Unsupervised Learning of Relational Entailment Graphs from Text

Mohammad Javad Hosseini

Doctor of Philosophy

Institute for Language, Cognition and Computation
School of Informatics
University of Edinburgh

2020

Abstract

Recognizing textual entailment and paraphrasing is critical to many core natural language processing applications including question answering and semantic parsing. The surface form of a sentence that answers a question such as *"Does Facebook own Instagram?"* frequently does not directly correspond to the form of the question, but is rather a paraphrase or an expression such as *"Facebook bought Instagram"*, that entails the answer. Relational entailments (e.g., *buys* entails *owns*) are crucial for bridging the gap between queries and text resources. In this thesis, we describe different unsupervised approaches to construct relational entailment graphs, with typed relations (e.g., *company buys company*) as nodes and entailment as directed edges. The entailment graphs provide an explainable resource for downstream tasks such as question answering; however, the existing methods suffer from noise and sparsity inherent to the data.

We extract predicate-argument structures from large multiple-source news corpora using a fast Combinatory Categorial Grammar parser. We compute entailment scores between relations based on the Distributional Inclusion Hypothesis which states that a word (relation) p entails another word (relation) q if and only if in any context that p can be used, q can be used in its place. The entailment scores are used to build local entailment graphs. We then build global entailment graphs by exploiting the dependencies between the entailment rules. Previous work has used transitivity constraints, but these constraints are intractable on large graphs. We instead propose a scalable method that learns globally consistent similarity scores based on new soft constraints that consider both the structures across typed entailment graphs and inside each graph. We show that our method significantly improves the entailment graphs.

Additionally, we show the duality of entailment graph induction with the task of link prediction. The link prediction task infers missing relations between entities in an incomplete knowledge graph and discovers new facts. We present a new method in which link prediction on the knowledge graph of assertions extracted from raw text is used to improve entailment graphs which are learned from the same text. The entailment graphs are in turn used to improve the link prediction task.

Finally, we define the contextual link prediction task that uses both the structure of the knowledge graph of assertions and their textual contexts. We fine-tune pretrained language models with an unsupervised contextual link prediction objective. We augment the existing assertions with novel predictions of our model and use them to build higher quality entailment graphs. Similarly, we show that the entailment graphs improve the contextual link prediction task.

Lay Summary

The recent advancement of technology has made it convenient to use personal computers, smartphones, and tablets. Nowadays, people often use their devices as the first contact point to fulfill their information needs. While only about a decade ago, we used to query computers using a combination of keywords, it is now possible for us to communicate with computers using natural human language thanks to the developments of the Natural Language Processing field. We expect computers to understand our questions and be able to communicate with us, especially because of the availability of personal assistants such as Amazon Alexa, Apple Siri, Google Assistant, and Microsoft Cortana.

Personal assistants are now very good at handling queries such as checking the weather or asking the birthplace of celebrities. They try to find the answers to many of our questions by searching the web. However, in many cases they struggle at providing a definite answer because there are many ways of asking the same question, while the answers to only a few of them are explicitly stated over the web. Therefore, we are usually left with snippets that might or might not contain the answer to our questions. For example, currently (September 2020), Google Assistant does not correctly answer the question *"Has Zidane played in a Champions League Final?"*; however, if we change the question to *"Has Zidane scored in a Champions League Final?"*, it can find a good answer.

In this thesis, we develop models that use raw textual data collected from the web, to learn *entailments* between natural language relations, e.g., knowing that when an *athlete scores in a sports event* entails that the *athlete has played in the sports event*. We process large amounts of text to extract consistent patterns of entailment between the relations. We form initial relational entailment graphs, with relations as nodes and entailment rules as edges. We model the dependencies between the entailment rules to improve the graphs. In addition, we develop neural network models to learn relation representations that can generalize to rules that could not be otherwise extracted. The relational entailment graphs can be used to improve question answering systems.

Acknowledgements

I would like to express my special appreciation and thanks to my supervisor, Mark Steedman, for his incredible support and guidance throughout my PhD journey. This thesis could not exist without his perspective, patience, and ambition. I am thankful to Mark for giving me the courage to work on difficult but fundamental research problems and helping me grow as a researcher. It was reassuring to work with Mark, knowing that he will help me in any possible way, both academically and personally. Mark was always available for meetings whenever I visited his office, and I have learned a great deal from our conversations. It has been a great pleasure working with Mark, and I am sure I will miss my time being his PhD student.

I would like to thank Shay Cohen for his generous help over the years. I got many interesting ideas from meetings with Shay that had a significant effect on this thesis. I was also fortunate to be able to talk to Mark Johnson over our Edinburgh-Sydney weekly Skype calls, despite the long time difference. I am thankful for Mark's guidance that improved my work in many ways.

I would like to thank my PhD examiners, Ivan Titov and Ido Dagan for their constructive feedback which helped improve the thesis content.

I am grateful to The Alan Turing Institute for funding my research and having me at the institute in my first year and frequently after that. The experiments in this thesis were made possible by Microsoft's donation of Azure credits to The Alan Turing Institute. I am thankful to Nate Chambers and Siva Reddy for helping me during my first years to get through the learning curve quickly and start doing novel research.

I have had numerous interesting conversations about research and otherwise with members of Mark's group during our Thursday group meetings and beyond. Thanks to Andrew, Elizabeth, Ida, John, Liane, Milos, Nick, Nikita, Sabine, Sander, Siva, and Thomas. Many thanks to the EdinburghNLP group for organizing great events and inviting amazing speakers. I was privileged to be a part of such a large and active NLP community.

Before my PhD studies in Edinburgh, I was a PhD student at the University of Washington (but moved to Edinburgh because of some U.S. visa issues, typical to Iranian students, that happened to my wife). I would like to thank my supervisors at UW, Su-In Lee, Hanna Hajishirzi, and Oren Etzioni, for their endless support during my time in Seattle, and for teaching me how to research in Machine Learning and NLP.

Thanks to the Persian community who made our life away from home much easier for me and my wife. Thanks to our lovely friends in Seattle, Abbas and Farnaz,

for always being there for us. Thanks to our Londoner friends Ali, Fauti, Mohsen, and Maryam, for the happy memories they made for us during the year we were in London and the later visits. Thanks to Erfan, Hadi, Hajar, Meisam, Mahshid, Mohammad, Ramin, and Mina for making the Informatics Forum and Edinburgh a much more enjoyable place for us.

I cannot thank my caring family enough for their unconditional love and support, especially during all these years that we have been away from home. Thanks to Maman (Zahra) and Baba (Mahmood), and my siblings, Mehdi, Atefeh, and Reza, for teaching me the core values of life, and encouraging me to believe in myself, and supporting me to go on this long journey, although it was extremely hard for them.

Most importantly, I express my deepest gratitude to my beloved and kind wife, Laleh. Thank you for making all this possible by accepting to go on this adventure with me. I know that you made huge sacrifices by that decision, and I will always be grateful. Thank you for tolerating my sometimes long and random working hours and spending sleepless nights with me. You have been the joy of my life all these years, and without doubt, this journey would have been meaningless if you were not by my side. Laleh, I dedicate this thesis to you!

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Mohammad Javad Hosseini*)

Table of Contents

Chapter 1

Introduction

Nowadays, search engines and personal assistants are the first places to look for the answer to most of our questions about the world's knowledge. The rapid and successful growth of natural language processing (NLP) techniques have made it possible to communicate with computers in a much easier way than a decade ago. We can now ask open-domain questions using human language rather than trying to retrieve the answers by providing a combination of keywords. This capability is especially expected because of the use of commercial personal assistants such as Google Assistant, Amazon Alexa, Apple Siri, and Microsoft Cortana where users can interact by talking to a device. If we ask Google Assistant *"Has Facebook bought Instagram?"*[1], it retrieves the correct answer with the relevant words boldfaced.

Facebook bought Instagram *in 2012 for $1bn (£760m), and WhatsApp in 2014 for $19bn.*

Similarly, if we ask *"Has Zidane scored in a Champions League Final?"*, we again get an acceptable answer:

18 years ago today, ***Zinedine Zidane scored*** *that volley in the* ***Champoins Leaguge Final****. On 15 May 2002, Real Madrid and Bayern Leverkusen played out ...*

The answer is usually out there somewhere on the web, but in most cases, the surface form of the answer does not exactly correspond to the form of the question, but is rather a *paraphrase* or an expression that *entails* the answer. This is because there are too many ways of asking the same question and only a small fraction of those

[1]These experiments were done on 18th September 2020.

forms are covered by the text over the web. For example, if we ask the first question in the form *"Does Facebook own Instagram?"*, we get the same answer:

***Facebook** bought **Instagram** in 2012 for $1bn (£760m), and WhatsApp in 2014 for $19bn.*

However, this time the word *bought* is not boldfaced anymore perhaps because the systems do not have an understanding that *buy* entails *own*. But if we change the second question to *"Has Zidane played in a Champions League Final?"*, we get a completely irrelevant answer:

*On 4 May, **Zidane** led Real Mardird to a place in the 2016 UEFA **Champions League Final** by beating Manchester City ...*

The answer could be easily retrieved by knowing that when a player *scores in* a sports event, it entails that the player has also *played in* that sports event. Some of these results are mainly retrieved because they contain both entities in the question, although the *relation* between the entities could be different than the one asked in the question. In fact, if we ask about a specific relation between two arbitrary entities, in many cases we are left with a possibly irrelevant snippet that contains the name of the two entities. For example, Google Assistant answers the question *"Does IBM own HP?"* as:

*What **does** power really mean to women? Like **IBM**, **HP** saw years ago that the future of big tech was not in ...*

As a result, despite all the recent success, the current technology cannot still find a definite answer to most factual questions. The lack of a well-established form-independent semantic representation for natural language is the most important single obstacle to bridging the gap between queries and text resources. This thesis seeks to learn meaning postulates such as *buying* entails *owning* that can be used to augment the standard form-dependent semantics. Our goal is to learn entailment rules between typed relations (e.g., *company buys company*) with two entities, where the type of each relation is determined by the types of its entities. We construct *relational entailment graphs*, with relations as nodes and entailment rules as edges.

Unlike pure distributional and neural-network based methods, entailment graphs are easy to interpret since they encode entailment between relations explicitly. They

provide an explainable resource that can be used to improve downstream tasks such as semantic parsing and question answering. In the above examples, it is mentioned in the text that *Facebook bought Instagram* and *Zidane scored in the Champions League Final.* Therefore, if we have access to the explicit knowledge that *buys* entails *owns*, and *scores in* entails *plays in*, we will not only be able to answer the two questions *"Does Facebook own Instagram?"* and *"Has Zidane played in a Champions League Final?"*, but also can explain why the system has extracted the answers.[2] The entailment graphs are easy to interpret even for non-experts; therefore, while we learn them automatically from text, it is possible to inspect the graphs for manual extension or removal of the existing errors. This is especially important since we can remove spurious correlations and biases that exist in the textual data used for building the graphs.

A naive approach to predict entailment rules is to use the distributional representations (embeddings) of relation tokens. However, the existing embeddings do not accurately capture entailments because they are not optimized directly for the task of entailment detection. Instead, entailment rules can be detected by computing entailment scores between typed relations extracted by machine reading over large amounts of text. The entailment scores are computed based on the Distributional Inclusion Hypothesis (DIH), which states that a word (relation) p entails another word (relation) q if and only if in any context that p can be used, q can be used in its place (Dagan et al., 1999; Geffet and Dagan, 2005). The scores are computed on a set of predicate-argument structures in the form of *facts* or *triples* such as *(bought, Facebook, Instagram)*, where the context of each relation is its set of entity pairs.

Unfortunately, even if we could process all the text over the web, many correct triples could not be still extracted because they are not directly stated in the text. In addition, the DIH could lead to noisy entailment decisions. Therefore, the entailment graphs suffer from the sparsity and the noise inherent to the data. This thesis investigates multiple unsupervised approaches to learn high quality entailment graphs. Our methods add correct missing edges and remove spuriously identified ones by both considering the dependencies between the rules and predicting missing facts in the corpus.

Our models are completely unsupervised, i.e., they only uses raw text as their input and do not depend on any annotated entailment dataset. Therefore, we can benefit from the huge amount of textual data available over the web. In addition, the entailment graphs do not suffer from potential drawbacks of entailment datasets. Finally, while

[2] In our search example, one simple way of explanation is to boldface the relevant tokens such as *bought*.

we have performed our experiments on English text, our methods can be used to learn entailment graphs for any other language as long as we have access to textual data as well as NLP tools to extract triples from text and assign types to the entities.

Thesis Statement. Relational entailment graphs explicitly encode entailment between natural language relations and provide an explainable resource that is necessary for downstream tasks such as open-domain factual question answering. In this thesis, we describe different unsupervised approaches to learn high-quality relational entailment graphs from text. We show that modeling the dependencies across different typed entailment graphs and inside each graph significantly improves the entailment decisions. Furthermore, we show the duality between learning entailment graphs and the link prediction task. Finally, we propose contextual link prediction and show that fine-tuning contextual representations of relations is an effective approach for learning relational entailments.

1.1 Thesis Contributions

The main contributions of the thesis are summarized below.

Relational Entailment Graph Framework. We have built a robust pipeline that processes large text corpora and builds entailment graphs. The pipeline first applies a fast supervised trained Combinatory Categorial Grammar (CCG; Steedman, 2000) parser to text and extracts triples. It links the entities to a knowledge base and assigns fine-grained types to them. It then builds large entailment graphs with tens of thousands of nodes corresponding to English predicates.

Global Soft Constraints. Previous work has considered global transitivity constraints on the edges of entailment graphs, but these constraints are intractable on large graphs. We instead propose two sets of global soft constraints. First, the cross graph constraints that learn when an entailment from one typed entailment graph can be transferred to another typed entailment graph. While the transfer adds missing edges to the graphs, it can only be applied for some relations and graph types. We jointly learn transfer coefficient that capture this property as well as the entailment scores. Second, the paraphrase resolution constraints that encourage relations that are paraphrase of each other to have the same patterns of entailments. Our empirical evaluation demonstrates that the global soft constraints significantly outperform previous state-of-the-art entailment graphs on standard entailment datasets. We also show that entailment graphs improve performance on a downstream question answering task. Our approach

can explain the answers that are retrieved based on the entailment graphs.

Duality of the Entailment Graph Building and Link Prediction Tasks. Similar to the entailment graph building task, the link prediction for knowledge graph completion operates on a set of triples. These methods are usually applied to existing manually annotated knowledge graphs in order to predict the missing relations (links) between the entities, but they can also be applied to extractions from text. The link prediction methods learn low-dimensional dense representations (embeddings) for relations and entities and are known to implicitly capture relational entailments. However, there is no straightforward method to find entailments from the embeddings. We present a method to learn relational entailment graphs based on the link prediction scores assigned to both observed and predicted triples. We show that using the novel triples in building the entailment graphs alleviate the sparsity issue. Moreover, we show that the entailment graphs can in turn be used to improve the link prediction task showing the two tasks are indeed complementary.

Contextual Link Prediction to Learn Entailment Graphs. The link prediction methods only use the structure of the knowledge graph extracted from text, but they ignore the actual contexts of the extractions. We introduce the contextual link prediction task, where given a triple in context, the goal is to predict all the relations that hold between the entities in the triple. We propose a model that fine-tunes existing pre-trained language models with an unsupervised objective function. We then use the observed and predicted triples to build entailment graphs. We show that the predicted triples from the contextual link prediction lead to more accurate entailment graphs. In addition, we show that the entailment graphs improve the contextual link prediction task.

1.2 Thesis Outline

The thesis is structured as follows:

Chapter 2. We review the relevant literature on a) entailment graph building and its application to semantic parsing; b) the link prediction models; and c) methods for learning contextual representations of relations and entities.

