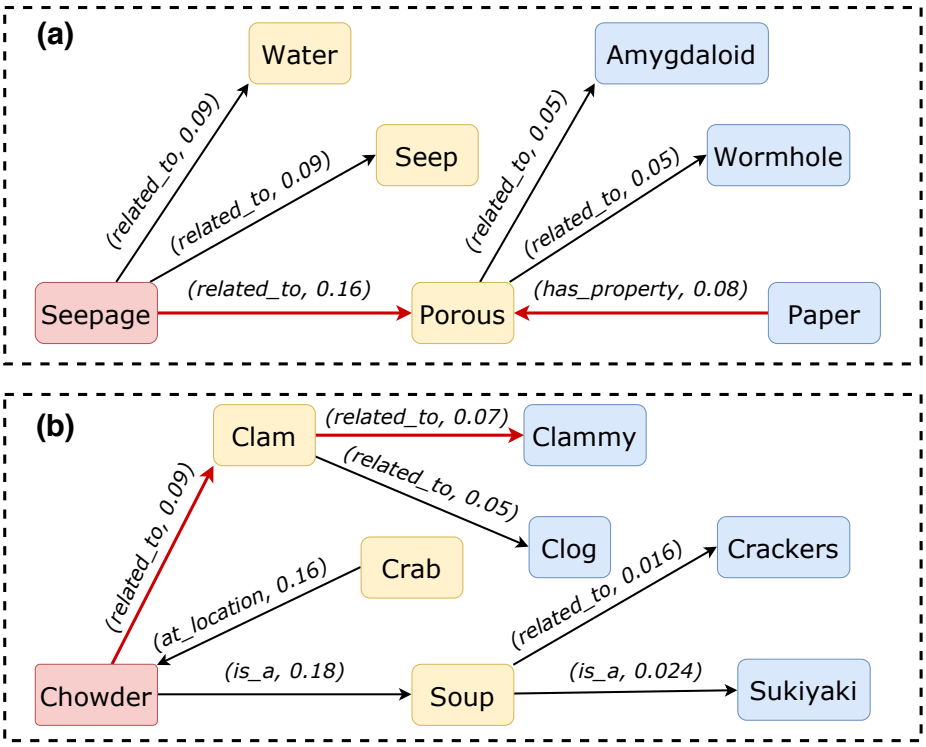

Figure 7.4: Visualization of the decoded multi-hop reasoning process.

7.4.6 Interpretability Results

One major contribution that differentiates *CoRReL* from other knowledge-enhanced representation learning is that it can provide reasoning paths as the justification for word closeness and correlation with minimal knowledge barrier for general machine learning users. To illustrate the pre-trained reasoning paths, we randomly sample the source nodes and then adopt beam search algorithm to search for the most likely paths for the source nodes. We keep the size of beam search as 5.

The results of the decoded paths are shown in Figure 7.4 and 7.5, which offer two views of our reasoning process. Figure 7.4 provides a relatively more global view of how the reasoning works, in which we provide multiple neighbors with high transition probability. We

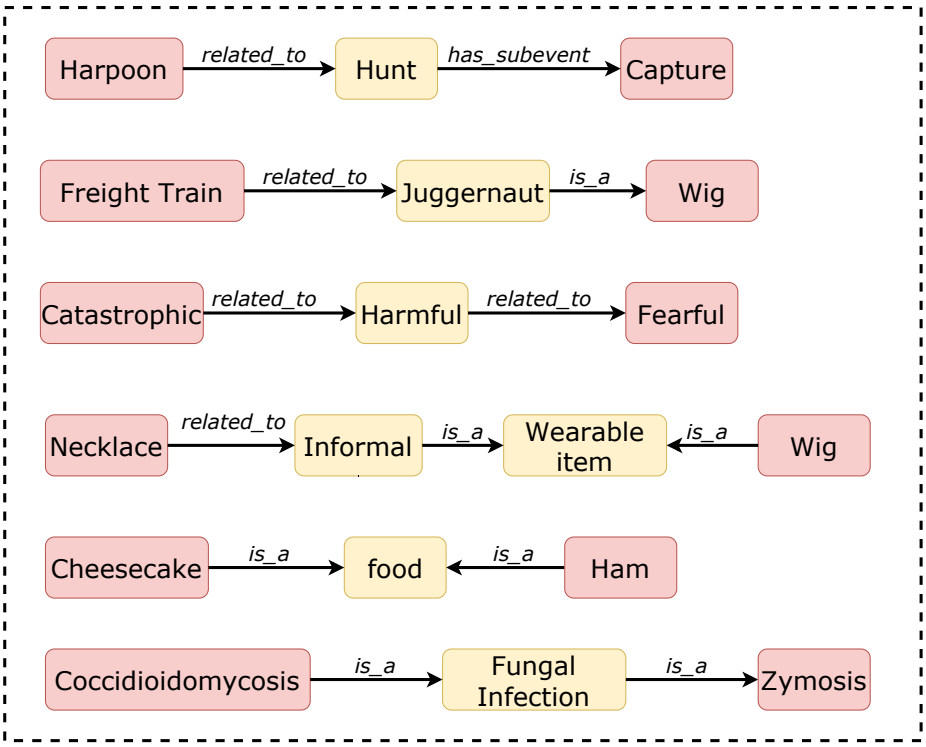

Figure 7.5: Examples of top-ranked paths decoded by beam search algorithm.

limited the number of decoding to 2 to save space and red paths represent the top-ranked ones in 2-hops. We also ignore the neighbors with too small probabilities to save space. The tuple above edges indicates the relation name and transition probability. On the other hand, Figure 7.5 simply presents the top-ranked paths to show more concrete examples. By observing Figure 7.4 (a) and (b), we can find that the pre-trained reasoning module has learned a preference when making a local decision, i.e., selecting which neighbor to go for the next hop. Such characteristics of transition distributions indicate that *CoRReL* has been trained successfully to perform reasoning based on commonsense knowledge. More examples in Figre 7.5 demonstrate that with reasoning path information, we can better understand the relationships between each pair. For example, we may wonder how "harpoon" and "capture" are connected and it turns out they can be connected by the word "hunt"

with two relations. In conclusion, reasoning paths express the commonsense relationships between pairs, and models that are equipped with our representations could leverage such commonsense knowledge to make a better prediction.

7.5 Discussion and Conclusion

In this chapter, we propose a new framework, *CoRReL* for learning interpretable neural word representations by commonsense knowledge and reasoning. We define a self-supervised task to jointly pre-train the commonsense reasoning module by structure knowledge and the corpus global statistics. The commonsense knowledge is pre-trained to be consolidated into the reasoning module. We further develop representations for single words by knowledge distillation. More importantly, with such a symbolic reasoning process, *CoRReL* can further provide reasoning paths to explain relationships between word pairs. We show the advancement of *CoRReL* in a popular NLP benchmark compared with two strong traditional embedding systems and the effectiveness of combining *CoRReL* with the BERT model. We also demonstrate interpretability by visualizing the reasoning paths.

CoRReL consists of two stages, pre-training on commonsense knowledge graph and adaptation on commonsense NLP tasks. The former is essentially a path-based graph neural network where we encode the path distribution between any two nodes in the graph with attention aggregation in between. This approach requires the calculation of the higher-order matrix of the adjacent matrix of the graph and scaling up this approach to a longer length, e.g., > 3, needs significant computation. Alternative approaches can explore efficient computation of adjacent matrix (Wang et al., 2021b). Also, the adaptation stage merges pre-stored reasoning knowledge with downstream NLP tasks in a very straightforward way and alternative methods should explore a deeper fusion of them and inject

more transparency into downstream modeling. Moreover, future works should explore an end-to-end system to combine the pre-training and downstream adaptation together. This requires backpropagating gradients from downstream tasks to the graph neural networks, where reparameterization tricks or reinforcement learning approaches can be leveraged.

Part VI: Conclusion and Future Work

Chapter 8: Conclusion and Future Work

The previous chapters introduce our work on advancing the knowledge-centric NLP by promoting a life cycle of incorporating human knowledge into existing AI systems, i.e., knowledge acquisition, representation, transfer, and reasoning. To the best of our knowledge, this dissertation is the first to introduce a life cycle systematically for addressing the knowledge gap between statistical learning models and human-like learning processes. The first two steps, acquisition and representation, mimic the process of human knowledge acquisition and organization by discovering structured knowledge from noisy data corpus and learning their neural representation. The last two steps are more advanced operations, knowledge transfer, and reasoning. The former takes an analogy of how humans generalize existing knowledge to new tasks, and the latter simulates human-like complex reasoning for a more transparent decision-making process of AI systems. This chapter will summarize our key contributions and then outline their limitations to motivate promising future directions.

8.1 Summary of Key Contributions

We first give an overview of our high-level contributions in this dissertation as follows.

- We demonstrate that structured and unstructured knowledge are highly complementary and beneficial to each other. Structured one can provide weak supervisions and

rich global structural information (Chapter 2), as well as the backbone for interpretable modeling (Chapter 6 and Chapter 7). When representing both knowledges, more attention must be paid to modeling the complex interactions among structured neighborhoods (Chapter 3).

- Reasoning and transferring are advanced knowledge operations for significantly boosting AI systems' interpretability and generalizability, which are also the keystones for the success of injecting human-like learning into AI systems. Our works (Chapter 4 and Chapter 5) highlight that the unified transfer framework (Chapter 4) and advanced knowledge compression via multi-task learning (Chapter 5) are the keys for knowledge transfer. We also propose novel data-driven approaches to derive the intrinsic reasoning process automatically guided by task-specific (Chapter 6) and self-supervised (Chapter 7) learning signals.

We now summarize the specific contributions of each chapter by discussing the state of the literature and the fundamental changes our works introduced to the field.

Part II, Knowledge Acquisition, aims to accumulate high-quality human knowledge automatically from the real but noisy world, such as unstructured text corpus. This falls into the field of automatic knowledge base construction (AKBC) or text mining, where structured knowledge, such as relations and entities, is extracted from the text corpus. We summarize our major contributions as follows:

- Chapter 2 brings a novel setting to the research field, where the raw text sequences are unavailable due to practical concerns, such as privacy or security. We thus propose to study the privacy-aware data, i.e., terms co-occurrence counts, without raw texts to discover one of the most important structured knowledge, synonyms. Existing

works on knowledge extraction focus on text input but ignore global contexts from the privacy-aware data. At the same time, our proposed method leverages two important pieces of information, surface form and global context information. Our research highlights the complementary relationship between both information and proposes a novel module to inductively predict relevant contexts to deal with out-of-vocabulary queries (i.e., queries not in the privacy-aware data).

Part III, Knowledge Representation, connects to the field of representation learning, such as learning embeddings for words, sentences, or graphs. After extracting structured knowledge from the previous stage, we focus on learning representation for structured knowledge in this dissertation, such as graphs, and summarize our major contributions as follows:

- Chapter 3 highlights the importance of explicit modeling the interactions of local contexts around nodes and proposes a new formulation that enables us to build more appropriate representations for node pairs. We propose two variants of modeling context pair interactions, node-centric and pair-centric, where the former formulates node representations for pairwise predictions, while the latter directly models pair representations. Motivated by the pair representation, we propose a new graph embedding to be pre-trained to capture the pair semantics of any node pairs and to be plugged into the pair-centric interaction modeling for superior performance. Most graph embedding works focus on learning node embedding, and our novelty comes from learning node pair embeddings to learn node pair semantics.

Part IV, Knowledge Transfer, involves the broad area of transfer learning to take advantage of existing knowledge and generalize it to new tasks. We study how to transfer

knowledge across neural systems under the paradigm of pre-training and finetuning, i.e., the pre-training stores task-specific knowledge into neural parameters, and the following finetuning will adapt such knowledge to new tasks. We summarize our major contributions as follows:

- Chapter 4 studies knowledge transfer between KBQA and TextQA systems to answer research questions like what knowledge is transferred or whether the transferred knowledge can help answer over one source using another one, which is yet to be answered. This is the first work to unify KBQA and TextQA systems on the CQA task. Our key contribution is to reformulate KBQA and TextQA as a retrieval-reranking framework and to propose a unified QA framework named SIMULTQA to enable knowledge transfer and bridge the distinct supervisions between KB and text sources. Our second contribution is to extensively explore how knowledge is transferred by leveraging the pre-training and finetuning paradigm. We focus on the low-resource finetuning to show that pre-training SIMULTQA on one source can substantially improve its performance on the other source. More fine-grained analyses on transfer behaviors reveal the types of transferred knowledge and transfer patterns.

- Chapter 5 focuses on parameter-efficient transfer learning by transferring pre-trained prompts to new downstream tasks. Our key innovation lies in the pre-training phase where we study how to exploit the rich cross-task knowledge in multiple source tasks. There is no existing work trying to learn task-shared knowledge in multiple tasks through prompt tuning. We then propose multitask prompt tuning (MPT), which first learns a single transferable prompt by decomposing and distilling knowledge from multiple task-specific source prompts. We also learn multiplicative low-rank updates to this shared prompt to efficiently adapt it to each downstream target task.

MPT outperforms the state-of-the-art methods, including the full finetuning baseline in some cases, despite only tuning 0.035% as many task-specific parameters.

Part V, Knowledge Reasoning, is related to neuro-symbolic reasoning or broader areas of interpretable machine learning. Our target is to inject explicit knowledge reasoning into advanced AI systems to increase their transparency and performance. We summarize our major contributions as follows:

- Chapter 6 demonstrates the significance of explicit reasoning over structured knowledge for better model transparency and interpretability. We take inspiration from human cognitive theories, recall, and recognition, to formulate the basic knowledge units to be reasoned on. We apply this framework to relation prediction on the corpus-level statistics, i.e., a global co-occurrence graph of a text corpus, where we first *recall* rich contexts associated with the target entities, and then *recognize* relational interactions between these contexts to form model rationales to be conditioned on for the final prediction. Our key contribution is to bridge graph reasoning with NLP tasks with inspiration from human cognitive theory. Our second key contribution is to deeply collaborate with medical experts to verify our model rationales' usefulness for clinical decision-making.

- Chapter 7 proposes a novel graph reasoning technique, a path-based graph neural network, for interpretable machine learning with the help of explicit path-based reasoning. It leverages commonsense knowledge and reasoning to enhance word representation learning, where the reasoning is used to pre-train the word representations to be adapted into downstream tasks. The reasoning module mimics how humans practice their knowledge and propose a self-supervision task to learn how to reason

over the commonsense graph automatically. Moreover, our proposed model offers reasoning paths to justify word closeness and correlation with minimal knowledge barriers. Our key contributions include the novel technical contribution and the idea of using self-supervised signals to practice the reasoning ability of AI models.

8.2 Limitations and Future Work

Our long-term research goal is to develop generic or even plug-and-play knowledge modules empowering existing AI systems with the abilities of (1) dynamically searching for the necessary pieces of human knowledge based on the input, (2) jointly modeling the knowledge and data with a human-like reasoning process, and (3) generalizing universal and transferable knowledge to out-of-distribution samples and tasks. Our prior works have paved the foundations for these directions regarding acquisition, representation, transfer, and reasoning. We now discuss their limitations to sandbox our contributions and motivate promising future directions toward knowledge-centric NLP/AI systems.

Knowledge Acquisition. We observe two potential limitations to our current works. First, the process of knowledge construction is static and takes enormous efforts to improve the coverage of collected knowledge for downstream tasks. Second, the amount of structured knowledge from the text corpus is bounded by the size of the corpus, which is hard to scale up. Targeting these two limitations, we point out the following directions.

- **Inductively collecting the knowledge** for the end-to-end systems. Specifically, pruned knowledge should be constructed and provided for the target tasks. Such an end-to-end system requires dynamically searching for relevant knowledge and differentiably pruning the knowledge space with gradients from the downstream objective

functions. This will reuse techniques developed in *Knowledge Acquisition* with new pruning algorithms.