Chapter 3. We discuss the pipeline that we have used to build the initial entailment graphs. We introduce a scalable method for learning globally consistent entailment graphs based on two sets of global soft constraints. We evaluate the learned entailment graphs on standard entailment datasets and compare them to the previous state-of-the-

art entailment graphs. We also show that the entailment graphs improve performance on a downstream task. This chapter is based on Hosseini et al. (2018).

Chapter 4. We investigate the connection between the entailment graph induction and the link prediction tasks. We show that the two tasks benefit from each other, although they operate over the same set of triples. This chapter is based on Hosseini et al. (2019).

Chapter 5. We introduce the contextual link prediction task and propose a model for the task. We learn entailment graphs using the predictions of our model and show improvements over the results in the previous chapters. This chapter is based on Hosseini et al. (2021).

Chapter 6. We conclude by summarizing our methods and discussing possible future work directions.

Chapter 2

Background

In this chapter, we provide the necessary background for the remainder of the thesis. We first review the relevant literature on relational entailment graphs in section 2.1. We then summarize the link prediction task for knowledge graphs and its existing solutions in section 2.2. We finally review some of the recent work on contextual representation learning for relations and entities in section 2.3.

2.1 Relational Entailment Graphs

In Recognizing Textual Entailment (RTE; Dagan et al., 2013), or natural language inference, a model is presented with a pair of text fragments (e.g., sentences), and is asked to decide whether the second text fragment can be reasonably *inferred* given the first one, i.e., whether one can say that the second text fragment holds given that we know the first one is correct (Dagan et al., 2005; Bos and Markert, 2005; MacCartney and Manning, 2009; Dagan et al., 2013).[1] An important subtask of RTE is to find entailments between natural language predicates (relations)[2]. Given two relations, the aim is to predict whether the first relation entails the second one. In other words, when we instantiate the relations over the same set of entities, we answer whether the second relation can be *inferred* if we know that the first relation holds between the entities. A relational entailment graph[3] is a directed graph $G = (R, E)$, with the set of relations R as nodes and the set of entailment rules[4] $E \subseteq R \times R$ as edges (Berant et al., 2010,

[1] In this thesis, we consider a binary decision problem, i.e., *entailment* or *not entailment*. In some works, a three-way classification is used, i.e., *entailment*, *contradiction* or *neutral*.

[2] We use relations and predicates interchangeably.

[3] We use the term "entailment graph" as an equivalent of "relataional entailment graph".

[4] Also known as inference rules.

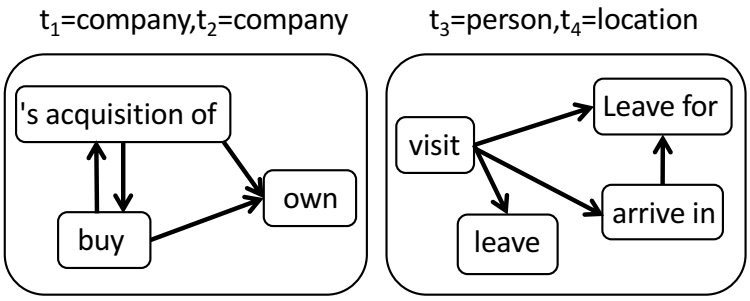


Figure 2.1: Examples of typed entailment graphs for entities of types *company,company* and *person,location*.

2011). An edge (p,q) from a relation $p \in R$ to a relation $q \in R$ means that p entails q, denoted by $p \rightarrow q$. If the entailment holds in both directions, the two relations are considered to be a paraphrase of each other. In fact, any clique in the entailment graph (a subgraph with edges between all pairs) is a paraphrase cluster.

We learn entailment rules between binary relations, i.e., relations between two entities. The entailment rules often depend on the context, e.g., when a *person visits a location*, it has different entailments from when a *person visits another person*. We follow the previous work and learn *typed* entailment graphs, where the type of relations are determined by the type of entities (Schoenmackers et al., 2010; Berant et al., 2011; Lewis, 2014). While entailments and paraphrasing can benefit from n-ary relations, e.g., *person visits a location in a time*, in this thesis, we confine our attention to binary relations and leave the construction of n-ary graphs to future work. We construct a typed entailment graph for any plausible type pair. Figure 2.1 shows small fragments of the typed entailment graphs with entities of types *company,company* and *person,location*.

Earlier attempts to learn relational entailments take a *local* learning approach. They learn a local entailment graph by predicting entailment or paraphrase rules independently from each other (Lin, 1998; Lin and Pantel, 2001; Szpektor et al., 2004; Bhagat et al., 2007; Szpektor and Dagan, 2008; Yates and Etzioni, 2009; Schoenmackers et al., 2010). Berant et al. (2010, 2011, 2012, 2015) propose a *global* learning approach. They learn global entailment graphs by taking the dependencies between the entailment rules into account. We discuss a number of local entailment scores in section 2.1.1 and the existing methods for learning global entailment graphs in section 2.1.2.

2.1.1 Local Learning of Entailment Graphs

In the local learning approach, the entailment rules are predicted by computing a local entailment score $w^0_{pq} \in [0,1]$ from the relation p to the relation q. The entailment rule holds when the entailment score is greater than or equal to a pre-defined threshold. The most common approach is to compute distributional entailment scores by processing a large text corpus. Symmetric (bi-directional) scores measure the amount of overlap between the contexts of two relations and can predict paraphrase relations. Directional scores are computed based on the Distributional Inclusion Hypothesis (DIH). The scores based on the DIH measure how much the context of one relation is included in the context of another relation and are suitable for predicting entailment.

In this section, we describe a number of distributional entailment scores. Let $F(p)$ denote a set of features f for a relation p that are extracted by processing the input text corpus. The feature representation can be binary, i.e., entity-pairs (e_1, e_2) that appear in a binary relation, or triple, (p, e_1, e_2) in the corpus (Szpektor et al., 2004; Yates and Etzioni, 2009; Schoenmackers et al., 2010). For example, *Facebook-Instagram* is a feature for the relation *buy*. Some work has also used unary feature representation, i.e., single entities that appear in the first or second slot of the relations (Lin and Pantel, 2001; Szpektor and Dagan, 2008). They define scores by combining the scores of the two slots. Let N be the number of all triples extracted from text, and $N(p)$ and $N(f)$ be the number of occurrences of the relation p or the feature f in the corpus. Let $N(p,f)$ be the number of occurrences of the feature f for the relation p. In the case of entity-pair features $f = (e_1, e_2)$, $N(p,f)$ denotes the number of occurrences of the triple (p, e_1, e_2).

The value $v(p,f)$ of the feature f is the frequency of its occurrence $N(p,f)$ or the pointwise mutual information (PMI) between the relation and the feature:

$$\text{PMI}(p,f) = \log_2 \frac{\Pr(p,f)}{\Pr(p)\Pr(f)},$$

where $\Pr(p) = N(p)/N$, $\Pr(f) = N(f)/N$, and $\Pr(p,f) = N(p,f)/N$.

Lin similarity (Lin, 1998) between relations p and q is defined as:

$$\text{Lin}(p,q) = \frac{\sum_{f \in F(p) \cap F(q)} v(p,f) + v(q,f)}{\sum_{f \in F(p)} v(p,f) + \sum_{f \in F(q)} v(q,f)}.$$

Weeds and Weir (2003) define Weeds precision[5] and Weeds recall:

[5]The Cover score defined by Szpektor and Dagan (2008) is equivalent to Weeds precision.

$$\text{Weeds\_Precision}(p,q) = \frac{\sum_{f \in F(p) \cap F(q)} v(p,f)}{\sum_{f \in F(p)} v(p,f)},$$
$$\text{Weeds\_Recall(p,q)} = \frac{\sum_{f \in F(p) \cap F(q)} v(q,f)}{\sum_{f \in F(q)} v(q,f)}.$$

Weeds similarity is the geometric average of Weeds precision and Weeds recall. Finally, the Balanced Inclusion (BInc; Szpektor and Dagan, 2008) score is the geometric average of Weeds precision and Lin similarity. The Weeds and Lin similarities are symmetric, but Weeds precision and recall, and BInc are directional.

A local entailment graph is defined as $G^0 = \{R, E^0_\delta\}$, where the relations R are the nodes and $E^0_\delta = \{(p,q) | p,q \in R, w^0_{pq} \geq \delta\}$ are the edges. In our experiments in chapter 3, we compute Lin, Weeds and Balanced Inclusion similarities as our local scores based on the PMI values. We choose to use binary feature representations since they performed similar to unary feature representations in our preliminary experiments and they only keep one vector, instead of two vectors of the unary case.

2.1.2 Global Learning of Entailment Graphs

The local learning approach ignores the dependencies between the entailment rules. Modeling the dependencies by applying constraints during learning of the entailment graphs improves the entailment decisions by removing wrong edges and adding correct missing edges. Berant et al. (2010, 2011) propose applying transitivity constraints, since entailment is a transitive property. If we know that $p \rightarrow q$ and $q \rightarrow r$, the entailment $p \rightarrow r$ must hold automatically. Figure 2.2 shows an example, where knowing that *person serve as president of country* $\rightarrow$ *person be elected president of country* and that *person be elected president of country* $\rightarrow$ *person run for presidency of country*, we can infer that *person serve as president of country* $\rightarrow$ *person run for presidency of country*.

2.1.2.1 Exact Integer Linear Programming Solution

Berant et al. (2010, 2011) compute a number of local similarity scores such as Lin and BInc score as $\mathbf{w}^0_{pq} \in [0,1]^l$, where l is the number of similarity scores. They define a scoring function $g : R \times R \rightarrow \mathbb{R}$ based on the local entailment scores. The function g is defined in a way that higher values indicate higher entailment probabilities. They then

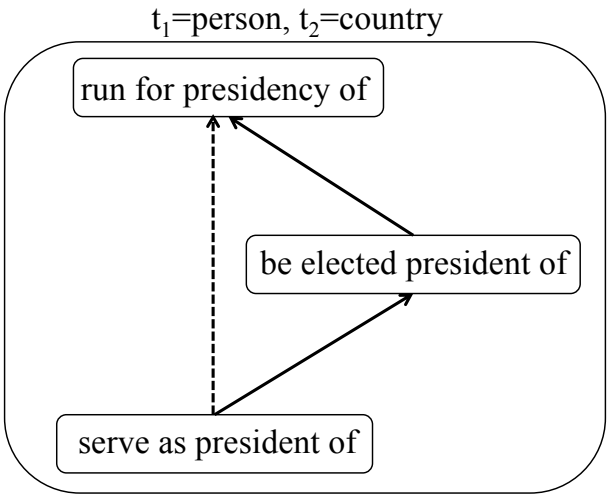


Figure 2.2: A fragment of the entailment graph with entities of types *person,country*. The solid edges are identified with local entailment scores, but the dotted edges can be added using transitivity constraints.

find a globally optimized graph $G = \{R, E\}$, that preserves the transitivity constraints and maximizes the sum of edges weights $\sum_{p,q \in E} g(p,q)$. They show that this problem is NP-hard by a reduction from the NP-hard Transitive Subgraph problem (Yannakakis, 1978).

They define a random variable $X_{pq} \in \{0,1\}$, where $X_{pq} = 1$, if $p \rightarrow q$, and $X_{pq} = 0$, otherwise. Let $X \in \{0,1\}^{R \times R}$ denote the set of all random variables X_{pq}. In order to find the optimal graph, they optimize the following Integer Linear Programming (ILP):

$$\hat{X} = \text{argmax} \sum_{p \neq q} g(p,q) X_{pq} \tag{2.1}$$

$$s.t. \quad \forall_{p,q,r \in R} \; X_{pq} + X_{qr} - X_{pr} \leq 1 \tag{2.2}$$

$$\forall_{(p,q) \in A_{\text{yes}}} \; X_{pq} = 1 \tag{2.3}$$

$$\forall_{(p,q) \in A_{\text{no}}} \; X_{pq} = 0 \tag{2.4}$$

$$\forall_{p \neq q} \; X_{pq} \in \{0,1\}.$$

Equation 2.1 is the objective function of the ILP which is the summation over the scoring functions of the final edges. The constraints in Equation 2.2 ensure that the transitivity holds. The constraints in equations 2.3 and 2.4 state that certain edges should be present or not based on background knowledge. The edges for A_{yes} have been constructed by syntactic rules. They normalize each predicate by omitting the first

word if it is a modal and turning passives into actives. If two normalized predicates are equal, they will be connected. A_{no} has been also constructed from different sources: a) predicates differing by a single pair of words that are antonyms in WordNet. b) predicates differing by a single word of negation and c) a transitive predicate p with types (t_1, t_2) and the same predicate p with the types in the opposite order, i.e., (t_2, t_1). After solving for $\hat{X}$, the optimal graph $\hat{G} = (R, \hat{E})$ is formed by having $p, q \in \hat{E}$ if and only if $X_{pq} = 1$.

In order to compute the function g, they first generate positive and negative relational entailments using WordNet (Miller, 1995). Positive examples are hyponym-hypernym pairs as well as synonym pairs, and negative examples are co-hyponyms, hyponyms at distance 2, and random pairs. They train a classifier to predict $X_{pq} = 1$, if $p \rightarrow q$, and $X_{pq} = 0$, otherwise. They then define the scoring function g as:

$$g(p,q) = \frac{\Pr(X_{pq} = 1 | \mathbf{w}^0_{pq})}{1 - \Pr(X_{pq} = 1 | \mathbf{w}^0_{pq})} + \log \eta, \tag{2.5}$$

where $\Pr(X_{pq} = 1 | \mathbf{w}^0_{pq})$ is the posterior probability of the entailment rule $p \rightarrow q$. The constant $\eta = \frac{\Pr(X_{pq}=1)}{\Pr(X_{pq}=0)}$, i.e., the prior odd ratio of an edge in the graph. They show that defining g as in Equation 2.5 leads to a graph with highest posterior probability given some independence assumptions between the edges. Because entailment graphs are sparse, it is reasonable to assume that $\eta \leq 1$, therefore, $\log \eta \leq 0$. We define $\lambda = -\log \eta \geq 0$ as a hyper-parameter that controls the sparsity of the entailment graphs. Larger values of λ yield sparser graphs.

In this thesis, we learn the entailment graphs in an unsupervised manner and do not use external resources. In the experiments with transitivity constraints in chapter 3, instead of estimating entailment posterior probability using a classifier, we simply use a single local entailment score w^0_{pq} to have:

$$g(p,q) = \frac{w^0_{pq}}{1 - w^0_{pq}} - \lambda.$$

Our preliminary experiments showed very similar results when we used $g'(p,q) = w^0_{pq} - \lambda$. We therefore used $g'(p,q)$ for its simplicity.

ILP is an NP-complete problem and the ILP solvers do not scale well. Berant et al. (2011) proposed two scaling techniques that exploit the sparsity of entailment graphs. First, they showed that if we can partition the set of nodes R into disjoint sets R_1 and R_2 such that for any crossing edge (p,q) between them (in both directions), $g(p,q) < 0$,

then the optimal edges $\hat{E}$ do not contain any crossing edges[6]. If we know that the original graph can be decomposed into smaller components so that there are no edges between the components, we can run the ILP solver on each component and scale up the algorithm. This can be done by finding the strongly connected components (SCC) of an undirected graph $G^U = (R, E^U)$ with edges $E^U = \{(p,q) : g(p,q) \geq 0 \ \vee g(q,p) \geq 0\}$. Second, they have used incremental ILP (Riedel and Clarke, 2006) following the intuition that even if we do not specify all the transitivity constraints in the ILP formulation, most of them will be automatically satisfied given a good local score. They define an active set of constraints. The active set is initially empty and is filled by iteratively running the ILP solver and adding the transitivity constraints whenever they are violated. The algorithm is guaranteed to converge because size of the active set increases in each iteration, but the total number of constraints cannot grow more than $|R|^3$, $|R|$ is the number of nodes in the graph. In practice, the incremental ILP reaches conversion in around at most 6 iterations (Berant et al., 2011).

The experiments in Berant et al. (2010) have not scaled to more than 50 nodes in a reasonable time[7], but the experiments in Berant et al. (2011) have scale to around 120 nodes using the above techniques. Nevertheless, the number of natural language predicates is far more which are beyond the capability of ILP solvers. In this thesis, we consider graphs of up to size 53K relations (section 3.5.1).

2.1.2.2 Approximate Solution

Berant et al. (2012, 2015) propose an approximate solution to the problem. They note that the entailment rules are typically from more specific relations (child nodes) to more general ones (parent nodes). Therefore, one could expect entailment graphs to have tree-like structures. A strongly connected component of a directed graph G is a subset of nodes where there is a path from any node to any other node. Any directed graph G can be converted into a strongly connected component (SCC) graph by the following procedure: 1) finding the SCCs; 2) collapsing all the nodes in any SCC to one node; 3) adding an edge from an SCC S_1 to another SCC S_2 if there is an edge from any node is S_1 to any node in S_2. If G is a transitive graph, the SCC graph will also be a transitive graph (Berant et al., 2012). In addition, any SCC will be a clique, i.e., all nodes have directed edges to each other. In entailment graphs, an SCC can be interpreted as a paraphrase cluster, where all the relations are paraphrase of each

[6] We note that it should also be assumed that A_{yes} will not have any edges between the disjoint sets.

[7] 2 hours to 24 hours.

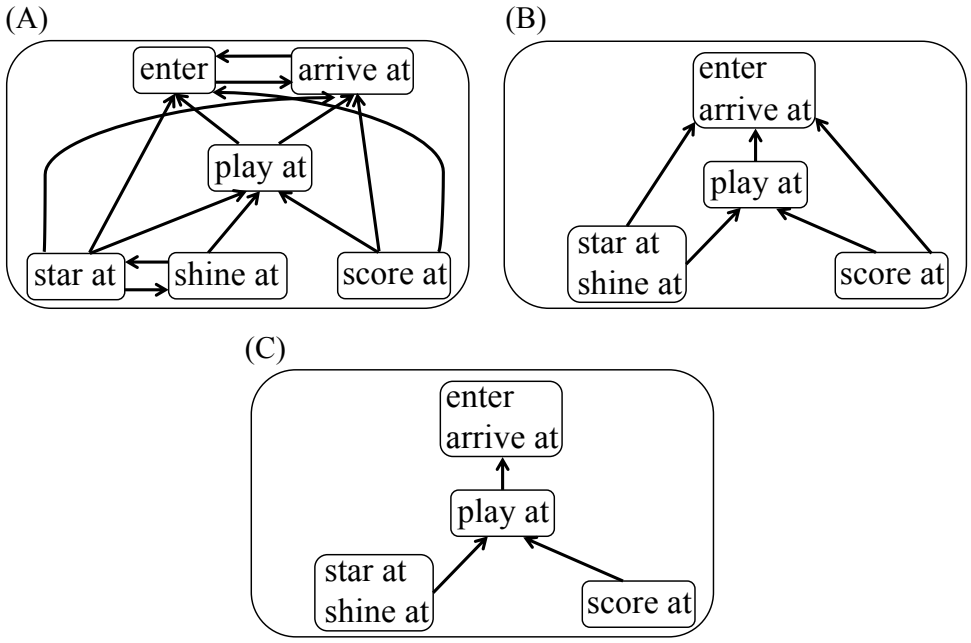


Figure 2.3: (A) A fragment of the entailment graph with entities of types *sports player,sports facility*; (B) its equivalent strongly connected component graph; and (C) its equivalent reduced graph.

other. Figure 2.3A shows an entailment graph with entities of types *sports player,sports facility*. Figure 2.3B shows the SCC graph of the entailment graph.

The *transitive closure* of a graph is obtained by adding an edge from any node p to any node q if there is any path from p to q. The *transitive reduction* of a graph is obtained by removing all edges that can be obtained by performing transitive closure on the graph. Berant et al. (2012) define the *reduced graph* of a directed graph G as the transitive reduction of the SCC graph of G. Figure 2.3C shows an example reduced graph. A *directed forest* is a directed acyclic graph (DAG) where each node has at most one parent.[8] Berant et al. (2012, 2015) hypothesize that the reduced graph of an entailment graph is a directed forest. They define a forest-reducible graph (FRG) as a graph that its reduced version is a directed forest. They test the FRG assumption on sample entailment graphs and observe that the assumption is correct for most of the edges.

Berant et al. (2012, 2015) propose an approximation algorithm, Tree-Node-Fix (TNF), that is based on the FRG assumption. They first construct an initial FRG and iteratively remove a node from the graph and attach it back with a set of edges with

[8]In some other literature, the definition of the directed forest is a DAG with a forest as underlying undirected graph, where a forest is a graph in which any two nodes are connected with at most one path.

higher sum of weights. The algorithm continues until convergence. They show that the approximation algorithm gets results close to the ILP on a relational entailment dataset and scales up to 20K nodes. They also show that a simple greedy algorithm, High to Low Forest-Reducible Graph (HTL-FRG), gets very similar results to the TNF algorithm and is much faster. HTL-FRG is an iterative algorithm that sorts the edges from high to low scores according to the scoring function. The algorithm starts with an empty graph and in each iteration, it adds a new edge, and the edges obtained by its transitive closure subject to two constraints: 1) the summation of the scores of all the new edges are positive and 2) the FRG assumption is not violated. Otherwise, none of the new edges will be added.

The FRG assumption ensures that a relation cannot entail two different relations unless at least one of those relations entails the other. While the FRG assumption leads to improved scalability, it is not correct for many real world entailments. For example, in the entailment graph with *person,location* types (Figure 2.1 right), *visit* $\rightarrow$ *arrive in* and *visit* $\rightarrow$ *leave*, but there is no entailment rule between *arrive in* and *leave* in any direction. In addition, both the TNF and HTL-FRG methods are greedy algorithms and are not guaranteed to converge to optimal graphs because early decisions in the procedure might prevent the methods from selecting optimal edges in the later steps.

In chapter 3, we propose new soft constraints that are not based on the FRG assumption and scale to large graphs.

2.1.3 Using Entailment Graphs to Obtain Form-Independent Semantics

Talking to a computing device about every-day needs (Artzi and Zettlemoyer, 2011; Kollar et al., 2018), teaching a computer to perform an action (Artzi and Zettlemoyer, 2013), or querying a knowledge base with natural language (Zettlemoyer and Collins, 2005) require communicating with machines in a language interpretable by them. *Semantic parsing* is the task of mapping natural language to meaning representations (logical forms) that are machine-interpretable.

Formal logical semantics can capture the meaning of sentences including *function words* such as *and*, *not*, and *every*. It is possible to model a variety of semantic phenomena including negation, quantification, composition, tense, and aspect using logical semantic representations such as first-order logic augmented with *lambda calculus* (Lewis and Steedman, 2013a; Kamath and Das, 2019). Logical forms are built

by first assigning an interpretation to each word in the sentence, and then combining the interpretations of the words to form the meaning of the sentences. Interpretations of *content words* can be generated automatically, by using the word itself as a symbol in the logical form (Lewis, 2014). Formal logical semantics is capable of performing multi-sentence inferences, e.g., by the use of theorem provers. However, standard approaches like Bos (2008) show low recall on applications such as entailment (Bos and Markert, 2005) or question answering. This is because they cannot generalize beyond the exact lexical form of the given sentences.

In this thesis, we learn entailment graphs between relations extracted based on an initial semantics (section 2.1.4). The entailment graphs can be directly used to obtain a form-independent semantics that can significantly improve the applicability of pure logical semantics on downstream tasks (section 2.1.5).

2.1.4 Combinatory Categorial Grammar

Our work is based on Combinatory Categorial Grammar (CCG; Steedman (2000)), a strongly lexicalized linguistic formalism. In particular, we use CCG to extract binary relations that are used to form the entailment graphs. In CCG, the syntax and semantics assigned at the word level dictate the syntax and semantics at the phrase and sentence levels. In addition, CCG syntax is highly transparent to its semantics, i.e., the syntactic properties of words and sentences, directly determine their semantics (Steedman and Baldridge, 2011). Therefore, it is possible to map the parser output into logical forms that capture predicate-argument structures (Bos, 2008). CCG can model a wide range of language phenomena and is known to capture long-range dependencies (Clark et al., 2002).

During CCG parsing, each word is assigned a *lexical entry*. The lexical entry maps a word to its *syntactic category* and *semantic interpretation*. Each word in the lexicon can have one or more lexical entries. For example, the lexical entries for the words *Facebook*, *bought* and *Instagram* include:

(1) Facebook := $NP : facebook$

(2) bought := $(S\backslash NP)/NP : \lambda y \lambda x \lambda e.bought(x, y, e)$

(3) Instagram := $NP : instagram$

The first lexical entry means that *Facebook* can be a noun phrase (syntactic category), with the symbol *facebook* (semantic interpretation). The categories can be either

primitive categories X such as N (noun), NP (noun phrase), PP (prepositional phrase), and S (sentence), or functional categories (Steedman and Baldridge, 2011). The functional categories are functions between other categories. The categories X/Y and $X\backslash Y$ are functions that consume the argument Y and return the category X. The Forward slash means that the argument is on the right-hand side and the backward slash means that it is on the left-hand side. The syntactic category $(S\backslash NP)/NP$ is a function that consumes NP (*Instagram*) on its right-hand side and returns the syntactic category $S\backslash NP$ (the phrase *bought Instagram*). The category $S\backslash NP$ returns S (*Facebook bought Instagram*) by consuming NP (*Facebook*) on its left-hand side.

Each lexical entry also provides a semantic interpretation. We have used lambda calculus and first-order logic to show the semantic representations. For example, the representation of *bought* is $\lambda y\lambda x\lambda e.bought(x,y,e)$, which is a function from two entities to a predicate on events. The variable e is a Davidsonian event variable (Davidson, 1967) and the variables x and y are two entities in the semantics corresponding to the two noun phrases in the syntactic category. The first noun phrase to be consumed is y (*instagram*) and the second one is x (*facebook*).

In contrast to Context Free Grammars (CFG), CCG contains only a handful of *combinators* (rules) and most of the information is in the parse tree nodes rather than the rules combining them. The combinators combine adjacent syntactic categories. For example, the forward application rule combines X/Y and Y into Y.[9] A CCG derivation combines categories such that the final expression has a single category (usually a sentence S). Derivation 4 shows how CCG maps the sentence *Facebook bought Instagram* to the binary relation $\lambda e.bought(facebook, instagram, e)$.

(4)

Facebook	bought	Instagram
NP	$(S\backslash NP)/NP$	NP
facebook	$\lambda y\lambda x\lambda e.bought(x,y,e)$	*instagram*
	$S\backslash NP$ (>)	
	$\lambda x\lambda e.bought(x, instagram, e)$	
	S (<)	
	$\lambda e.bought(facebook, instagram, e)$	

[9] See Steedman and Baldridge (2011) for more details.

2.1.5 Relational Entailment Learning to Improve Logical Semantics

Unfortunately, standard formal semantic approaches that use CCG or other semantic representations show low recall on downstream tasks. This is because there are too many ways of stating the same information, and it is not clear which one of them will be used when asking and answering questions (Lewis, 2014). For example, to answer the question *Does Facebook own Instagram?*, we might have access to any of the following propositions.

1. Facebook purchased Instagram.
2. Facebook has bought Instagram.
3. Facebook tried to acquire Instagram.
4. Facebook's acquisition of Instagram.
5. Facebook did not buy any company.
6. Facebook does not plan to sell Instagram.
7. Instagram is not a part of Facebook.

Some of the propositions (1, 2, 4, and 6) *entail* that the answer to the question is *yes*, some others (5 and 7) entail that the answer is *no*, and some do not give a definite answer (3). Existing hand-built ontologies like WordNet (Miller, 1995) contain some of the above information, but those resources suffer from low recall themselves and cannot be used to handle all the variabilities of language.

Distributional semantics offer a solution to the above problem by learning paraphrase and entailment rules between the content words, in this case natural language predicates. In distributional semantics, the meaning of words are modeled based on their usage in large corpora rather than hand-built annotations of training data; hence, it can be used to capture the huge variety of meanings in natural language. Words are represented as vectors learned from collocations in text. This kind of semantics has been effective in learning content words. The vectors can be composed in different ways to represent longer utterances. However, it is not clear how it can be used to model the meaning of function words indicating semantic operators such as negation

or quantification, and their current ability to capture the long-range dependencies in sentences is limited (Lewis, 2014).

An attractive way to gain advantage from both logical and distributional semantics is to combine them into a single model. Lewis and Steedman (2013a) introduce a method to map natural language to first-order logical representation using CCG. Their representation can capture function words such as *not* and *every*, and also uses distributional semantics to capture the meaning of content words. Instead of having non-logical symbols for relations in the logical representation, they use arbitrary cluster identifiers. For example, the relations *buy*, *purchase* and *'s acquisition of* get mapped to a cluster identifier such as *relation41* and the relations *own* and *be owner of* get mapped to *relation52*. The clustering is done at the level of predicate-argument structure and the model is capable of performing paraphrasing. Lewis and Steedman (2013a) suggest to change the CCG lexicon using the cluster identifiers. For example, we will have the same semantic interpretations for *buy* and *purchase*:

(5) buy := $(S\backslash NP)/NP : \lambda y \lambda x \lambda e.rel41(x, y, e)$

(6) purchase := $(S\backslash NP)/NP : \lambda y \lambda x \lambda e.rel41(x, y, e)$

Entailment graphs serve as a rich model of lexical semantics. They not only identify paraphrase clusters that correspond to their strongly connected components (section 2.1.2.2), but also specify entailment rules between the paraphrase clusters (Lewis, 2014). For example, The entailment rule between *relation41* to *relation52* can be used to answer *yes* to the above question given propositions 1, 2, and 4.

We learn entailment graphs by processing large text corpora and extracting binary relations using an initial CCG parser. Our learned entailment graphs will be used to redefine the lexicon as explained above. The text will be then re-parsed with the new semantics. The new extractions immediately support question-answering involving paraphrasing and entailment. In addition, we can answer questions by looking into negative facts and traversing the entailment graphs in the reverse order. For example, knowing that *Facebook does now own Yahoo*, we can infer that *Facebook has not purchased Yahoo*.

In this thesis, our focus is to learn high-quality entailment graphs that could support such inferences. We leave the actual construction of such semantics to future work.

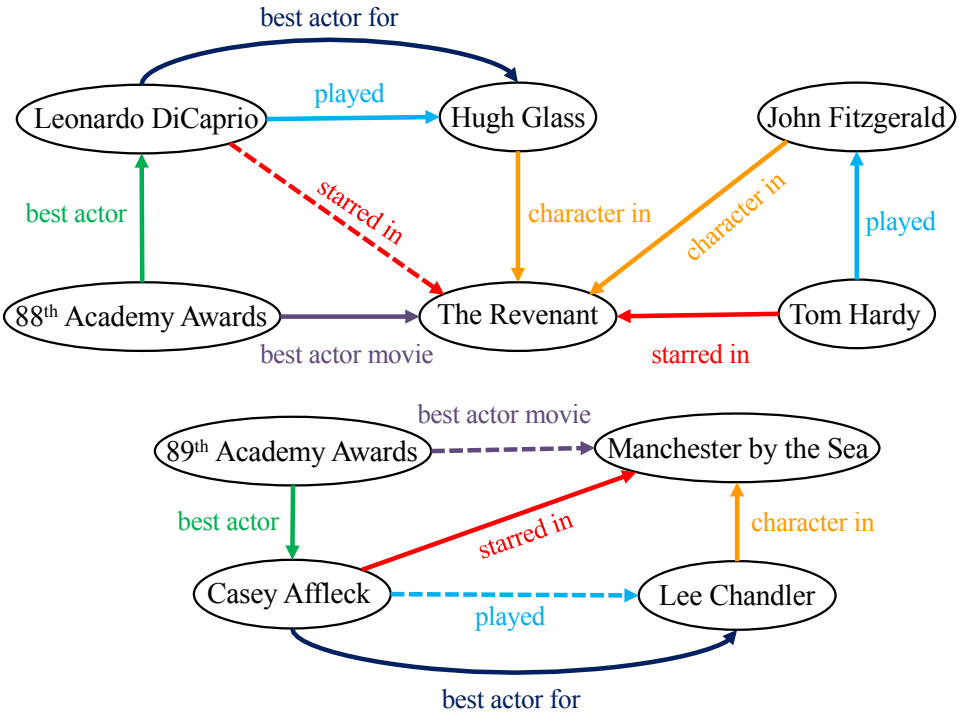


Figure 2.4: Examples knowledge graph. The dotted links are not observed, but can be predicted based on the observed links.