- **Interplay of LLMs and structured knowledge**. Two broad threads run through the works I have included above: *knowledge as a purpose of language learning*, and *knowledge as a regularization for augmenting and understanding existing AI systems*. These two can mutually enhance each other in an iterative way. First, extracting knowledge directly from LLMs can greatly help automate knowledge graph construction (LLMs $\rightarrow$ Knowledge), which can be scaled up easily given that LLMs have been pre-trained on massive text corpora. On the other hand, we can further leverage the extracted knowledge to probe, debug, edit, audit, and boost the model itself (Knowledge $\leftarrow$ LLMs). Such a closed-loop system can be iterative and open a new gate to deploy LLMs to more human-centered applications and scientific discovery problems.

Knowledge Representation. Our work on knowledge representation focuses on structured knowledge representation, e.g., graph structures, and we consider two possible limitations. First, Chapter 3 considers the context interactions between 1-hop neighbors while ignoring deeper interactions between higher-order neighborhood nodes. Second, we need to consider more types of knowledge for their representation learning, especially multimodal knowledge considering the interaction of structured knowledge with the real physical world. We thus outline two promising future directions to solve the above limitations.

- **Modeling deeper interactions of contexts on graphs when learning graph representations**. High-order neighbors on graphs can provide richer semantic information, while Chapter 3 only consider shallow interactions between 1-hop neighbors. This

requires considering higher-order context on graphs, such as multi-hop graph neural networks (Wang et al., 2021b), with multi-hop diffusion and interactions, which leads to better knowledge representations.

- **Learning multimodal knowledge representations with grounding in physical world**. We need to ground knowledge into multimodal worlds to learn more foundational semantic meanings since knowledge is not constrained in text or graph formats where a large portion of human knowledge can be grounded in the physical world. Learning representations jointly across modalities, such as images, text, and graphs, will greatly boost the coverage of universal knowledge to be transferred across multiple domains and tasks.

Knowledge Transfer. Our works in Part IV have studied two different transfer settings, one is to transfer knowledge between KBQA and TextQA on complex question answering, and the other is to transfer knowledge between generic NLP tasks. We believe there are three weaknesses that need to be further improved. First, our SIMULTQA system bridges KBQA and TextQA systems but runs parallelly on both knowledge sources, which only produces single-modality reasoning paths. Second, the transfer behaviors among generic NLP tasks are demonstrated empirically in Chapter 5 but have a lack of theoretical understanding. Third, as an analogy of human-like generalization in the few-shot setting, how to leverage knowledge transfer to greatly boost the few-shot adaptation to reach the full-dataset fine-tuning performance is underexplored. Therefore, we list the following future directions to address the above limitations.

- **Generating hybrid reasoning paths in SIMULTQA**. When producing reasoning paths for complex question answering, SIMULTQA cannot combine mixed sources

for the same question, i.e., the first and second hop need to be on the same source. This limits the further improvement of its performance. One future direction is to bridge the knowledge sources in the first place, e.g., using techniques proposed in Part II, which can enable deeper knowledge transfer. Then we can run SIMULTQA on the hybrid knowledge source to generate hybrid reasoning paths.

- **Theoretical understanding of transfer behaviors among tasks**. Though we showed great empirical results for transferring rich cross-task knowledge via multitask prompt tuning in Chapter 5, there is a lack of fundamental and theoretical understanding of how the transferring happens and how it is related to the task similarities between source and target tasks, There are existing theoretical works on explaining how transfer learning is related to task diversity (Tripuraneni et al., 2020), which can be borrowed and adjusted to our problem.

- **Improving few-shot adaptation by transferring knowledge from the general domain to specific domains**. One of the most advantages of human learning is learning efficiency where enormous background world knowledge can be shared and generalized to new tasks with only a few samples. We provided initial few-shot learning results in Chapter 5 (Section 5.4.7), but there is still a large gap between few-shot performance and full-dataset fine-tuning and we need to study how to practice few-shot adaptation on multiple source tasks, e.g., meta-datasets (Triantafillou et al., 2019) so that the parameters can be quickly adapted to new target tasks with only a few samples.

Knowledge Reasoning. Our works in this dissertation on knowledge reasoning mainly consider data-driven approaches that perform explicit reasoning in the space of structured

knowledge and induce greater transparency and interpretability to AI systems. Potential limitations include that we need to consider more diverse types of AI systems and reasoning algorithms, such as injecting more explicit reasoning into LLMs, and we also need to apply reasoning into the few-shot generalization similar to how humans can leverage logical inference to produce new knowledge.

- **Advanced neuro-symbolic reasoning to slow down the fast thinking of LLMs**. Given the great power of in-context learning of LLMs, they often fail in scenarios that require reasoning over symbolic knowledge, or over rarely re-iterated common knowledge. Thus, we can enforce a slow thinking process, i.e., human-like reasoning, into LLMs to make them more robust to in-context perturbation and more interpretable (Ozturkler et al., 2022).

- **Reasoning-based domain adaptation for better efficiency and robustness.** In addition to providing transparency and interpretability, reasoning for humans is also a very useful tool to navigate through new environments and generalize to new samples. For example, for humans, it is a common practice to start learning from established knowledge and easily adapt the knowledge to new situations with few samples by reasoning over existing knowledge. One key insight is that this reasoning-based adaptation favors local adjustment in a multi-hop reasoning process for effective transfer learning, i.e., only adjusting a few hops in the reasoning chain to adapt new samples and not hurt the whole reasoning process. Thus, one promising direction is to formulate the reasoning-based adaptation process as a meta-learning problem and demonstrate we can generalize to new samples with the bridge of this explicit reasoning, which is expected to work in the few-shot domain and improve the model robustness.

Bibliography

Sami Abu-El-Haija, Bryan Perozzi, and Rami Al-Rfou. 2017. Learning edge representations via low-rank asymmetric projections. In *CIKM*.

Accountability Act. 1996. Health insurance portability and accountability act of 1996. *Public law*, 104:191.

Lada A Adamic and Eytan Adar. 2003. Friends and neighbors on the web. *Social networks*, 25(3):211–230.

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021a. Muppet: Massive multi-task representations with pre-finetuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5799–5811.

Armen Aghajanyan, Sonal Gupta, and Luke Zettlemoyer. 2021b. Intrinsic dimensionality explains the effectiveness of language model fine-tuning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 7319–7328.

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *International Conference on Learning Representations*.

Akari Asai, Mohammadreza Salehi, Matthew E Peters, and Hannaneh Hajishirzi. 2022. Attentional mixtures of soft prompt tuning for parameter-efficient multi-task knowledge sharing. *arXiv preprint arXiv:2205.11961*.

Aitziber Atutxa, Arantza Díaz de Ilarraza, Koldo Gojenola, Maite Oronoz, and Olatz Perez-de Viñaspre. 2019. Interpretable deep learning to map diagnostic texts to icd-10 codes. *International Journal of Medical Informatics*, 129:49–59.

Goonmeet Bajaj, Sean Current, Daniel Schmidt, Bortik Bandyopadhyay, Christopher W Myers, and Srinivasan Parthasarathy. 2022. Knowledge gaps: A challenge for agent-based automatic task completion. *Topics in Cognitive Science*, 14(4):780–799.

M. Ballesteros, C. Dyer, and N. A. Smith. 2015. Improved transition-based parsing by modeling characters instead of words with lstms. In *EMNLP*.

Sambaran Bandyopadhyay, Anirban Biswas, MN Murty, and Ramasuri Narayanam. 2019. Beyond node embedding: A direct unsupervised edge representation framework for homogeneous networks. *arXiv preprint arXiv:1912.05140*.

Pratyay Banerjee, Kuntal Kumar Pal, Arindam Mitra, and Chitta Baral. 2019. Careful selection of knowledge to solve open book question answering. *arXiv preprint arXiv:1907.10738*.

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2503–2514.

Yujia Bao, Shiyu Chang, Mo Yu, and Regina Barzilay. 2018. Deriving machine attention from human rationales. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1903–1913.

Virginia E Barker, Dennis E O'Connor, Judith Bachant, and Elliot Soloway. 1989. Expert systems for configuration at digital: Xcon and beyond. *Communications of the ACM*, 32(3):298–318.

Jim Barnett, Kevin Knight, Inderjeet Mani, and Elaine Rich. 1990. Knowledge and natural language processing. *Communications of the ACM*, 33(8):50–71.