2.2 Link Prediction for Knowledge Graph Completion

Knowledge Graphs (KGs) are graph-structured knowledge bases that contain collections of facts about real world entities. Each fact or triple (p, e_1, e_2) in knowledge graphs represent a relation p between a head entity e_1 and a tail entity e_2.[10] Figure 2.4 shows an example knowledge graph between *actors*, *awards*, *roles* and *movies*. There are several knowledge graphs with expert or community annotated triples (Nguyen, 2020) including a) domain specific KGs such as GeneOntology KG (Consortium, 2015); b) generic KGs such as WordNet (Miller, 1995, 1998), YAGO (Suchanek et al., 2007), Freebase (Bollacker et al., 2008), NELL (Carlson et al., 2010), and DBPedia (Lehmann et al., 2015); and c) commercial KGs such as Google Knowledge Graph.

Knowledge graphs are useful for many natural language processing tasks (Nguyen, 2020). For example, both manually curated and automatically constructed KGs have been used for question answering (Ferrucci, 2012; Yao and Van Durme, 2014; Fader et al., 2014; Yin et al., 2015; Das et al., 2017b; Saha et al., 2018), semantic parsing

[10] It is common to write a triple as (s, p, o), i.e. a relation p between a subject s and an object o, but since we can have both subjects and objects as the head entity, and to be consistent with the notation in the rest of this thesis, we use (p, e_1, e_2).

(Krishnamurthy and Mitchell, 2012; Berant et al., 2013; Krishnamurthy et al., 2017), and coreference resolution (Dutta and Weikum, 2015; Zhang et al., 2019).

Unfortunately, even the largest existing KGs are incomplete. For example, more than 70% of people included in Freebase do not have a known place of birth, and more than 99% do not have an ethnicity (West et al., 2014). One of the key tasks in statistical relational learning is link prediction (Popescul and Ungar, 2003; Koller et al., 2007). The goal of link prediction for knowledge graph completion is to predict the missing links based on the existing ones. In our example KG (Figure 2.4), the solid links are observed, but the dotted links are missing. For example, the relation *starred in* between *Leonardo DiCaprio* and *The Revenant* is missing. However, it can be predicted by checking the existing patterns. For example, based on the relations between *Tom Hardy*, *John Fitzgerald* and *The Revenant*, we can extract the pattern that (*played, A, B*) and (*character in, B, C*) implies that (*starred in, A, C*), which in turn can be used to predict the missing *starred in* link.

There have been many attempts on the link prediction problem over nearly the last decade. Currently, embedding models that learn latent feature vectors for entities and relations hold the state-of-the-art for the link prediction task. The entities are usually encoded with one-dimensional vectors, but the relations are encoded with k-dimensional vectors (usually $k \in \{1,2,3\}$). Let $\mathcal{K}$ denote a knowledge graph of correct triples. For each triple (p, e_1, e_2), the methods learn a scoring function of its plausibility $f(p, e_1, e_2)$. The methods choose f such that the score $f(p, e_1, e_2)$ of a plausible triple $(p, e_1, e_2) \in \mathcal{K}$ is higher than the score $f(p', e_1', e_2')$ of an implausible triple $(p', e_1', e_2') \notin \mathcal{K}$ (Nguyen, 2020). The plausible triples are the correct ones in the knowledge graph and the implausible ones are usually built by corrupting the head or the tail entity of a correct triple. The methods optimize a loss function to learn the relation and entity embeddings.

In chapter 4, we show that the link prediction task and the entailment graph construction task are complementary. We perform link prediction on the knowledge graph of extracted triples from text. We predict new triples and use them to reduce the sparsity issue of the entailment graphs. In addition, we show that the learned entailment graphs can be used to improve the link prediction task.

In the rest of this section, we first introduce the loss functions used for the link prediction problem and then review some of the most notable link prediction methods categorized based on their scoring functions. We also discuss methods that explicitly model reasoning chains to tackle the link prediction problem.

2.2.1 Loss Functions for the Link Prediction Problem

For each correct triple (p, e_1, e_2), the methods create a corrupted triple by replacing the head or the tail entity with a random entity. We denote by $\mathcal{K}'(p, e_1, e_2)$ the set of triples $(p, e_1, e_2') \notin \mathcal{K}$ and $(p, e_1', e_2) \notin \mathcal{K}$. A common loss functions is the margin-based pairwise ranking loss that ranks correct triples higher than incorrect ones (Bordes et al., 2013). They define the loss function as:

$$\mathcal{L}_{\text{Margin}} = \sum_{\substack{(p,e_1,e_2)\in\mathcal{K} \\ (p,e_1',e_2')\in\mathcal{K}'(p,e_1,e_2)}} [\gamma - f(p,e_1,e_2) + f(p,e_1',e_2')]_+, \tag{2.6}$$

where $[x]_+ = \max(0,x)$ and γ is the margin hyper-parameter. The methods usually sample one incorrect triple per correct triple.

In addition, the negative log-likelihood of the logistic model is commonly used in recent link prediction methods. Trouillon et al. (2016) define the loss function as:

$$\mathcal{L}_{\text{Logistic}} = \sum_{(p,e_1,e_2)\in\mathcal{K}\cup\mathcal{K}'} \log(1+\exp(-I(p,e_1,e_2)f(p,e_1,e_2))), \tag{2.7}$$

where $I(p,e_1,e_2) = 1$ if $(p,e_1,e_2) \in \mathcal{K}$ and $I(p,e_1,e_2) = 0$, otherwise.

Equations 2.6 and 2.7 compute the score of one given triple (p,e_1,e_2) (and possibly its corresponding corrupted version) at a time (i.e., 1-1 scoring). Dettmers et al. (2018) propose to take a relation p and the head entity e_1 and compute the score for all candidate entities e_2 (i.e., 1-N scoring). They consider all non-positive entities as candidate negative entities. This speeds up the computation and makes it possible to have more number of corrupted triples.[11] They minimize the negative log-likelihood of the logistic model as:

$$\mathcal{L}_{\text{Logistic}} = \frac{1}{N} \sum_{(p,e_1,.)\in\mathcal{K}} \sum_{e_2} \log(1+\exp(-I(p,e_1,e_2)f(p,e_1,e_2)), \tag{2.8}$$

where $(p,e_1,.) \in \mathcal{K}$ denotes the set of all relation and head entities in the knowledge graph; and N is the number of entities. Equation 2.8 can be equivalently written as the binary cross entropy loss:

[11] In addition, for each triple (p,e_1,e_2), they add reciprocal (reverse) relations, i.e., (p^{-1},e_2,e_1), to the knowledge graph $\mathcal{K}$.

$$\mathcal{L}_{\mathbf{BCE}} = -\frac{1}{N} \sum_{(p,e_1,.)\in\mathcal{K}} [\sum_{e_2:\ (p,e_1,e_2)\in\mathcal{K}} \log(\sigma(f(p,e_1,e_2))) + \sum_{e_2':\ (p,e_1,e_2')\notin\mathcal{K}} \log(1-\sigma(f(p,e_1,e_2')))],$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ denotes the sigmoid function, and $\sigma(f(p,e_1,e_2))$ denotes the probability that (p,e_1,e_2) exists in the knowledge graph.

The above loss functions can be used for any of the scoring functions described in the following sections; however, each of the methods have usually used one of the two main loss functions defined in this section.

2.2.2 Translation-based Models

A number of link prediction models are inspired by the Word2Vec Skip-gram model (Mikolov et al., 2013). The embeddings learned by Word2Vec are known to preserve interesting translation invariance relationships in the embeddings space. For example, v_{king} - $v_{\text{man}} \approx v_{\text{queen}}$ - v_{woman}, where $v_{\text{word}} \in \mathbb{R}^d$ is the embedding vector of a word and d is the number of dimensions. Another example is the translation regularities between countries and their capitals (Mikolov et al., 2013). Figure 2.5 shows the two-dimensional projection of Word2Vec embeddings of some countries and their capitals. It can be seen that $v_{\text{country}} - v_{\text{capital}}$ is approximately a constant vector. For example, $v_{\text{France}} - v_{\text{Paris}} \approx v_{\text{Poland}} - v_{\text{Warsaw}}$. We can think of the constant vector as a vector $v_{\text{be\_capital\_of}}$ that translates a *country* into its *capital*. Therefore, we expect:

$$\begin{aligned} v_{\text{Paris}} + v_{\text{be\_capital\_of}} - v_{\text{France}} &\approx \mathbf{0} \\ v_{\text{Poland}} + v_{\text{be\_capital\_of}} - v_{\text{Warsaw}} &\approx \mathbf{0} \\ v_{\text{Rome}} + v_{\text{be\_capital\_of}} - v_{\text{Italy}} &\approx \mathbf{0}, \end{aligned}$$

where $\mathbf{0}$ is a d-dimensional vector of all zeros.

The TransE model (Bordes et al., 2013) generalizes the idea to all relations and entity pair, so that each relation's vector translates the head entity into the tail entity. They learn low-dimensional dense vectors for entities and relations to have $v_{e_1} + v_p \approx v_{e_2}$ for any correct triple (p,e_1,e_2), where $v_p \in \mathbb{R}^d$, $v_{e_1} \in \mathbb{R}^d$, and $v_{e_2} \in \mathbb{R}^d$ are the embedding vectors of p, e_1, and e_2. They define $f_{\text{TransE}}(p,e_1,e_2) = -\|v_{e_1} + v_p - v_{e_2}\|_{\ell_{1/2}}$, where $\ell_{1/2}$ means either the L_1 or the L_2-norm. In order to learn the embeddings, they

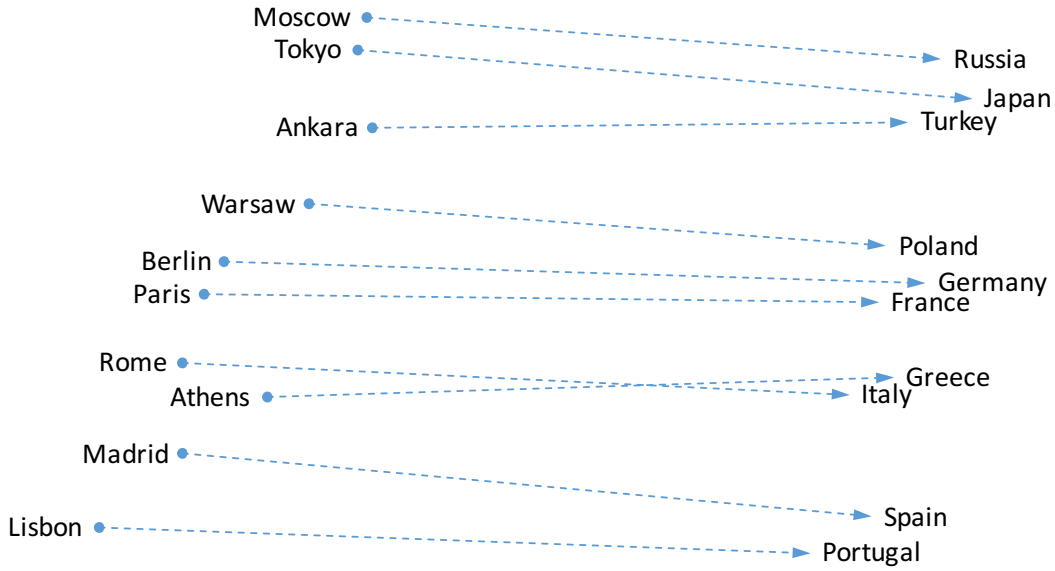


Figure 2.5: Two-dimensional projection of Word2Vec embeddings of some countries and their capitals. This figure is taken from Nguyen (2020) which is drawn based on the figure in Mikolov et al. (2013).

minimize the margin-based pairwise ranking loss to rank correct triples higher than incorrect ones.

The TransE model is capable of capturing one-to-one relationships, i.e., the case where each head and tail entity e_1 and e_2 are connected with at most one relation p. This is because after learning the embeddings, $v_{e_1} - v_{e_2}$ is a fixed vector that can be assigned to exactly one relation vector.[12] However, it cannot model one-to-many, many-to-one or, many-to-many relationships (Nguyen, 2020).

Various extensions of the TransE model resolve the above issue while still being based on the translation property (Wang et al., 2014; Ji et al., 2015; Nguyen et al., 2016; Ji et al., 2016; Ebisu and Ichise, 2018). For example, the TransH model (Wang et al., 2014) allows each relation to have its own hyper-plane, denoted by its normal vector $w_p \in \mathbb{R}^d$ as well as the translation vector $v_p \in \mathbb{R}^d$. The entities e_1 and e_2 are first projected to the hyper-plane as $v_{e_i} - w_p^T v_{e_i} w_p$, where $i \in \{1,2\}$. They are then connected by the translation vector. The scoring function is defined as:

$$f_{\text{TransH}}(p,e_1,e_2) = -\|(v_{e_1} - w_p^T v_{e_1} w_p) + v_p - (v_{e_1} - w_p^T v_{e_1} w_p).\|_{\ell_{1/2}}$$

[12] It can also be assigned to any paraphrase relation, but not to semantically different relations.

2.2.3 Linear Tensor Factorization based Models

A number of effective approaches for link prediction are linear or bi-linear and factorize a third-order binary tensor (Balazevic et al., 2019). These methods represent the set of triples with a binary tensor $\mathcal{X} \in \{0,1\}^{n \times n \times m}$, where n is the number of entities and m is the number of relations. We have $\mathcal{X}_{e_1 e_2 p} = 1$ if the triple (p, e_1, e_2) is present in the knowledge graph. Otherwise, $\mathcal{X}_{e_1 e_2 p} = 0$ if the triple does not exist in the knowledge graph or is unknown. $\mathcal{X}_p$ ($p \in \{1, \ldots, m\}$) refers to the frontal slice of the tensor $\mathcal{X}$ and represents the entity pairs of the p-th relation (Nickel et al., 2011). The low-dimensional representations for relations and entities are learned by doing tensor factorization on $\mathcal{X}$ and estimating it with low-dimensional matrices.

RESCAL[13] (Nickel et al., 2011) estimates $\mathcal{X}_p$ by doing a rank-d matrix factorization:

$$\mathcal{X}_p \approx A Y_p A^T,$$

where $p \in \{1, \ldots, m\}$. The matrix $A \in \mathbb{R}^{n \times d}$ stores the d-dimensional representations of the entities and the matrix $Y_p \in \mathbb{R}^{d \times d}$ stores the $d \times d$ representation of the relation p. The link prediction scoring function is defined as $f_{\text{RESCAL}}(p, e_1, e_2) = A_{e_1} Y_p A_{e_2}^T$, where A_{e_i} is the e_i-th row of the matrix A and stores the embedding of the entity e_i.

RESCAL stores d^2 parameters for each relation which makes the model computationally expensive and also might lead to overfitting to the data. The DISTMULT model (Yang et al., 2015) reduces the complexity of RESCAL by restricting Y_p to be a diagonal matrix. They define $f_{\text{DISTMULT}}(p, e_1, e_2) = A_{e_1} \text{diag}(v_p) A_{e_2}^T$, where $\text{diag}(x)$ is an operator that maps a vector $x \in \mathbb{R}^d$ to a diagonal matrix with the vector x as its diagonal. The DISTMULT model reduces the computational complexity of RESCAL and also achieves better results on the link prediction task.

RESCAL and DISTMULT can only model symmetric relations because they assign the same link prediction scores to (p, e_1, e_2) and (p, e_2, e_1). Therefore, they are not suitable for modeling asymmetric relations. More recent tensor factorization approaches resolve this issue. ComplEx (Trouillon et al., 2016) extends DISTMULT by leveraging complex valued embeddings and is capable of handling asymmetric relations. In addition, SimplE (Kazemi and Poole, 2018) learns two embeddings for each relation, one for being a head entity and one for being a tail entity. TuckER (Balazevic et al., 2019) is a linear model based on the Tucker decomposition of the matrix $\mathcal{X}$.

[13] Also known as Bilinear.

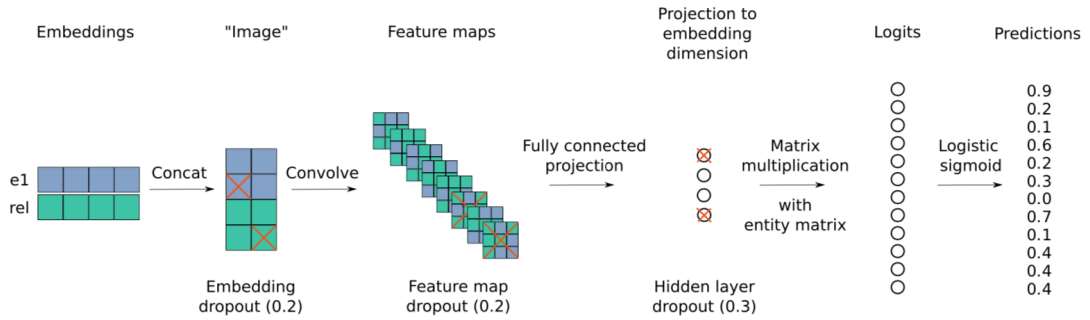


Figure 2.6: The overview of the ConvE model. This figure is taken from Dettmers et al. (2018).

We note that all the link prediction models that are capable of predicting the scoring function $f(p, e_1, e_2)$ for any given triple can be used to complete the third-order tensor $\mathcal{X}$ by computing scores for the entries of the tensor; however, the methods in this section are linear and based on tensor decomposition models. In contrast, the methods in section 2.2.4 are non-linear and based on neural networks.

2.2.4 Neural Network based Models

There are several attempts that use deep neural networks to solve the link prediction problem. The neural tensor network (NTN) model uses a bilinear tensor layer that relates the two entity vectors across multiple dimensions (Socher et al., 2013). ProjE (Shi and Weninger, 2017) is a simpliefied version of NTN. ER-MLP (Dong et al., 2014) concatenates the embeddings of the head entity, relation, and the tail entity. It then feeds the concatenated vector to a multi-layer perceptron (MLP with only a single layer) to compute the scoring function.

ConvE (Dettmers et al., 2018) is designed based on convolutional neural networks. It reshapes the one-dimensional vector of the head entity and the relation into two-dimensional matrices. It concatenates the reshaped matrices and applies a convolutional layer to the resulting matrix. The resulting feature map is then vectorized and projected into a d-dimensional space. The result is matched with candidate tail entities by computing the dot product of the two vectors. Figure 2.6 shows the overview of the ConvE model. The probability of each candidate tail entity is computed by applying the sigmoid function (Dettmers et al., 2018). The link prediction scoring function is defined as (Nguyen, 2020):

$$f_{\text{ConvE}}(p, e_1, e_2) = v_{e_2}^T \text{ReLU}(vec(\text{RELU}([\overline{v_{e_1}}; \overline{v_r}] * \omega))\mathbf{W}),$$

where $\overline{v_{e_1}}$ and $\overline{v_r}$ are reshaped matrices of v_{e_1} and v_r; $\text{vec}(x)$ vectorizes a matrix x; ω is the set of convolutional filters; $x * \omega$ means that a convolution operation is applied to a matrix x via convolutional filters ω; and $\text{ReLU}(x) = \max(\mathbf{0}, x)$ for a vector $x \in \mathbb{R}^d$ with $\mathbf{0} \in \mathbb{R}^d$ being a vector of all 0 values. ConvE optimizes the negative log-likelihood of the logistic model (binary cross-entropy loss) and uses 1-N scoring (section 2.2.1).

ConvKB (Nguyen et al., 2018) extends ConvE by applying a convolutional layer over the embedding triples by representing each triple (p, e_1, e_2) as a 3-column matrix. ConvE-TransE (Shang et al., 2019) is similar to ConvE, but it tries to keep translational propertites between the entities and relations.

In our experiments in chapter 4, we use ConvE since it is one of the state-of-the-art models and scales to the size of our extracted triples.

2.2.5 Modeling Reasoning Chains for Link Prediction

Recently, there has been an increasing interest in statistical relational learning to explicitly model reasoning chains for link prediction (Luo et al., 2015; Guu et al., 2015; Lin et al., 2015a; Toutanova et al., 2016; Das et al., 2017a, 2018; Shen et al., 2018; Chen et al., 2018). The triple-based models discussed in section 2.2 only consider direct relations between entities, e.g., (*starred in, Leonardo DiCaprio, The Revenant*). However, they do not use reasoning chains (multi-step relation paths) to train link prediction models, e.g., (*played , Leonardo DiCaprio, Hugh Glass*) and (*character in , Hugh Glass, The Revenant*).

In particular, Luo et al. (2015) generate relation paths between entities and form word sequences by considering the entities and relations as pseudo-words. They apply Word2Vec (Mikolov et al., 2013) to the constructed sequences and learn embeddings for the relations and entities. They show that initializing relation and entity vectors with the learned Word2Vec embeddings improve link prediction models such as TransE. Guu et al. (2015) use relation paths to construct new triples and augment the training data with them. They propose a compositional training objective which is applicable to link prediction scoring functions that are composable, i.e., it is possible to perform vector space composition to combine the embeddings of the individual relations in a path into the embedding of the path (Guu et al., 2015; Toutanova et al., 2016). This includes models such as Bilinear (RESCAL) and DISTMULT, where the embedding matrix of a path is obtained by multiplication of the embedding matrices of the relations. It also includes TransE, where the embedding vector of a path is ob-

tained by summing up the embeddings of its relations. The path-based TransE-RNN (PTransE-RNN) method of Lin et al. (2015a) also constructs relation paths between entity pairs and models them by using a recurrent neural network (RNN). They extend TransE to use reliable relation paths in addition to the direct relations between head and tail entities.

ALL-PATHS method of Toutanova et al. (2016) is similar to the method of Guu et al. (2015), but they consider both relations and entities on the path, instead of using only the relations. In addition, they propose an exact dynamic programming model based on the observation that compositional representation of relation paths are in fact decomposable (Toutanova et al., 2016). Their method is scalable to all paths of bounded length between entity pairs. Similar to PTransE-RNN, the method of Das et al. (2017a) uses RNNs to model relation paths, however, they also take the intermediate entities into account (Nguyen, 2020).

The above methods construct relation paths in advance and use them to learn a link prediction model. These models use random paths between fixed entity pairs. Das et al. (2018) propose a new algorithm, MINERVA, which does not need to pre-compute inference paths and can answer queries given a relation and a head entity, e.g., (*starred in, Leonardo DiCaprio, ?*). Performing random walk is impractical in large knowledge graphs without access to the destination entity because there are combinatorially many paths starting from one entity. Das et al. (2018) propose a neural reinforcement learning approach which learns how to navigate the knowledge graph to obtain the answer conditioned on an input query. Their method learns an RNN-based policy, that given a query $(p, e_1, ?)$, it starts from the entity e_1 and at each step selects a relation edge based on the query and the entire history. MINERVA learns to take an optimal set of decisions (relation edges) to maximize the reward by reaching the correct answer entity (entities). They train their agent using policy gradients, more specifically REINFORCE (Williams, 1992).

The methods for random walk in knowledge graphs that use policy gradients suffer from sparse rewards, i.e., only a small fraction of walks reach the correct entity. In order to overcome this challenge, Shen et al. (2018) propose a graph-walking agent called M-Walk. M-Walk uses an RNN to encode the states that are mapped separately to the policy and the Q-values (action values). Their method combines the RNN policy with Monte Carlo Tree Search (MCTS) that generate trajectories yielding more positive rewards. Chen et al. (2018) frame the knowledge graph reasoning task as two sub-steps, i.e., path-finding and path-reasoning. MINERVA and M-Walk handle path-

finding by discovering paths that end in the correct entity node; however, they do not take path-reasoning into account since all paths ending in the correct entity, regardless of being meaningful or not, get the reward of $+1$. Chen et al. (2018) propose a variational inference framework (DIVA) which jointly models the path-finding and path-reasoning steps.

The most recent triple-based models such as TuckER (Balazevic et al., 2019) still perform well on the knowledge graph completion task because they can implicitly learn from the existing paths; however, the main advantage of the methods that explicitly model the reasoning chains is their transparency. A triple-based model can only output a scoring function $f(p, e_1, e_2)$ that assigns high scores to triples based on the embeddings of their relations and entities. For example, translation-based models assign a high score to (p, e_1, e_2) if $v_{e_1} + v_p \approx v_{e_2}$, where v_{e_i} (v_p) is the embedding of e_i (p), but they cannot necessarily say why the embeddings are involved in this geometric property. In addition to the scoring function, methods that model reasoning chains can provide explanations for high-scoring triples in complex scenarios. Some of the above methods can assign scores to different paths between entity pairs, and some others (MINERVA, M-Walk, and DIVA) can even find the high scoring paths. The paths can be used as explanations for assigning high scores to correct triples.

In chapter 4, we discuss how we can use entailment graphs as another interpretable resource to improve the link prediction task.

2.3 Contextual Relation and Entity Embeddings

Recently, large pre-trained language models (LMs) such as ELMo (Peters et al., 2018), GPT (Radford et al., 2018), BERT (Devlin et al., 2019), XLM (Lample and Conneau, 2019), XLNet (Yang et al., 2019) and RoBERTa (Liu et al., 2019) have shown significant performance gains on many natural language processing tasks. They generalize traditional word embeddings such as Word2Vec (Mikolov et al., 2013) and GloVe embeddings (Pennington et al., 2014) by extracting *contextual* (context sensitive) word representations.

These models usually have two steps: *pre-training* and *fine-tuning* (Devlin et al., 2019). During pre-training, the models are trained on large unlabeled text by performing different pre-training tasks such as predicting the next word in a sequence (language modeling) or predicting a masked word in an arbitrary position in a sequence (masked language modeling). The earlier models have used multi-layer Bidirectional

Long short-term memory (LSTM; Hochreiter and Schmidhuber, 1997) architecture (Peters et al., 2018), but the more recent ones are based on the Transformers architecture (Vaswani et al., 2017). During fine-tuning, the models are first initialized with the pre-trained parameters, and then fine-tuned using data from downstream tasks. Each of the downstream tasks could have its own model; however, they are all initialized with the same pre-trained parameters so that the information in the pre-trained models can be transferred to them (Devlin et al., 2019).

In chapter 5, we propose a model that uses contextualized word embeddings to learn relational entailment. In the rest of this section, we first summarize some of the existing work on representation learning for relations that use contextualized word embeddings (section 2.3.1). We then review some works that analyze the relational knowledge that are already present in the existing pre-trained LMs (section 2.3.2).

2.3.1 Contextual Relation Representation

The Matching the Blanks (MTB) method of Baldini Soares et al. (2019) learns relation representations directly from text by building on Lin and Pantel (2001)'s extensions of Harris' distributional representation hypothesis (Harris, 1954) to relations, and recent advances in learning contextual word representations.

They assume access to a an entity linked text corpus and learn mappings from *relation statements* to *relation representations*. They define a sequence of tokens as $x = \{x_0, \ldots, x_n\}$, where $x_0 = [\text{CLS}]$ and $x_n = [\text{SEP}]$ are special start and end tokens as used in Devlin et al. (2019). Let $s_1 = (i, j)$ and $s_2 = (k, l)$ be non-overlapping open intervals with $j < k$ that specify two entities. They define a relation statement as $\rho = (x, s_1, s_2)$. They learn a function $h_\rho = f_\theta(\rho)$ that maps the relation statement to a fixed length vector $h_\rho \in \mathbb{R}^d$ representing the relation stated in x between the two entities that are marked by s_1 and s_2 (Baldini Soares et al., 2019). They test different architectures for the relation encoder f_θ based on the Transformers architecture used in BERT. They note that adding special start and end tokens before and after each entity mention, and representing the relation between the two entities by concatenating the final hidden states of the entity start tokens is an effective way of learning f_θ.

They train f_θ in an unsupervised way by encouraging the dot product $f_\theta(\rho)^T f_\theta(\rho')$ to be high if they express semantically similar relations. The dot product should be low, otherwise. Similar to the intuition behind the similarity scores defined in section 2.1.1, they assume that if the two relation ρ and ρ' hold between the same entity pairs,

they are likely to be semantically similar. They define a label l between two relation statements. The label is 1 if the two entities are the same, and 0, otherwise. They define the following binary classifier:

$$\Pr(l = l|\rho,\rho') = \frac{1}{1+\exp(-f_\theta(\rho)^T f_\theta(\rho'))}.$$

They denote their training data by $\mathcal{D}=\left[(\rho^i, e_1^i, e_2^i)\right]_{i\in\{1,\ldots,N\}}$, where N is the number of relation statements ρ^i linked to entities e_1^i and e_2^i. In order to avoid the encoder to just rely on the entity pairs rather than the relation itself, they use a modified corpus $\tilde{\mathcal{D}}=\left[(\tilde{\rho}^i, e_1^i, e_2^i)\right]_{i\in\{1,\ldots,N\}}$, where $\tilde{\rho}_i$ is the same as ρ_i except that one or two entities are masked with a special [BLANK] token. They mask each entity with the probability $\alpha = 0.7$, so in only half of the examples both entities are masked. Without the blank tokens, the encoder can be optimized to learn the named entity linker used to generate $\mathcal{D}$. They define the matching the blank loss function as a binary cross entropy loss that assigns high scores to pairs of relation statements sharing the same entity pairs, and low scores to other pairs of relation statements. They combine their loss function with the original masked language model loss in BERT to train a task agnostic encoder f_θ. The encoder can be used in either an unsupervised manner, or a supervised manner by fine-tuning on labeled data. Their method currently holds the state-of-the-art on a number of relation extraction datasets.

In chapter 5, we propose a novel method to lean relation representations based on contextualized embeddings of the relations. Similar to the MTB model, we use a named entity linked corpus and learn the representations in an unsupervised way. However, we work on a corpus of parser-based extractions and our architecture depends on the relation tokens, not the entity tokens. We learn two different sets of contextual and out-of-context embeddings for relations. Hence, while the MTB model learns representations that can capture paraphrasing, our representations learn a directional score that is suitable for entailment. In addition, we do not need to introduce the blank tokens and our method is more scalable since it simultaneously compares contextual embeddings of one relation statement with the out-of-context embeddings of a large number of relations.