A. L. Beam, B. Kompa, I. Fried, N. P. Palmer, X. Shi, T. Cai, and I. S. Kohane. 2018. Clinical concept embeddings learned from massive sources of medical data. *arXiv preprint arXiv:1804.01486*.

Yonatan Belinkov, Sebastian Gehrmann, and Ellie Pavlick. 2020. Interpretability and analysis in neural nlp. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts*, pages 1–5.

Mikhail Belkin and Partha Niyogi. 2002. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in neural information processing systems*, pages 585–591.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544.

Derek Bickerton. 2017. *Language and human behavior*. University of Washington Press.

O. Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov. 2016. Enriching word vectors with subword information. *TACL*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Danushka Bollegala, Alsuhaibani Mohammed, Takanori Maehara, and Ken-ichi Kawarabayashi. 2015. Joint word representation learning using a corpus and a semantic lexicon. *arXiv preprint arXiv:1511.06438*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Antoine Bosselut, Hannah Rashkin, Maarten Sap, Chaitanya Malaviya, Asli Celikyilmaz, and Yejin Choi. 2019. Comet: Commonsense transformers for automatic knowledge graph construction. *arXiv preprint arXiv:1906.05317*.

Diane Bouchacourt and Ludovic Denoyer. 2019. Educe: Explaining model decisions through unsupervised concepts extraction. *arXiv preprint arXiv:1905.11852*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Raluca Budiu. 2014. Memory recognition and recall in user interfaces. *Nielsen Norman Group*.

Shaosheng Cao, Wei Lu, and Qiongkai Xu. 2016. Deep neural networks for learning graph representations. In *AAAI*.

Z. Cao, T. Qin, T. Liu, M. Tsai, and H. Li. 2007. Learning to rank: from pairwise approach to listwise approach. In *ICML*.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R Hruschka, and Tom M Mitchell. 2010. Toward an architecture for never-ending language learning. In *Twenty-Fourth AAAI conference on artificial intelligence*.

Ohio Supercomputer Center. 1987. Ohio supercomputer center.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*.

Ismail Ilkan Ceylan, Adnan Darwiche, and Guy Van den Broeck. 2016. Open-world probabilistic databases. In *Fifteenth International Conference on the Principles of Knowledge Representation and Reasoning*.

Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. In *NeurIPs*, pages 4869–4880.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879.

Wenhu Chen, Ming-Wei Chang, Eva Schlinger, William Wang, and William W Cohen. 2020a. Open question answering over tables and text. *arXiv preprint arXiv:2010.10439*.

Wenhu Chen, Wenhan Xiong, Xifeng Yan, and William Wang. 2018. Variational knowledge graph reasoning. *arXiv preprint arXiv:1803.06581*.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020b. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 1026–1036.

Zi-Yuan Chen, Chih-Hung Chang, Yi-Pei Chen, Jijnasa Nayak, and Lun-Wei Ku. 2019. Uhop: An unrestricted-hop relation extraction framework for knowledge-based question answering. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 345–356.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, 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)*, pages 1724–1734.

William J Clancey and Reed Letsinger. 1982. *NEOMYCIN: Reconfiguring a rule-based expert system for application to teaching*. Department of Computer Science, Stanford University Stanford, CA, USA.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019a. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Kevin Clark, Minh-Thang Luong, Urvashi Khandelwal, Christopher D Manning, and Quoc Le. 2019b. Bam! born-again multi-task networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5931–5937.

Peter Clark, Isaac Cowhey, Oren Etzioni, Tushar Khot, Ashish Sabharwal, Carissa Schoenick, and Oyvind Tafjord. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Jordan Clive, Kris Cao, and Marek Rei. 2021. Control prefixes for text generation. *arXiv preprint arXiv:2110.08329*.

Allan M Collins and Elizabeth F Loftus. 1975. A spreading-activation theory of semantic processing. *Psychological review*, 82(6):407.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of machine learning research*, 12(ARTICLE):2493–2537.

Alexis Conneau, Germán Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. 2018. What you can cram into a single vector: Probing sentence embeddings for linguistic properties. *arXiv preprint arXiv:1805.01070*.

Michael C Corballis. 2014. *The recursive mind*. Princeton University Press.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2018. Multi-step retriever-reader interaction for scalable open-domain question answering. In *International Conference on Learning Representations*.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2017a. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. *arXiv preprint arXiv:1711.05851*.

Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2017b. Chains of reasoning over entities, relations, and text using recurrent neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 132–141.

Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. 2017c. Question answering on knowledge bases and text using universal schema and memory networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 358–365.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9594–9611.

Ernest Davis and Gary Marcus. 2015. Commonsense reasoning and commonsense knowledge in artificial intelligence. *Communications of the ACM*, 58(9):92–103.

Marie-Catherine De Marneffe, Mandy Simons, and Judith Tonhauser. 2019. The commitmentbank: Investigating projection in naturally occurring discourse. In *proceedings of Sinn und Bedeutung*, volume 23, pages 107–124.

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets. *arXiv preprint arXiv:1809.02922*.

Xiang Deng and Huan Sun. 2019. Leveraging 2-hop distant supervision from table entity pairs for relation extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 410–420.

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

Bhuwan Dhingra, Manzil Zaheer, Vidhisha Balachandran, Graham Neubig, Ruslan Salakhutdinov, and William W. Cohen. 2020. Differentiable reasoning over a virtual knowledge base. In *International Conference on Learning Representations*.

Ning Ding, Yujia Qin, Guang Yang, Fuchao Wei, Zonghan Yang, Yusheng Su, Shengding Hu, Yulin Chen, Chi-Min Chan, Weize Chen, et al. 2022. Delta tuning: A comprehensive study of parameter efficient methods for pre-trained language models. *arXiv preprint arXiv:2203.06904*.

Bill Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*.

D. A. Dorr, W.F. Phillips, S. Phansalkar, S. A. Sims, and J. F. Hurdle. 2006. Assessing the difficulty and time cost of de-identification in clinical narratives. *Methods of information in medicine*.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of NAACL-HLT*, pages 2368–2378.

Nan Duan, Duyu Tang, and Ming Zhou. 2020. Machine reasoning: Technology, dilemma and future. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 1–6.

Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Alessandro Epasto and Bryan Perozzi. 2019. Is a single embedding enough? learning node representations that capture multiple social contexts. In *WWW*.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. 2020. Hierarchical graph network for multi-hop question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8823–8838.

Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2014. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166*.

Yair Feldman and Ran El-Yaniv. 2019. Multi-hop paragraph retrieval for open-domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2296–2309.

Samuel G Finlayson, Paea LePendu, and Nigam H Shah. 2014. Building the graph of medicine from millions of clinical narratives. *Scientific data*, 1:140032.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of 2nd Machine Reading for Reading Comprehension (MRQA) Workshop at EMNLP*.

Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. 2018. Born again neural networks. In *International Conference on Machine Learning*, pages 1607–1616.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3816–3830.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*.

Matt Gardner, Partha Talukdar, Jayant Krishnamurthy, and Tom Mitchell. 2014. Incorporating vector space similarity in random walk inference over knowledge bases. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 397–406.

S. L Garfinkel. 2015. De-identification of personal information. *NISTIR*.

Michael S Gazzaniga, Richard B Ivry, and GR Mangun. 2006. Cognitive neuroscience. the biology of the mind,(2014).

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9.

Gary Gillund and Richard M Shiffrin. 1984. A retrieval model for both recognition and recall. *Psychological review*, 91(1):1.

Justin Gilmer, Samuel S Schoenholz, Patrick F Riley, Oriol Vinyals, and George E Dahl. 2017. Neural message passing for quantum chemistry. *arXiv preprint arXiv:1704.01212*.

Yoav Goldberg. 2016. A primer on neural network models for natural language processing. *Journal of Artificial Intelligence Research*, 57:345–420.

Ira Goldstein and Seymour Papert. 1977. Artificial intelligence, language, and the study of knowledge. *Cognitive science*, 1(1):84–123.

W. H. Gomaa and A. A. Fahmy. 2013. A survey of text similarity approaches. In *IJCA*.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*, volume 1. MIT Press.

Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. 2021. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129(6):1789–1819.