2.3.2 Language Models as Knowledge Graphs

Recent work has shown promising results in extracting relational knowledge from pretrained LMs without any fine-tuning. Petroni et al. (2019) introduce the LAMA (LAn-

guage Model Analysis) benchmark, consisting of a set of facts in a number of knowledge sources. They define that a pre-trained LM knows a fact or triple (r, e_1, e_2) such as *(bought, Facebook, Instagram)* if it can predict masked objects in clozed sentences t_r such as *Facebook bought* __. The model predicts a fact as:

$$\hat{e}_2 = \text{argmax}_{e'_2 \in V} \text{Pr}_{\text{LM}}(e'_2 | t_r),$$

where V is the vocabulary and $\text{Pr}_{\text{LM}}(e'_2|t_r)$ is the probability that the LM predicts e'_2 in the blank conditioned on the prompt t_r (Jiang et al., 2020).

Petroni et al. (2019) analyze various pre-trained LMs on different types of knowledge. They test for relations between entities in Wikipedia using Google-RE corpus[14] and the T-REx knowledge source (Elsahar et al., 2018). They also investigate commonsense relations in ConceptNet (Speer and Havasi, 2012) and the knowledge necessary to answer context-insensitive natural language questions in SQuAD (Rajpurkar et al., 2016). Their analysis show that the largest BERT pre-trained LM captures relational knowledge comparable to the relations extracted by an off-the-shelf relation extractor, although the performance is poor for some types of relations, particularly the many-to-many relations.

The prompts in Petroni et al. (2019) are based on manually created templates. Jiang et al. (2020) argue that these prompts are sub-optimal because LMs might be aware of the knowledge in a completely different context. Therefore, the prompts provide a lower-bound estimate of the knowledge that actually exists in the LMs. They propose methods to generate alternative prompts that could reveal that the LMs in fact capture the queried knowledge. They test a mining-based approach that relies on the shared entity pairs between relations, and a paraphrasing approach that translates the templates into a target language and translates them back to English. They show that these approaches as well as their combination improve the reported accuracy in the previous works (Jiang et al., 2020).

In our work in chapter 5, we also query language models, but our work has substantial differences. The above work handles single token queries because querying for multiple tokens has some technical challenges. They only consider objects (the entity e_2), but not relations as many of them span multiple tokens (Petroni et al., 2019). In contrast, we query relations that hold between the entities and handle multiple-token queries. In addition, while the previous work investigates the pre-trained LMs, we fine-tune the embeddings and learn directional scores for all relations. The pre-trained

[14] https://code.google.com/archive/p/relation-extraction-corpus/

LMs have access to a static set of knowledge, but our parser-based approach can extract relations from any new text and generalize to novel relations using the discovered entailment rules.

Chapter 3

Learning Typed Entailment Graphs with Global Soft Constraints

In this chapter, we present a new method for learning typed entailment graphs from text. We extract predicate-argument structures in the form of relation and entity-pair triples from multiple-source news corpora, and compute local distributional similarity scores to learn entailments between relations with typed entities (e.g., *person* contracted *disease*). Previous work has used transitivity constraints to improve local decisions, but these constraints are intractable on large graphs. We instead propose a scalable method that learns globally consistent similarity scores based on new soft constraints that consider both the structures across typed entailment graphs and inside each graph. Learning takes only a few hours to run over 100K relations and our results show large improvements over local similarity scores on two entailment datasets. We further show improvements over paraphrases and entailments from the Paraphrase Database, and prior state-of-the-art entailment graphs. We show that the entailment graphs improve performance in a downstream task.

3.1 Introduction

The lack of a well-established form-independent semantic representation for natural language is the most important single obstacle to bridging the gap between queries and text resources. In this chapter, we learn meaning postulates such as *buying* entails *owning* that can be used to augment the standard form-dependent semantics. Our goal is to learn entailment rules between typed relations with two entities, where the type of each relations is determined by the types of its entities. We construct *typed entailment*

graphs, with typed relations as nodes and entailment rules as edges.

Entailment rules are detected by computing a similarity score between the typed relations based on the distributional inclusion hypothesis, which states that a word (relation) p entails another word (relation) q if in any context that p can be used so can be q (Dagan et al., 1999; Geffet and Dagan, 2005; Herbelot and Ganesalingam, 2013; Kartsaklis and Sadrzadeh, 2016). Earlier works have taken a "local learning" approach (Lin, 1998; Weeds and Weir, 2003; Szpektor and Dagan, 2008; Schoenmackers et al., 2010), i.e., learning entailment rules independently from each other.

One problem facing local learning approaches is that many correct edges are not identified because of data sparsity and many wrong edges are spuriously identified as valid entailments. A "global learning" approach, where dependencies between entailment rules are taken into account, can improve the local decisions significantly. Berant et al. (2011) imposed *transitivity constraints* on the entailments, such that the inclusion of rules $p \rightarrow q$ and $q \rightarrow r$ implies that of $p \rightarrow r$. While they showed transitivity constraints to be effective in learning entailment graphs, the Integer Linear Programming (ILP) solution of Berant et al. (2011) is not scalable beyond a few hundred nodes. In fact, the problem of finding a maximally weighted transitive subgraph of a graph with arbitrary edge weights is NP-hard (Berant et al., 2011).

We proposes a scalable solution that does not rely on transitivity closure, but instead uses two global soft constraints that maintain structural similarity both across and within each typed entailment graph (Figure 3.1). We introduce an unsupervised framework to learn globally consistent similarity scores given local similarity scores (section 3.4). Our method is highly parallelizable and takes only a few hours to apply to more than 100K relations.[1,2]

Our experiments (section 3.6) show that the global scores improve significantly over local scores and outperform state-of-the-art entailment graphs on two standard entailment rule datasets (Berant et al., 2011; Levy and Dagan, 2016; Holt, 2018). We ultimately intend the typed entailment graphs to provide a resource for entailment and paraphrase rules for use in semantic parsing and open domain question-answering, as has been done for similar resources such as the Paraphrase Database (PPDB; Ganitkevitch et al., 2013; Pavlick et al., 2015) in Wang et al. (2015b); Dong et al. (2017).[3]

[1] We performed our experiments on a 32-core 2.3 GHz machine with 256GB of RAM.

[2] Our code, extracted triples and the learned entailment graphs are available at `https://github.com/mjhosseini/entGraph`.

[3] The relations inside each clique in the entailment graphs are considered to be paraphrases.

With that end in view, we have included a comparison with PPDB in our evaluation on the entailment datasets. We also show that the learned entailment rules improve performance on a question-answering task (section 3.7) with no tuning or prior knowledge of the task.

The structure of this chapter is as follows. We discuss related works in section 3.2 and our framework for computing local entailment scores in section 3.3. We propose our new model in section 3.4 and discuss the experimental setup in section 3.5. We analyze the results in section 3.6 and use the entailment graphs to improve a downstream task in section 3.7. We conclude the chapter in section 3.8.

3.2 Related Work

Our work in this chapter is closely related to Berant et al. (2011), where entailment graphs are learned by imposing transitivity constraints on the entailment rules. However, the exact solution to the problem is not scalable beyond a few hundred relations, while the number of relations that we capture is two orders of magnitude larger (section 3.5). Hence, it is necessary to resort to approximate methods based on assumptions concerning the graph structure. Berant et al. (2012) and Berant et al. (2015) propose Tree-Node-Fix (TNF), an approximation method that scales better by additionally assuming the entailment graphs are “Forest-Reducible", where a relation cannot entail two (or more) relations q and r such that neither $q \rightarrow r$ nor $r \rightarrow q$ (FRG assumption). However, as discussed in section 2.1.2, *the FRG assumption is not correct for many real-world domains*. For example, a person *visiting* a place entails both *arriving* at that place and *leaving* that place, while the latter do not necessarily entail each other. Our work injects two other types of prior knowledge about the structure of the graph that are less expensive to incorporate and yield better results on entailment rule datasets.

Abend et al. (2014) learn entailment rules over multi-word relations with different levels of compositionality. We also consider multi-word relations in our graphs (section 3.3.1). Pavlick et al. (2015) add various relationships, including entailment, to phrase pairs in PPDB. This includes a broader range of entailment rules such as lexical entailment. In contrast to our method, these works rely on supervised data and take a local learning approach.

Another related strand of research is link prediction (Socher et al., 2013; Bordes et al., 2013; Riedel et al., 2013; Yang et al., 2015; Trouillon et al., 2016; Dettmers et al., 2018; Balazevic et al., 2019), where the source data are extractions from text, facts in

knowledge bases, or both (section 2.2). Unlike our work, which directly learns entailment rules between relations, these methods aim at predicting the source data, i.e., whether two entities have a particular relation. The common wisdom is that entailment rules are by-product of these methods (Riedel et al., 2013). However, this assumption has not usually been explicitly evaluated. Explicit entailment rules provide explainable resources that can be used in downstream tasks. Our experiments show that our method significantly outperforms ConvE, a state-of-the-art link prediction method.

3.3 Computing Local Similarity Scores

We first extract predicate-argument structures in the form of relation and entity-pair triples using a Combinatory Categorial Grammar (CCG; Steedman, 2000) semantic parser (section 3.3.1). We map the entities to their Wikipedia URLs using a named entity linker (section 3.3.2). We extract types such as *person* and *disease* for each entity (section 3.3.2). We then compute local similarity scores between relation pairs (section 3.3.3).

3.3.1 Relation Extraction

We run the semantic parser of Reddy et al. (2014), GraphParser, on the NewsSpike corpus (Zhang and Weld, 2013) to extract binary relations between entity pairs. GraphParser uses CCG syntactic derivations and λ-calculus to convert sentences to neo-Davisonian semantics, a first-order logic that uses event identifiers (Parsons, 1990). For example, for the sentence, *Obama visited Hawaii in 2012*, GraphParser produces the logical form

$$\exists e.visit_1(e, Obama) \wedge visit_2(e, Hawaii) \wedge visit_{in}(e, 2012),$$

where e denotes an event. We consider a relation for each pair of entities, hence, there will be three relations for the above sentence: $visit_{1,2}$ with entities *(Obama,Hawaii)*, $visit_{1,in}$ with entities *(Obama,2012)* and $visit_{2,in}$ with entities *(Hawaii,2012)*. We currently only use extracted triples that involve two named entities or one named entity and a noun. We constrain the triples to have at least one named entity to reduce ambiguity in finding entailments.

We perform a few automatic post-processing steps on the output of the parser. First, we normalize the predicates by lemmatization of their head words. Passive predicates

are mapped to active ones and we extract negations and particle verb predicates. Next, we discard unary relations and relations involving coordination of arguments, e.g., we discard the relation $visit_{1,1}$ with entities *(Barack,Michelle)* extracted from the sentence *Barack and Michelle visited Hawaii in 2012*. Finally, whenever we see a relation between a subject and an object, and a relation between object and a third argument connected by a prepositional phrase, we add a new relation between the subject and the third argument by concatenating the relation name with the object. For example, for the sentence *China has a border with India*, we extract a relation $have\ border_{1,with}$ between *China* and *India*. We perform a similar process for PPs attached to VPs. For example, for the sentence *Bangladesh maintains relations with Japan.*, we extract a relation $maintain\ relationships_{1,with}$ between *Bangladesh* and *Japan*. Most of the light verbs and multi-word predicates will be extracted by the above post-processing (e.g., $take\ care_{1,of}$) which will recover many salient ternary relations.

While entailments and paraphrasing can benefit from n-ary relations, e.g., *person visits a location in a time*, we currently follow previous work (Lewis and Steedman, 2013a; Berant et al., 2015) in confining our attention to binary relations, leaving the construction of n-ary graphs to future work.

3.3.2 Linking and Typing entities

Entailment and paraphrasing depend on the context. While using exact context is impractical in forming entailment graphs, many authors have used the type of the entities to disambiguate polysemous predicates (Berant et al., 2011, 2015; Lewis and Steedman, 2013a; Lewis, 2014). Typing also reduces the size of the entailment graphs.

Since named entities can be referred to in many different ways, we use a named entity linking tool to normalize the named entities. In the experiments below, we use AIDALight (Nguyen et al., 2014), a fast and accurate named entity linker, to link named entities to their Wikipedia URLs (if any). We thus type all entities that can be grounded in Wikipedia. We first map the Wikipedia URL of the entities to Freebase (Bollacker et al., 2008). We select the most notable type of the entity from Freebase and map it to FIGER types (Ling and Weld, 2012) such as *building*, *disease*, *person* and *location*, using only the first level of the FIGER type hierarchy.[4] For example, instead of *event/sports_event*, we use *event* as type. If an entity cannot be grounded in Wikipedia or its Freebase type does not have a mapping to FIGER, we assign the

[4] 49 types out of 113 FIGER types.

default type *thing* to it.

Lewis and Steedman (2013a) and Lewis (2014) learn types using topic modeling, in particular, Latent Dirichlet Allocation (LDA; Blei et al., 2003). They form a document for each unary predicate (e.g., visit$_1$), based on all its arguments (e.g., *Obama*). The types correspond to the learned topics, and are assigned human-readable formats according to their arguments. We tried this method in our preliminary experiments; however, we noticed significant improvements using our typing method. The topic modeling types were relatively accurate for frequent entities, but they were very noisy for infrequent ones. In contrast, FIGER types provide more accurate and fine-grained types for entities.

3.3.3 Local Distributional Similarities

For each typed relation (e.g., $visit_{1,2}$ with types *person,location*), we extract a feature vector. We use as feature types the set of entity pair strings (e.g., *Obama-Hawaii*) that instantiate the relation. The value of each feature is the pointwise mutual information (PMI) between the relation and the feature. We use the feature vectors to compute three local similarity scores (both symmetric and directional) between typed relations: Weeds (Weeds and Weir, 2003), Lin (Lin, 1998), and Balanced Inclusion (BInc; Szpektor and Dagan, 2008) similarities (section 2.1.1).

3.4 Learning Globally Consistent Entailment Graphs

We learn globally consistent similarity scores based on local similarity scores. The global scores will be used to form typed entailment graphs.

3.4.1 Problem Formulation

Let T be a set of types and U be a set of untyped relations. We denote by $\bar{R}(t_1,t_2)$ the set of typed relations $u(:t_1,:t_2)$, where $t_1,t_2 \in T$ and $u \in U$. Each $u(:t_1,:t_2) \in \bar{R}(t_1,t_2)$ takes as input entities of types t_1 and t_2. An example of a typed relation is $win_{1,2}$*(:team,:event)* that can be instantiated with $win_{1,2}$*(Seahawks: team, Super Bowl: event)*.

We define $R(t_1,t_2) = \bar{R}(t_1,t_2) \cup \bar{R}(t_2,t_1)$. We often denote elements of $R(t_1,t_2)$ by p, q and r, where each element is a typed relation as above. For a $p{=}u(:t_1,:t_2) \in R(t_1,t_2)$, we denote by $\pi(p){=}u$, $\tau_1(p){=}t_1$ and $\tau_2(p){=}t_2$. We compute distributional

T	set of types (e.g., person, disease)
U	set of untyped relations (e.g., $win_{1,2}$)
$u(:t_1,:t_2)$	typed relations: $t_1,t_2 \in T$
$\bar{R}(t_1,t_2)$	set of typed relations $u(:t_1,:t_2)$
$R(t_1,t_2)$	union of $\bar{R}(t_1,t_2)$ and $\bar{R}(t_2,t_1)$
R	$\bigcup_{t_1,t_2} R(t_1,t_2)$
$p,q,r \in R$	convenience variables
$\pi(p)$	projection of p to its untyped relation
$\tau_1(p)$	projection of p to its 1st type
$\tau_2(p)$	projection of p to its 2nd type
$w^0_{pq}(w_{pq})$	local (global) similarity score for typed relations p and q
$\mathbf{W}^0(t_1,t_2)$ ($\mathbf{W}(t_1,t_2)$)	matrix containing all local (global) similarity scores between relations with types t_1,t_2
$\mathbf{W}^0$ ($\mathbf{W}$)	local (global) block-diagonal matrix of all similarity matrices $\mathbf{W}^0(t_1,t_2)$ ($\mathbf{W}(t_1,t_2)$)

Table 3.1: The main variables in our formulation.

similarities between relations with the same entity types. We denote by $\mathbf{W}^0(t_1,t_2) \in [0,1]^{|R(t_1,t_2)|\times|R(t_1,t_2)|}$ the (sparse) matrix containing all local similarity scores w^0_{pq} between relations p and q with types t_1 and t_2, where $|R(t_1,t_2)|$ is the size of $R(t_1,t_2)$.[5]

The relations can entail each other with the same entity order (direct) or in the reverse order, i.e., $u(:t_1,:t_2)$ might entail $v(:t_1,:t_2)$ or $v(:t_2,:t_1)$. For the graphs with the same types (e.g., t_1=t_2=*person*), we keep two copies of the relations one for each of the possible orderings. This allows us to model entailments with reverse entity orders, e.g., *is son of*$_{1,2}$*(:person*$_1$*,:person*$_2$*)* $\rightarrow$ *is parent of*$_{1,2}$*(:person*$_2$*,:person*$_1$*)*.

We define $R = \bigcup_{t_1,t_2} R(t_1,t_2)$, the set of all typed relations, and $\mathbf{W}^0$ as a block-diagonal matrix consisting of all the local similarity matrices $\mathbf{W}^0(t_1,t_2)$. Similarly, we define $\mathbf{W}(t_1,t_2)$ and $\mathbf{W}$ as the matrices consisting of globally consistent similarity scores w_{pq} we wish to learn.

The global similarity scores are used to form entailment graphs by thresholding $\mathbf{W}$. For a $\delta > 0$, we define typed entailment graphs as $G_\delta(t_1,t_2) = \big(R(t_1,t_2), E_\delta(t_1,t_2)\big)$,

[5]For each similarity measure, we define one separate matrix and run the learning algorithm separately, but for simplicity of notation, we do not show the similarity measure names.

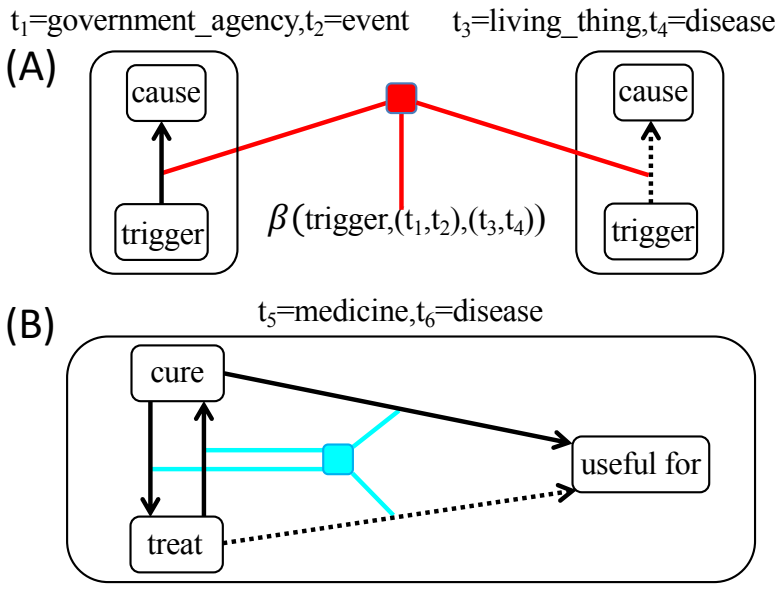


Figure 3.1: Learning entailments that are consistent (A) across different but related typed entailment graphs and (B) within each graph. $0 \leq \beta \leq 1$ determines how much different graphs are related. The dotted edges are missing, but will be recovered by considering relationships shown by across-graph (red) and within-graph (light blue) connections.

where $R(t_1,t_2)$ are the nodes and $E(t_1,t_2) = \{(p,q)|p,q \in R(t_1,t_2), w_{pq} \geq \delta\}$ are the edges of the entailment graphs.

Table 3.1 shows a summary of the variables we use in our formulation.

3.4.2 Learning Algorithm

Existing approaches to learn entailment graphs from text miss many correct edges because of data sparsity, i.e., the lack of explicit evidence in the corpus that a relation p entails another relation q. The goal of our method is to use evidence from the existing edges that have been assigned high confidence to predict missing ones, and remove spurious edges. We propose two global soft constraints that maintain structural similarity both across and within each typed entailment graph. The constraints are based on the following two observations.

First, it is standard to learn a separate typed entailment graph for each (plausible) type-pair because entities provide necessary disambiguation for the relation meaning (Berant et al., 2011, 2012; Lewis and Steedman, 2013a,b; Berant et al., 2015). However, many entailment rules for which we have direct evidence only in a *few* subgraphs

$$J(\mathbf{W} \geq \mathbf{0}, \vec{\beta} \geq \mathbf{0}) = \mathcal{L}_{\text{withinGraph}} + \mathcal{L}_{\text{crossGraph}} + \mathcal{L}_{\text{pResolution}} + \lambda_1 \|\mathbf{W}\|_1 \tag{3.1}$$

$$\mathcal{L}_{\text{withinGraph}} = \sum_{p,q \in R} (w_{pq} - w^0_{pq})^2 \tag{3.2}$$

$$\mathcal{L}_{\text{crossGraph}} = \frac{1}{2} \sum_{p,q \in R} \sum_{\substack{(p',q') \in \\ N(p,q)}} \beta\Big(\pi(p), \big(\tau_1(p), \tau_2(p)\big), \big(\tau_1(p'), \tau_2(p')\big)\Big)(w_{pq} - w_{p'q'})^2 + \frac{\lambda_2}{2} \|\vec{1} - \vec{\beta}\|_2^2 \tag{3.3}$$

$$\mathcal{L}_{\text{pResolution}} = \frac{1}{2} \sum_{t_1,t_2 \in T} \sum_{\substack{p,q,r \in R(t_1,t_2) \\ r \neq p, r \neq q}} I_\varepsilon(w_{pq}) I_\varepsilon(w_{qp}) \left[(w_{pr} - w_{qr})^2 + (w_{rp} - w_{rq})^2\right] \tag{3.4}$$

Figure 3.2: The objective function to jointly learn global scores $\mathbf{W}$ and the compatibility function β, given local scores $\mathbf{W}^0$. $\mathcal{L}_{\text{withinGraph}}$ encourages global and local scores to be close; $\mathcal{L}_{\text{crossGraph}}$ encourages similarities to be consistent between different typed entailment graphs; $\mathcal{L}_{\text{pResolution}}$ encourages paraphrase relations to have the same pattern of entailment. We use an ℓ_1 regularization penalty to remove entailments with low confidence.

may in fact apply over *many* others (Figure 3.1A). For example, we may not have found direct evidence that mentions of a *living_thing* (e.g., a virus) *triggering* a *disease* are accompanied by mentions of the *living_thing causing* that *disease* (because of data sparsity), whereas we have found that mentions of a *government_agency triggering* an *event* are reliably accompanied by mentions of *causing* that *event*. While we show that typing is necessary to learning entailments (section 3.6), we propose to learn all typed entailment graphs jointly.

Second, we encourage paraphrase relations (where $p \rightarrow q$ and $q \rightarrow p$) to have the same patterns of entailment (Figure 3.1B), i.e. to entail and be entailed by the same relations, global soft constraints that we call *paraphrase resolution*. Using these soft constraint, a missing entailment (e.g., *medicine treats disease* $\rightarrow$ *medicine is useful for disease*) can be identified by considering the entailments of a paraphrase relation (e.g., *medicine cures disease* $\rightarrow$ *medicine is useful for disease*).

Sharing entailments across different typed entailment graphs is only semantically correct for some relations and types. In order to learn when we can generalize an entailment from one graph to another, we define a compatibility function $\beta : U \times (T \times T) \times (T \times T) \rightarrow [0,1]$. The function is defined for an untyped relation and two type pairs

(Figure 3.1A). It specifies the extent of compatibility for a single untyped relation *between different typed entailment graphs*, with 1 being completely compatible and 0 being irrelevant. In particular $\beta\big(u,(t_1,t_2),(t_1',t_2')\big)$ determines how much we expect the outgoing edges of $u(:t_1,:t_2)$ and $u(:t_1',:t_2')$ to be similar. We constrain β to be symmetric between t_1,t_2 and t_1',t_2' as compatibility of outgoing edges of $u(:t_1,:t_2)$ with $u(:t_1',:t_2')$ should be the same as $u(:t_1',:t_2')$ with $u(:t_1,:t_2)$. We denote by $\vec{\beta}$, a vectorization consisting of the values of β for all possible input untyped relations and types.

Note that the global similarity scores $\mathbf{W}$ and the compatibility vector $\vec{\beta}$ are not known in advance. Given local similarity scores $\mathbf{W}^0$, we learn $\mathbf{W}$ and $\vec{\beta}$ jointly by only relying on the set of extractions and without any external supervised data. We minimize the loss function defined in Equation 3.1 which consists of three soft constraints defined below and an ℓ_1 regularization term (Figure 3.2).

$\mathcal{L}_{\text{withinGraph}}$. Equation 3.2 encourages global scores w_{pq} to be close to local scores w_{pq}^0, so that the global scores will not stray too far from the original scores.

$\mathcal{L}_{\text{crossGraph}}$. Equation 3.3 encourages each relation's entailments to be similar across typed entailment graphs (Figure 3.1A) if the relations have similar neighbors. We penalize the difference of entailments in two different graphs, when the compatibility function is high. For each pair of typed relations $(p,q) \in R(t_1,t_2)$, we define a set of neighbors (relations with different types):

$$\begin{aligned} N(p,q) = \Big\{ & (p',q') \in R(t_1',t_2') | t_1',t_2' \in T, \\ & (p',q') \neq (p,q), \pi(p) = \pi(p'), \pi(q) = \pi(q'), \\ & a(p,q) = a(p',q') \Big\}, \end{aligned} \tag{3.5}$$

where $a(p,q)$ is true if the relation orders of p and q match, and false otherwise. For each $(p',q') \in N(p,q)$, we penalize the difference of entailments by adding the term $\beta(\cdot)(w_{pq} - w_{p'q'})^2$. We add a prior term on $\vec{\beta}$ as $\lambda_2 \|\vec{1} - \vec{\beta}\|_2^2$, where $\vec{1}$ is a vector of the same size as $\vec{\beta}$ with all 1s. Without the prior term (i.e., λ_2=0), all the elements of $\vec{\beta}$ will become zero. Increasing λ_2 will keep (some of the) elements of $\vec{\beta}$ non-zero and encourages communications between related graphs.

$\mathcal{L}_{\text{pResolution}}$. Equation 3.4 denotes the paraphrase resolution global soft constraints that encourage paraphrase relations to have the same patterns of entailments (Figure 3.1B). The function $I_\varepsilon(x)$ equals x if $x > \varepsilon$ and zero, otherwise.[6] Unlike $\mathcal{L}_{\text{crossGraph}}$ in

[6] In our experiments, we set $\varepsilon = .3$. Smaller values of ε yield similar results, but learning is slower.

Equation 3.3, Equation 3.4 operates on the edges within each graph. If both w_{pq} and w_{qp} are high, their incoming and outgoing edges from/to nodes r are encouraged to be similar. We name this global constraint, *paraphrase resolution*, since it might add missing links (e.g., $p \rightarrow r$) if p and q are paraphrases of each other and $q \rightarrow r$, or break the paraphrase relationship, if the incoming and outgoing edges are very different.

We impose an ℓ_1 penalty on the elements of $\mathbf{W}$ as $\lambda_1 \|\mathbf{W}\|_1$, where λ_1 is a non-negative tuning hyperparameter that controls the strength of the penalty applied to the elements of $\mathbf{W}$. This term removes entailments with low confidence from the entailment graphs. Note that Equation 3.1 has $\mathbf{W}^0$ and average of $\mathbf{W}^0$ across different typed entailment graphs (section 3.5.4) as its special cases. The former is achieved by setting $\lambda_1{=}\lambda_2{=}0$ and $\varepsilon{=}1$ and the latter by $\lambda_1{=}0$, $\lambda_2{=}\infty$ and $\varepsilon{=}1$. We do not explicitly weight the different components of the loss function, as the effect of $L_{\text{crossGraph}}$ and $L_{\text{pResolution}}$ can be controlled by λ_2 and ε, respectively.

Equation 3.1 can be interpreted as an inference problem in a Markov Random Field (MRF) (Kindermann and Snell, 1980), where the nodes of the MRF are the global scores w_{pq} and the parameters $\beta\big(u,(t_1,t_2),(t_1',t_2')\big)$. The MRF will have five log-linear factor types: one unary factor type for $\mathcal{L}_{\text{withinGraph}}$, one three-variable factor type for the first term of $\mathcal{L}_{\text{crossGraph}}$ and a unary factor type for the prior on $\vec{\beta}$, one four-variable factor type for $\mathcal{L}_{\text{pResolution}}$ and a unary factor type for the ℓ_1 regularization term. Figure 3.1 shows an example factor graph (unary factors are not shown for simplicity).

We learn $\mathbf{W}$ and $\vec{\beta}$ jointly using a message passing approach based on the Block Coordinate Descent method (Xu and Yin, 2013) . We initialize $\mathbf{W} = \mathbf{W}^0$. Assuming that we know the global similarity scores $\mathbf{W}$, we learn how much the entailments are compatible between different types ($\vec{\beta}$) and vice versa. Given $\mathbf{W}$ fixed, each w_{pq} sends messages to the corresponding $\beta(\cdot)$ elements, which will be used to update $\vec{\beta}$. Given $\vec{\beta}$ fixed, we do one iteration of learning for each w_{pq}. Each $\beta(\cdot)$ and w_{pq} elements send messages to the related elements in $\mathbf{W}$, which will be in turn updated.

Figure 3.3 shows the update rules of the learning algorithm. The global similarity scores w_{pq} are updated using Equation 3.6, where c_{pq} and η_{pq} are defined in Equation 3.7 and Equation 3.8, respectively. $\mathbb{1}(x)$ equals 1 if the condition x is satisfied and zero, otherwise. The compatibility functions $\beta(\cdot)$ are updated using Equation 3.9.

Based on the update rules, we always have $w_{pq} \leq 1$ and $\vec{\beta} \leq \vec{1}$. Each iteration of the learning method takes:

$$w_{pq} = \mathbb{1}(c_{pq} > \lambda_1)(c_{pq} - \lambda_1)/\eta_{pq} \tag{3.6}$$

$$\begin{aligned} c_{pq} = {} & w^0_{pq} + \sum_{(p',q') \in N(p,q)} \beta(\cdot) w_{p'q'} \\ & - \mathbb{1}(w_{pq} > \varepsilon) I_\varepsilon(w_{qp}) \sum_{r \in R(\tau_1(p), \tau_2(p))} \left[(w_{pr} - w_{qr})^2 + (w_{rp} - w_{rq})^2 \right] \\ & + 2 \sum_{r \in R(\tau_1(p), \tau_2(p))} I_\varepsilon(w_{qr}) I_\varepsilon(w_{rq}) w_{pr} + I_\varepsilon(w_{pr}) I_\varepsilon(w_{rp}) w_{rq} \end{aligned} \tag{3.7}$$

$$\eta_{pq} = 1 + \sum_{(p',q') \in N(p,q)} \beta(\cdot) + 2 \sum_{r \in R(\tau_1(p), \tau_2(p))} I_\varepsilon(w_{qr}) I_\varepsilon(w_{rq}) + I_\varepsilon(w_{pr}) I_\varepsilon(w_{rp}) \tag{3.8}$$

$$\beta(\cdot) = I_0\Big(1 - \Big(\sum_{q \in R(\tau_1(p), \tau_2(p))} \sum_{(p',q') \in N(p,q)} (w_{pq} - w_{p'q'})^2 \Big) / \lambda_2 \Big). \tag{3.9}$$

Figure 3.3: The update rules for w_{pq} and $\beta(\cdot)$. The global scores w_{pq} are initially set to w^0_{pq}. In each iteration, the compatibility values $\beta(\cdot)$ are computed given fixed values of w_{pq}. Then, the global scores w_{pq} are updated given fixed $\beta(\cdot)$.

$$O\big(\|\mathbf{W}\|_0 |T|^2 + \sum_{p \in R} (\|w_{p:}\|_0 + \|w_{:p}\|_0)^2\big)$$

time, where $\|W\|_0$ is the number of nonzero elements of W (number of edges in the current graph), $|T|$ is the number of types and $\|w_{p:}\|_0$ ($\|w_{:p}\|_0$) is the number of nonzero elements of the pth row (column) of the matrix (out-degree and in-degree of the node p). In our experiments, the total number of edges is $\approx .01|R|^2$ and most of relation pairs are seen in less than 20 subgraphs, instead of $|T|^2$. In practice, learning converges after 5 iterations of full updates. The method is highly parallelizable, and our efficient implementation does the learning in only a few hours.