A. Grover and J. Leskovec. 2016a. node2vec: Scalable feature learning for networks. In *KDD*.

Aditya Grover and Jure Leskovec. 2016b. Node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 855–864.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2020. Beyond iid: Three levels of generalization for question answering on knowledge bases. *arXiv e-prints*, pages arXiv–2011.

Jian Guan, Yansen Wang, and Minlie Huang. 2019. Story ending generation with incremental encoding and commonsense knowledge. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6473–6480.

Riccardo Guidotti, Anna Monreale, Salvatore Ruggieri, Dino Pedreschi, Franco Turini, and Fosca Giannotti. 2018. Local rule-based explanations of black box decision systems. *arXiv preprint arXiv:1805.10820*.

Demi Guo, Alexander M. Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *Proceedings of ACL*.

Kelvin Guu, John Miller, and Percy Liang. 2015. Traversing knowledge graphs in vector space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 318–327.

M. Hagiwara, Y. Ogawa, and K. Toyama. 2009. Supervised synonym acquisition using distributional features and syntactic patterns. *IMT*.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, pages 1024–1034.

Xiaochuang Han, Byron C Wallace, and Yulia Tsvetkov. 2020. Explaining black box predictions and unveiling data artifacts through influence functions. *arXiv preprint arXiv:2005.06676*.

Zellig S Harris. 1954. Distributional structure. *Word*, 10(2-3):146–162.

K. Hashimoto, Y. Tsuruoka, R. Socher, and o. 2017. A joint many-task model: Growing a neural network for multiple nlp tasks. In *ACL*.

Bin He, Yi Guan, and Rui Dai. 2019. Classifying medical relations in clinical text via convolutional neural networks. *Artificial Intelligence in Medicine*, 93:43–49.

Yun He, Steven Zheng, Yi Tay, Jai Gupta, Yu Du, Vamsi Aribandi, Zhe Zhao, YaGuang Li, Zhao Chen, Donald Metzler, et al. 2022. Hyperprompt: Prompt-based task-conditioning of transformers. In *International Conference on Machine Learning*, pages 8678–8690.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 33–38.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799.

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

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2021. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Minghao Hu, Yuxing Peng, Furu Wei, Zhen Huang, Dongsheng Li, Nan Yang, and Ming Zhou. 2018. Attention-guided answer distillation for machine reading comprehension. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2077–2086.

Weihua Hu*, Bowen Liu*, Joseph Gomes, Marinka Zitnik, Percy Liang, Vijay Pande, and Jure Leskovec. 2020. Strategies for pre-training graph neural networks. In *ICLR*.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. 2016. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer.

Qiang Huang, Makoto Yamada, Yuan Tian, Dinesh Singh, Dawei Yin, and Yi Chang. 2020. Graphlime: Local interpretable model explanations for graph neural networks. *arXiv preprint arXiv:2001.06216*.

Hamish Ivison and Matthew E Peters. 2022. Hyperdecoders: Instance-specific decoders for multi-task nlp. *arXiv preprint arXiv:2203.08304*.

Xiaoqi Jiao, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. Tinybert: Distilling bert for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174.

Xisen Jin, Zhongyu Wei, Junyi Du, Xiangyang Xue, and Xiang Ren. 2020. Towards hierarchical importance attribution: Explaining compositional semantics for neural sequence models. In *International Conference on Learning Representations*.

Mandar Joshi, Eunsol Choi, Omer Levy, Daniel Weld, and Luke Zettlemoyer. 2019. pair2vec: Compositional word-pair embeddings for cross-sentence inference. In *NAACL*.

Rudolf Kadlec, Ondrej Bajgar, and Jan Kleindienst. 2017. Knowledge base completion: Baselines strike back. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 69–74.

Michael Kahana, Marc Howard, and Sean Polyn. 2008. Associative retrieval processes in episodic memory. *Psychology*.

Rabeeh Karimi Mahabadi, James Henderson, and Sebastian Ruder. 2021. Compacter: Efficient low-rank hypercomplex adapter layers. *Advances in Neural Information Processing Systems*, 34:1022–1035.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262.

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. Scitail: A textual entailment dataset from science question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Y. Kim, Y. Jernite, D. Sontag, and A. M. Rush. 2016. Character-aware neural language models. In *AAAI*.

D. P. Kingma and J. Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*.

Thomas N Kipf and Max Welling. 2016. Variational graph auto-encoders. *NIPS Workshop on Bayesian Deep Learning*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *ICLR*.

Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. *arXiv preprint arXiv:1703.04730*.

István A Kovács, Katja Luck, Kerstin Spirohn, Yang Wang, Carl Pollis, Sadie Schlabach, Wenting Bian, Dae-Kyum Kim, Nishka Kishore, Tong Hao, et al. 2019. Network-based prediction of protein interactions. *Nature communications*, 10(1):1–8.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 969–974.

Yunshi Lan, Shuohang Wang, and Jing Jiang. 2019. Multi-hop knowledge base question answering with an iterative sequence matching model. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 359–368. IEEE.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature*, 521(7553):436–444.

Jinhyuk Lee, Seongjun Yun, Hyunjae Kim, Miyoung Ko, and Jaewoo Kang. 2018. Ranking paragraphs for improving answer recall in open-domain question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 565–569.

Joohong Lee, Sangwoo Seo, and Yong Suk Choi. 2019. Semantic relation classification via bidirectional lstm networks with entity-aware attention using latent entity typing. *Symmetry*, 11(6):785.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic web*, 6(2):167–195.

Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2016. Rationalizing neural predictions. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 107–117.

Douglas B Lenat. 1995. Cyc: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38.

Benjamin J Lengerich, Andrew L Maas, and Christopher Potts. 2017. Retrofitting distributional embeddings to knowledge graphs with functional relations. *arXiv preprint arXiv:1708.00112*.

Paea LePendu, Srinivasan V Iyer, Cédrick Fairon, and Nigam H Shah. 2012. Annotation analysis for testing drug safety signals using unstructured clinical notes. In *Journal of biomedical semantics*, volume 3, page S5. BioMed Central.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059.

Hector Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. In *Thirteenth international conference on the principles of knowledge representation and reasoning*.

O. Levy and Y. Goldberg. 2014a. Linguistic regularities in sparse and explicit word representations. In *ACL*.

Omer Levy and Yoav Goldberg. 2014b. Neural word embedding as implicit matrix factorization. *Advances in neural information processing systems*, 27.

Chunyuan Li, Heerad Farkhoor, Rosanne Liu, and Jason Yosinski. 2018. Measuring the intrinsic dimension of objective landscapes. In *International Conference on Learning Representations*.

Jiwei Li, Will Monroe, and Dan Jurafsky. 2016. Understanding neural networks through representation erasure. *arXiv preprint arXiv:1612.08220*.

Pan Li, Yanbang Wang, Hongwei Wang, and Jure Leskovec. 2020. Distance encoding: Design provably more powerful neural networks for graph representation learning. *Advances in Neural Information Processing Systems*, 33:4465–4478.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. 2019. Graph matching networks for learning the similarity of graph structured objects. In *ICML*.

J. Liang, P. Jacobs, J. Sun, and S. Parthasarathy. 2018. Semi-supervised embedding in attributed networks with outliers. In *SDM*.

David Liben-Nowell and Jon Kleinberg. 2007. The link-prediction problem for social networks. *Journal of the American society for information science and technology*, 58(7):1019–1031.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. *arXiv preprint arXiv:1909.02151*.

Xi Victoria Lin, Richard Socher, and Caiming Xiong. 2018. Multi-hop knowledge graph reasoning with reward shaping. *arXiv preprint arXiv:1808.10568*.

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133.

Haokun Liu, Derek Tam, Mohammed Muqeeth, Jay Mohta, Tenghao Huang, Mohit Bansal, and Colin Raffel. 2022a. Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning. *arXiv preprint arXiv:2205.05638*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021a. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Quan Liu, Hui Jiang, Zhen-Hua Ling, Xiaodan Zhu, Si Wei, and Yu Hu. 2016. Commonsense knowledge enhanced embeddings for solving pronoun disambiguation problems in winograd schema challenge. *arXiv preprint arXiv:1611.04146*.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. In *AAAI*, pages 2901–2908.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2021b. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022b. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68.

Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2019. Multi-task deep neural networks for natural language understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4487–4496.

Peter LoBue and Alexander Yates. 2011. Types of common-sense knowledge needed for recognizing textual entailment. In *Proceedings of the 49th annual meeting of the association for computational linguistics: human language technologies*, pages 329–334.

Robert L Logan IV, Nelson F Liu, Matthew E Peters, Matt Gardner, and Sameer Singh. 2019. Barack's wife hillary: Using knowledge-graphs for fact-aware language modeling. *arXiv preprint arXiv:1906.07241*.

H. J. Lowe, T. A. Ferris, P. M. Hernandez, and S. C. Weber. 2009. Stride–an integrated standards-based translational research informatics platform. In *AMIA*.

Ryan Lowe, Nissan Pow, Iulian Serban, Laurent Charlin, and Joelle Pineau. 2015. Incorporating unstructured textual knowledge sources into neural dialogue systems. In *Neural information processing systems workshop on machine learning for spoken language understanding*.

Kevin Lund and Curt Burgess. 1996. Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior research methods, instruments, & computers*, 28(2):203–208.

Scott M Lundberg and Su-In Lee. 2017. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30.

Kangqi Luo, Fengli Lin, Xusheng Luo, and Kenny Zhu. 2018. Knowledge base question answering via encoding of complex query graphs. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2185–2194.

Xinbo Lv, Yi Guan, Jinfeng Yang, and Jiawei Wu. 2016. Clinical relation extraction with deep learning. *International Journal of Hybrid Information Technology*, 9(7):237–248.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 565–576.

Gaurav Maheshwari, Priyansh Trivedi, Denis Lukovnikov, Nilesh Chakraborty, Asja Fischer, and Jens Lehmann. 2019. Learning to rank query graphs for complex question

answering over knowledge graphs. In *International semantic web conference*, pages 487–504. Springer.

Nikolay Malkin, Zhen Wang, and Nebojsa Jojic. 2022. Coherence boosting: When your pretrained language model is not paying enough attention. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8214–8236.

Christopher Manning and Hinrich Schutze. 1999. *Foundations of statistical natural language processing*. MIT press.

Yuning Mao, Lambert Mathias, Rui Hou, Amjad Almahairi, Hao Ma, Jiawei Han, Scott Yih, and Madian Khabsa. 2022. Unipelt: A unified framework for parameter-efficient language model tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6253–6264.

Y. Matsuo, T. Sakaki, and K. Uchiyama. 2006. Graph-based word clustering using a web search engine. In *EMNLP*.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. 2018. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391.

Todor Mihaylov and Anette Frank. 2018. Knowledgeable reader: Enhancing cloze-style reading comprehension with external commonsense knowledge. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 821–832.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Tim Miller. 2019. Explanation in artificial intelligence: Insights from the social sciences. *Artificial Intelligence*, 267:1–38.

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019. Knowledge guided text retrieval and reading for open domain question answering. *arXiv preprint arXiv:1911.03868*.

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011.

Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3994–4003.

Lingbo Mo, Zhen Wang, Jie Zhao, and Huan Sun. 2022. Knowledge transfer between structured and unstructured sources for complex question answering. In *Proceedings*

of the Workshop on Structured and Unstructured Knowledge Integration (SUKI), pages 55–66.

J. Mueller and A. Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *AAAI*.

James Mullenbach, Sarah Wiegreffe, Jon Duke, Jimeng Sun, and Jacob Eisenstein. 2018. Explainable prediction of medical codes from clinical text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1101–1111.

W. James Murdoch, Chandan Singh, Karl Kumbier, Reza Abbasi-Asl, and Bin Yu. 2019. Definitions, methods, and applications in interpretable machine learning. *Proceedings of the National Academy of Sciences*, 116(44):22071–22080.

P. Neculoiu, M. Versteegh, and M. Rotaru. 2016. Learning text similarity with siamese recurrent networks. In *Workshop on Representation Learning for NLP*.

Arvind Neelakantan, Benjamin Roth, and Andrew McCallum. 2015. Compositional vector space models for knowledge base completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 156–166.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, page 809–816.

Yixin Nie, Songhe Wang, and Mohit Bansal. 2019. Revealing the importance of semantic retrieval for machine reading at scale. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2553–2566.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial nli: A new benchmark for natural language understanding. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4885–4901.

Peter A Nobel and Richard M Shiffrin. 2001. Retrieval processes in recognition and cued recall. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 27(2):384.

Jekaterina Novikova, Ondřej Dušek, and Verena Rieser. 2017. The e2e dataset: New challenges for end-to-end generation. *arXiv preprint arXiv:1706.09254*.

Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Scott Yih. 2020. Unified open-domain question answering with structured and unstructured knowledge. *arXiv preprint arXiv:2012.14610*.

Batu Ozturkler, Nikolay Malkin, Zhen Wang, and Nebojsa Jojic. 2022. Thinksum: Probabilistic reasoning over sets using large language models. *arXiv preprint arXiv:2210.01293*.

S. V. Pakhomov, G. Finley, R. McEwan, Y. Wang, and G. B. Melton. 2016. Corpus domain effects on distributional semantic modeling of medical terms. *Bioinformatics*, 32(23):3635–3644.

Sinno Jialin Pan and Qiang Yang. 2009. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

Chanyoung Park, Donghyun Kim, Qi Zhu, Jiawei Han, and Hwanjo Yu. 2019. Task-guided pair embedding in heterogeneous network. In *CIKM*.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, et al. 2017. Automatic differentiation in pytorch. In *NIPS-W*.

Judea Pearl. 2014. *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Elsevier.

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

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

Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, page 701–710.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Matthew E Peters, Mark Neumann, Robert L Logan IV, Roy Schwartz, Vidur Joshi, Sameer Singh, and Noah A Smith. 2019. Knowledge enhanced contextual word representations. *arXiv preprint arXiv:1909.04164*.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2018. Wic: the word-in-context dataset for evaluating context-sensitive meaning representations. *arXiv preprint arXiv:1808.09121*.

Edoardo M Ponti, Alessandro Sordoni, and Siva Reddy. 2022. Combining modular skills in multitask learning. *arXiv preprint arXiv:2202.13914*.

David Premack. 2004. Is language the key to human intelligence? *Science*, 303(5656):318–320.

Peng Qi, Haejun Lee, Oghenetegiri Sido, Christopher D Manning, et al. 2020. Retrieve, rerank, read, then iterate: Answering open-domain questions of arbitrary complexity from text. *arXiv preprint arXiv:2010.12527*.

Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D Manning. 2019. Answering complex open-domain questions through iterative query generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2590–2602.

Jiezhong Qiu, Yuxiao Dong, Hao Ma, Jian Li, Kuansan Wang, and Jie Tang. 2018. Network embedding as matrix factorization: Unifying deepwalk, line, pte, and node2vec. In *WSDM*.

Meng Qu, Xiang Ren, and Jiawei Han. 2017. Automatic synonym discovery with knowledge bases. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 997–1005. ACM.

Meng Qu, Xiang Ren, Yu Zhang, and Jiawei Han. 2018. Weakly-supervised relation extraction by pattern-enhanced embedding learning. In *Proceedings of the 2018 World Wide Web Conference*, WWW '18, page 1257–1266.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

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

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

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

Raymond Reiter. 1981. On closed world data bases. In *Readings in artificial intelligence*, pages 119–140. Elsevier.

Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *KDD*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "why should i trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 1135–1144.

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Proceedings of the 2010 European Conference on Machine Learning and Knowledge Discovery in Databases: Part III*, ECML PKDD'10, page 148–163.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84.

Sebastian Ruder. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*.

Sebastian Ruder, Joachim Bingel, Isabelle Augenstein, and Anders Søgaard. 2019a. Latent multi-task architecture learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4822–4829.

Sebastian Ruder, Matthew E Peters, Swabha Swayamdipta, and Thomas Wolf. 2019b. Transfer learning in natural language processing. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics:*

Tutorials, pages 15–18.

Cynthia Rudin. 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215.

Sunil Sahu, Ashish Anand, Krishnadev Oruganty, and Mahanandeeshwar Gattu. 2016. Relation extraction from clinical texts using domain invariant convolutional neural network. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, pages 206–215.

Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*, 64(9):99–106.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Victor Sanh, Albert Webson, Colin Raffel, Stephen Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, et al. 2022. Multitask prompted training enables zero-shot task generalization. In *The Tenth International Conference on Learning Representations*.

Cicero dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. 2016. Attentive pooling networks. *arXiv preprint arXiv:1602.03609*.

Maarten Sap, Ronan Le Bras, Emily Allaway, Chandra Bhagavatula, Nicholas Lourie, Hannah Rashkin, Brendan Roof, Noah A Smith, and Yejin Choi. 2019. Atomic: An atlas of

machine commonsense for if-then reasoning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3027–3035.

Timo Schick and Hinrich Schütze. 2021a. Exploiting cloze-questions for few-shot text classification and natural language inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 255–269.

Timo Schick and Hinrich Schütze. 2021b. It's not just size that matters: Small language models are also few-shot learners. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2339–2352.

Siamak Shakeri, Abhinav Sethy, and Cheng Cheng. 2019. Knowledge distillation in document retrieval. *arXiv preprint arXiv:1911.11065*.

J. Shen, R. Lv, X. Ren, M. Vanni, B. Sadler, and J. Han. 2019. Mining entity synonyms with efficient neural set generation. In *AAAI*.

Yelong Shen, Jianshu Chen, Po-Sen Huang, Yuqing Guo, and Jianfeng Gao. 2018. M-walk: Learning to walk over graphs using monte carlo tree search. In *Advances in Neural Information Processing Systems*, pages 6786–6797.

Jiaxin Shi, Shulin Cao, Lei Hou, Juanzi Li, and Hanwang Zhang. 2021. Transfernet: An effective and transparent framework for multi-hop question answering over relation graph. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4149–4158.

Yu Shi, Qi Zhu, Fang Guo, Chao Zhang, and Jiawei Han. 2018. Easing embedding learning by comprehensive transcription of heterogeneous information networks. In *KDD*.

Sam Shleifer and Alexander M Rush. 2020. Pre-trained summarization distillation. *arXiv preprint arXiv:2010.13002*.

Edward Hance Shortliffe. 1974. Mycin: a rule-based computer program for advising physicians regarding antimicrobial therapy selection. Technical report, STANFORD UNIV CALIF DEPT OF COMPUTER SCIENCE.

Vered Shwartz, Yoav Goldberg, and Ido Dagan. 2016. Improving hypernymy detection with an integrated path-based and distributional method. In *ACL*.

Chandan Singh, W. James Murdoch, and Bin Yu. 2019. Hierarchical interpretations for neural network predictions. In *International Conference on Learning Representations*.

Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. 2013a. Reasoning with neural tensor networks for knowledge base completion. *Advances in neural information processing systems*, 26.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Linfeng Song, Yue Zhang, Daniel Gildea, Mo Yu, Zhiguo Wang, and Jinsong Su. 2019. Leveraging dependency forest for neural medical relation extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the*

9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 208–218.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Thirty-first AAAI conference on artificial intelligence*.

Donald P Spence and Kimberly C Owens. 1990. Lexical co-occurrence and association strength. *Journal of Psycholinguistic Research*, 19(5):317–330.

Josua Stadelmaier and Sebastian Padó. 2019. Modeling paths for explainable knowledge base completion. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 147–157.

A. Stubbs and Ö. Uzuner. 2015. Annotating longitudinal clinical narratives for de-identification: The 2014 i2b2/uthealth corpus. *Journal of biomedical informatics*, 58:S20–S29.

Yusheng Su, Xiaozhi Wang, Yujia Qin, Chi-Min Chan, Yankai Lin, Huadong Wang, Kaiyue Wen, Zhiyuan Liu, Peng Li, Juanzi Li, et al. 2022. On transferability of prompt tuning for natural language processing. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3949–3969.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. 2019a. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2380–2390.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4231–4242.

Ximeng Sun, Rameswar Panda, Rogerio Feris, and Kate Saenko. 2020a. Adashare: Learning what to share for efficient deep multi-task learning. *Advances in Neural Information Processing Systems*, 33:8728–8740.

Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019b. Ernie: Enhanced representation through knowledge integration. *arXiv preprint arXiv:1904.09223*.

Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. 2020b. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *AAAI*.

Yi-Lin Sung, Varun Nair, and Colin A Raffel. 2021. Training neural networks with fixed sparse masks. In *Advances in Neural Information Processing Systems*, volume 34, pages 24193–24205. Curran Associates, Inc.

William Swartout, Cecile Paris, and Johanna Moore. 1991. Explanations in knowledge systems: Design for explainable expert systems. *IEEE Expert*, 6(3):58–64.

C. N. Ta, M. Dumontier, G. Hripcsak, N. P. Tatonetti, and C. Weng. 2018. Columbia open health data, clinical concept prevalence and co-occurrence from electronic health records. *Scientific data*, 5:180273.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2018. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *arXiv preprint arXiv:1811.00937*.

Niket Tandon, Gerard De Melo, and Gerhard Weikum. 2017. Webchild 2.0: Fine-grained commonsense knowledge distillation. In *Proceedings of ACL 2017, System Demonstrations*, pages 115–120.

Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, WWW '15, page 1067–1077.

Ian Tenney, Patrick Xia, Berlin Chen, Alex Wang, Adam Poliak, R Thomas McCoy, Najoung Kim, Benjamin Van Durme, Samuel R Bowman, Dipanjan Das, et al. 2019. What do you learn from context? probing for sentence structure in contextualized word representations. *arXiv preprint arXiv:1905.06316*.

Kristina Toutanova and Danqi Chen. 2015. Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66.

Kristina Toutanova, Xi Victoria Lin, Wen-tau Yih, Hoifung Poon, and Chris Quirk. 2016. Compositional learning of embeddings for relation paths in knowledge base and text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1434–1444.

Eleni Triantafillou, Tyler Zhu, Vincent Dumoulin, Pascal Lamblin, Utku Evci, Kelvin Xu, Ross Goroshin, Carles Gelada, Kevin Swersky, Pierre-Antoine Manzagol, et al. 2019. Meta-dataset: A dataset of datasets for learning to learn from few examples. In *International Conference on Learning Representations*.

Nilesh Tripuraneni, Michael Jordan, and Chi Jin. 2020. On the theory of transfer learning: The importance of task diversity. *Advances in Neural Information Processing Systems*, 33:7852–7862.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200.

Anton Tsitsulin, Davide Mottin, Panagiotis Karras, and Emmanuel Müller. 2018. Verse: Versatile graph embeddings from similarity measures. In *WWW*.

Cunchao Tu, Han Liu, Zhiyuan Liu, and Maosong Sun. 2017. Cane: Context-aware network embedding for relation modeling. In *ACL*.

Özlem Uzuner, Brett R South, Shuying Shen, and Scott L DuVall. 2011. 2010 i2b2/va challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556.

Shikhar Vashishth, Shyam Upadhyay, Gaurav Singh Tomar, and Manaal Faruqui. 2019. Attention interpretability across nlp tasks. *arXiv preprint arXiv:1909.11218*.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. Attention is all you need. In *NeurIPS*.

P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio. 2018. Graph attention networks. In *ICLR*.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. In *ICLR*.

Patrick Verga, Emma Strubell, and Andrew McCallum. 2018. Simultaneously self-attending to all mentions for full-abstract biological relation extraction. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 872–884.

Janu Verma, Srishti Gupta, Debdoot Mukherjee, and Tanmoy Chakraborty. 2019. Heterogeneous edge embedding for friend recommendation. In *ECIR*.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. 2022. Spot: Better frozen model adaptation through soft prompt transfer. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5039–5059.

Tu Vu, Tong Wang, Tsendsuren Munkhdalai, Alessandro Sordoni, Adam Trischler, Andrew Mattarella-Micke, Subhransu Maji, and Mohit Iyyer. 2020. Exploring and predicting transferability across nlp tasks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7882–7926.

Eric Wallace, Matt Gardner, and Sameer Singh. 2020. Interpreting predictions of nlp models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Tutorial Abstracts*, pages 20–23.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018a. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

C. Wang, L. Cao, and B. Zhou. 2015a. Medical synonym extraction with concept space models. In *IJCAI*.

Chang Wang, Liangliang Cao, and James Fan. 2016a. Building joint spaces for relation extraction. In *IJCAI*, pages 2936–2942.

Chang Wang and James Fan. 2014. Medical relation extraction with manifold models. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 828–838.

Chengyu Wang, Jianing Wang, Minghui Qiu, Jun Huang, and Ming Gao. 2021a. Transprompt: Towards an automatic transferable prompting framework for few-shot text classification. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2792–2802.

Daixin Wang, Peng Cui, and Wenwu Zhu. 2016b. Structural deep network embedding. In *KDD*.

Guangtao Wang, Rex Ying, Jing Huang, and Jure Leskovec. 2021b. Multi-hop attention graph neural networks. In *IJCAI*.

Peng Wang, BaoWen Xu, YuRong Wu, and XiaoYu Zhou. 2015b. Link prediction in social networks: the state-of-the-art. *Science China Information Sciences*, 58(1):1–38.

Q. Wang, B. Wang, and L. Guo. 2015c. Knowledge base completion using embeddings and rules. In *IJCAI*.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018b. R 3: Reinforced ranker-reader for open-domain question answering. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yanjie Wang, Rainer Gemulla, and Hui Li. 2018c. On multi-relational link prediction with bilinear models. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. 2022. Benchmarking generalization via in-context instructions on 1,600+ language tasks. *arXiv preprint arXiv:2204.07705*.

Zhen Wang, Jennifer Lee, Simon Lin, and Huan Sun. 2020. Rationalizing medical relation prediction from corpus-level statistics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8078–8092.

Zhen Wang, Rameswar Panda, Leonid Karlinsky, Rogerio Feris, Huan Sun, and Yoon Kim. 2023. Multitask prompt tuning enables parameter-efficient transfer learning. *arXiv preprint arXiv:2303.02861*.

Zhen Wang, Xiang Yue, Soheil Moosavinasab, Yungui Huang, Simon Lin, and Huan Sun. 2019b. Surfcon: Synonym discovery on privacy-aware clinical data. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 1578–1586.

Zhen Wang, Bo Zong, and Huan Sun. 2021c. Modeling context pair interaction for pairwise tasks on graphs. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 851–859.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. 2019. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641.

J. Weeds, D. Weir, and D. McCarthy. 2004. Characterising measures of lexical distributional similarity. In *COLING*.

Jason Wei, Maarten Bosma, Vincent Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. In *International Conference on Learning Representations*.

Yeming Wen, Dustin Tran, and Jimmy Ba. 2020. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*.

J. Wieting, M. Bansal, K. Gimpel, and K. Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. In *EMNLP*.

Adina Williams, Nikita Nangia, and Samuel R Bowman. 2017. A broad-coverage challenge corpus for sentence understanding through inference. *arXiv preprint arXiv:1704.05426*.

Terry Winograd. 1972. Understanding natural language. *Cognitive psychology*, 3(1):1–191.

Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. DeepPath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 564–573.

Wenhan Xiong, Xiang Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, and Barlas Oguz. 2021. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations*.

Wenhan Xiong, Mo Yu, Shiyu Chang, Xiaoxiao Guo, and William Yang Wang. 2019. Improving question answering over incomplete kbs with knowledge-aware reader. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4258–4264.

Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. 2020. Knowledge distillation meets self-supervision. In *European Conference on Computer Vision*, pages 588–604.

Kun Xu, Siva Reddy, Yansong Feng, Songfang Huang, and Dongyan Zhao. 2016. Question answering on Freebase via relation extraction and textual evidence. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2326–2336.

Bishan Yang, Scott Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding entities and relations for learning and inference in knowledge bases. In *Proceedings of the International Conference on Learning Representations (ICLR) 2015*.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with bertserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77.

Z. Yang, W. W. Cohen, and R. Salakhutdinov. 2016a. Revisiting semi-supervised learning with graph embeddings. In *ICML*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380.

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016b. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489.

Yuan Yao, Deming Ye, Peng Li, Xu Han, Yankai Lin, Zhenghao Liu, Zhiyuan Liu, Lixin Huang, Jie Zhou, and Maosong Sun. 2019. DocRED: A large-scale document-level relation extraction dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 764–777.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021a. Crossfit: A few-shot learning challenge for cross-task generalization in nlp. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7163–7189.

Xi Ye, Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, and Caiming Xiong. 2021b. Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. 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 (Volume 1: Long Papers)*, pages 1321–1331.

Zhitao Ying, Dylan Bourgeois, Jiaxuan You, Marinka Zitnik, and Jure Leskovec. 2019. Gnnexplainer: Generating explanations for graph neural networks. In *Advances in neural information processing systems*, pages 9244–9255.

Donghan Yu, Chenguang Zhu, Yiming Yang, and Michael Zeng. 2020. Jaket: Joint pre-training of knowledge graph and language understanding. *arXiv preprint arXiv:2010.00796*.

Wenhao Yu, Dan Iter, Shuohang Wang, Yichong Xu, Mingxuan Ju, Soumya Sanyal, Chenguang Zhu, Michael Zeng, and Meng Jiang. 2022. Generate rather than retrieve: Large language models are strong context generators. *arXiv preprint arXiv:2209.10063*.

Xiang Yue, Zhen Wang, Jingong Huang, Srinivasan Parthasarathy, Soheil Moosavinasab, Yungui Huang, Simon M Lin, Wen Zhang, Ping Zhang, and Huan Sun. 2019. Graph embedding on biomedical networks: methods, applications and evaluations. *Bioinformatics*, 36(4):1241–1251.

Omar Zaidan, Jason Eisner, and Christine Piatko. 2007. Using "annotator rationales" to improve machine learning for text categorization. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*, pages 260–267.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1–9.

Rowan Zellers, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. From recognition to cognition: Visual commonsense reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6720–6731.

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762.

Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao. 2014. Relation classification via convolutional deep neural network. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pages 2335–2344.

C. Zhang, Y. Li, N. Du, W. Fan, and P. S. Yu. 2018a. Synonymnet: Multi-context bilateral matching for entity synonyms. *arXiv preprint arXiv:1901.00056*.

Muhan Zhang and Yixin Chen. 2018. Link prediction based on graph neural networks. In *NeurIPS*.

Sheng Zhang, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme. 2018b. Record: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.

Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. Paws: Paraphrase adversaries from word scrambling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1298–1308.

Zhen Zhang, Hongxia Yang, Jiajun Bu, Sheng Zhou, Pinggang Yu, Jianwei Zhang, Martin Ester, and Can Wang. 2018c. Anrl: Attributed network representation learning via deep neural networks. In *IJCAI*.

Jie Zhao, Ziyu Guan, and Huan Sun. 2019. Riker: Mining rich keyword representations for interpretable product question answering. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, page 1389–1398.

Cheng Zheng, Bo Zong, Wei Cheng, Dongjin Song, Jingchao Ni, Wenchao Yu, Haifeng Chen, and Wei Wang. 2020. Robust graph representation learning via neural sparsification. In *ICLR*.

Qihuang Zhong, Liang Ding, Juhua Liu, Bo Du, and Dacheng Tao. 2022. Panda: Prompt transfer meets knowledge distillation for efficient model adaptation. *arXiv preprint arXiv:2208.10160*.

Ruiqi Zhong, Kristy Lee, Zheng Zhang, and Dan Klein. 2021. Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2856–2878.

Chunting Zhou, Jiatao Gu, and Graham Neubig. 2019. Understanding knowledge distillation in non-autoregressive machine translation. In *International Conference on Learning Representations*.

Peng Zhou, Wei Shi, Jun Tian, Zhenyu Qi, Bingchen Li, Hongwei Hao, and Bo Xu. 2016. Attention-based bidirectional long short-term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212.

Yang Zhou, Sixing Wu, Chao Jiang, Zijie Zhang, Dejing Dou, Ruoming Jin, and Pengwei Wang. 2018. Density-adaptive local edge representation learning with generative adversarial network multi-label edge classification. In *ICDM*.

Yunchang Zhu, Liang Pang, Yanyan Lan, Huawei Shen, and Xueqi Cheng. 2021. Adaptive information seeking for open-domain question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3615–3626.