3.5 Experimental Setup

We extract binary relations from a multiple-source news corpus (section 3.5.1) and compute local and global scores. We form entailment graphs based on the similarity scores and test our model on two entailment rules datasets (section 3.5.2). We then discuss parameter tuning (section 3.5.3) and baseline systems (section 3.5.4).

3.5.1 Training Corpus: Multiple-Source News

We use the multiple-source NewsSpike corpus of Zhang and Weld (2013). NewsSpike was deliberately built to include different articles from different sources describing identical news stories. They scraped RSS news feeds from January-February 2013 and linked them to full stories collected through a web search of the RSS titles. The corpus contains 550K news articles (20M sentences). Since this corpus contains multiple sources covering the same events, it is well-suited to our purpose of learning entailment and paraphrase relationships.

We extracted 29M triples using the procedure in Section 3.3.1. In our experiments, we used two cutoffs within each typed subgraph to reduce the effect of noise in the corpus: (1) remove any entity-pair that is observed with less than C_1=3 unique relations; (2) remove any relation that is observed with less than C_2=3 unique entity-pairs. This leaves us with $|U|$=101K unique untyped relations and $|R|$=304K unique typed relations in 346 entailment graphs. The maximum graph size is 53K nodes[7] and the total number of non-zero local scores in all graphs is 66M. In the future, we plan to test our method on an even larger corpus, but preliminary experiments suggest that data sparsity will persist regardless of the corpus size, due to the power law distribution of the terms. We compared our extractions qualitatively with Stanford Open IE (Etzioni et al., 2011; Angeli et al., 2015). Our CCG-based extraction generated noticeably better relations for longer sentences with long-range dependencies such as those involving coordination.

3.5.2 Evaluation Entailment Datasets

3.5.2.1 Levy/Holt's Entailment Dataset

Levy and Dagan (2016) proposed a new annotation method (and a new dataset) for collecting relational inference data in context. Their method removes a major bias in other inference datasets such as Zeichner's (Zeichner et al., 2012), where candidate entailments were selected using a directional similarity measure. Levy & Dagan form questions of the type *which city (q_{type}), is located near ($q_{relation}$), mountains (q_{entity})*? and provide possible answers of the form *Kyoto* (p_{answer}), *is surrounded by* ($p_{relation}$), *mountains (p_{entity})*. Annotators are shown a question with multiple possible answers, where p_{answer} is masked by q_{type} to reduce the bias towards world knowledge. If the

[7] There are 4 graphs with more than 20K nodes, 3 graphs with 10K to 20K nodes, and 16 graphs with 1K to 10K nodes.

annotator indicates the answer as *True* (*False*), it is interpreted that the relation in the answer *entails* (*does not entail*) the relation in the question.

While the Levy entailment dataset removes a major bias in previous datasets, a recent evaluation identified high labeling error rate for entailments that hold only in one direction (Holt, 2018). Holt analyzed 150 positive examples and showed that 33% of the claimed entailments are correct only in the *opposite* direction, while 15% do not entail in any direction. Holt (2018) designed a task to crowd-annotate the dataset by a) adding the reverse entailment ($q \rightarrow p$) for each original positive entailment ($p \rightarrow q$) in Levy's dataset; and b) directly asking the annotators if a positive example (or its reverse) is an entailment or not (as opposed to relying on a factoid question). We test our method on this re-annotated dataset of 18,407 examples (3,916 positive and 14,491 negative), which we refer to as Levy/Holt.[8] We run our CCG-based binary relation extraction on the examples. The parser might return multiple extractions given the sentence of a single example; however, when possible we select a parse with the same entities as in the example. We perform our typing procedure (section 3.3.2) on p_{answer} (e.g., *Kyoto*) and p_{entity} (e.g., *mountains*) to find the types of the entities. We split the re-annotated dataset into dev (30%) and test (70%) such that all the examples with the same q_{type} and $q_{relation}$ are assigned to only one of the sets.

3.5.2.2 Berant's Entailment Dataset

Berant et al. (2011) annotated all the edges of 10 typed entailment graphs based on the relations in their corpus. The dataset contains 3,427 edges (positive), and 35,585 non-edges (negative). We evaluate our method on all the examples of Berant's entailment dataset. The types of this dataset do not match with FIGER types, but we perform a simple hand-mapping between their types and FIGER types.[9]

3.5.3 Parameter Tuning

We selected λ_1=.01 and ε=.3 based on preliminary experiments on the dev set of Levy/Holt's dataset. We select λ_2 from $\{0, 0.01, 0.1, 0.5, 1, 1.5, 2, 10, \infty\}$.[10] We do not tune λ_2 for Berant's dataset. We instead use the selected value based on the Levy/Holt dev set. In all our experiments, we remove any local score $w^0_{pq} < .01$. We show precision-recall curves by changing the threshold δ on the similarity scores.

[8] www.github.com/xavi-ai/relational-implication-dataset

[9] 10 mappings in total (e.g., *animal* to *living_thing* and *place* to *location*).

[10] The selected value was usually around 1.5.

3.5.4 Comparison

We test our model by ablation of the global soft constraints $\mathcal{L}_{\text{crossGraph}}$ and $\mathcal{L}_{\text{pResolution}}$, testing simple baselines to resolve sparsity and comparing to the state-of-the-art resources. We also compare with two distributional approaches that can be used to predict relation similarity. We compare the following models and resources.

CG_PR is our novel model with both global soft constraints. **CG** is our model without $\mathcal{L}_{\text{pResolution}}$. **Local** is the local distributional similarities without any change.

AVG is the average of the local scores across all the entailment graphs that contain both relations in an entailment of interest. We set $\lambda_2 = \infty$ which forces all the values of $\vec{\beta}$ to be 1, hence resulting in a uniform average of local scores. **Untyped** scores are local scores learned without types. We set the cutoffs C_1=20 and C_2=20 to have a graph with total number of edges similar to the typed entailment graphs.

ConvE scores are cosine similarities of low-dimensional relation representations learned by ConvE (Dettmers et al., 2018), a state-of-the-art model for link prediction. ConvE is a multi-layer convolutional network model that is highly parameter efficient (section 2.2). We learn 200-dimensional vectors for each relation (and entity) by applying ConvE to the set of extractions of the above untyped graph. We learned embeddings for each relation and its reverse to handle examples where the entity order of the two relations are different. Additionally, we tried TransE (Bordes et al., 2013), another link prediction method which despite its simplicity, produces very competitive results in knowledge base completion. However, we do not present its full results as they were worse than ConvE.[11]

PPDB is based on the Paraphrase Database (PPDB) of Pavlick et al. (2015). We accept an example as entailment if it is labeled as a paraphrase or entailment in the PPDB XL lexical or phrasal collections.[12] **Berant_ILP** is based on the entailment graphs of Berant et al. (2011).[13] For Berant's dataset, we directly compared our results to the ones reported in Berant et al. (2011). For Levy/Holt's dataset, we used publicly available entailment rules derived from Berant et al. (2011) that gives us one point of precision and recall in the plots. While the rules are typed and can be applied in

[11] We also tried the average of GloVe embeddings (Pennington et al., 2014) of the words in each relation, but the results were worse than ConvE.

[12] We also tested the largest collection (XXXL) , but the precision was very low on Berant's dataset (below 30%).

[13] We also tested the graphs from Berant et al. (2015), but do not report the results as they are very similar.

a context sensitive manner, ignoring the types and applying the rules out of context yields much better results (Levy and Dagan, 2016). This is attributable to both the non-standard types used by Berant et al. (2011) and also the general data sparsity issue.

In all our experiments, we first test a set of rule-based constraints introduced by Berant et al. (2011) on the examples before the prediction by our methods (section 2.1.2.1). In the experiments on Levy/Holt's dataset, in order to maintain compatibility with Levy and Dagan (2016), we also run the lemma based heuristic process used by them before applying our methods. If an example is labeled as entailment by the lemma baseline, we label it as positive. After lemmatizing the premise relation ($p_{relation}$) and the hypothesis relation ($q_{relation}$), the baseline classifies an example as positive if four conditions are satisfied: (1) $p_{relation}$ contains all of $q_{relation}$'s content words, (2) The two relations share a verb, (3) The relations' active/passive voice match the alignments of their entities and (4) The relations agree on negation. We do not apply the lemma based process on Berant's dataset in order to compare with Berant et al's (2011) reported results directly. In experiments with CG_PR and CG, if the typed entailment graph corresponding to an example does not have one or both relations, we resort to the average score between all typed entailment graphs.

3.6 Results and Discussion

To test the efficacy of our globally consistent entailment graphs, we compare them with the baseline systems in Section 3.6.1. We test the effect of approximating transitivity constraints in Section 3.6.2. Section 3.6.3 concerns error analysis.

3.6.1 Globally Consistent Entailment Graphs

We test our method using three distributional similarity measures: Weeds similarity (Weeds and Weir, 2003), Lin similarity (Lin, 1998) and Balanced Inclusion (BInc; Szpektor and Dagan, 2008). The first two similarity measures are symmetric,[14] while BInc is directional. Figures 3.4A and 3.4B show precision-recall curves of the different methods on Levy/Holt's and Berant's datasets, respectively, using BInc. We show the full curve for BInc as it is directional and on the development portion of Levy/Holt's dataset, it yields better results than Weeds and Lin.

[14] Weeds similarity is the harmonic average of Weeds precision and Weeds recall, hence a symmetric measure.

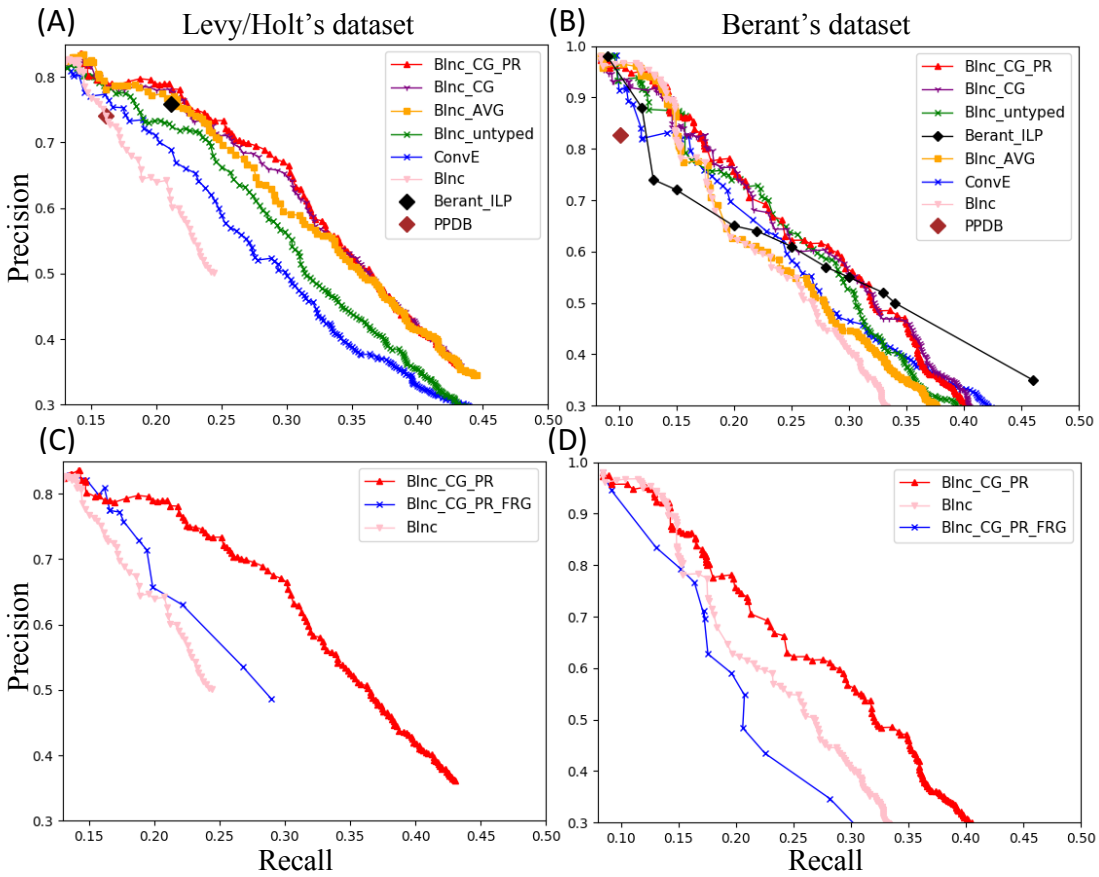


Figure 3.4: Comparison of globally consistent entailment graphs to the baselines on Levy/Holt's (A) and Berant's (B) datasets. The results are compared to graphs learned by Forest-Reducible Graph Assumption on Levy/Holts's (C) and Berant's (D) datasets.

In addition, Table 3.2 shows the area under the precision-recall curve (AUC) for all variants of the three similarity measures. Note that each method covers a different range of precisions and recalls. We compute AUC for precisions in the range $[0.5, 1]$, because all methods cover this range and predictions with precision better than random guess are more important for end applications such as question-answering and semantic parsing. For each similarity measure, we tested statistical significance between the methods using bootstrap resampling with 10K experiments (Efron and Tibshirani, 1985; Koehn, 2004). In Table 3.2, the best result for each dataset and similarity measure is boldfaced. If the difference of another model with the best result is not significantly different with p-value $< .05$, the second model is also boldfaced.

Among the distributional similarities based on BInc, BInc_CG_PR outperforms all the other models in both datasets. In comparison to BInc score's AUC, we observe more than 100% improvement on Levy/Holt's dataset and about 30% improvement on

	local	untyped	AVG	CG	CG_PR
	LEVY/HOLT'S dataset				
BInc	.076	.127	.157	.162	**.165**
Lin	.074	.120	.146	**.151**	**.149**
Weed	.073	.115	.143	**.149**	**.147**
ConvE	-	.112	-	-	-
	BERANT'S dataset				
BInc	.138	.167	.144	**.177**	**.179**
Lin	.147	.158	.172	.186	**.189**
Weed	.146	.154	.171	.184	**.187**
ConvE	-	.144	-	-	-

Table 3.2: Area under precision-recall curve (for precision > 0.5) for different variants of similarity measures: local, untyped, AVG, crossGraph (CG) and crossGraph + pResolution (CG_PR). We report results on two datasets. Bold indicates stat significance (see text).

Berant's. Given the consistent gains, our proposed model appears to alleviate the data sparsity and the noise inherent to local scores. Our method also outperforms PPDB and Berant_ILP on both datasets. The second best performing model is BInc_CG, which improves the results significantly over the BInc_AVG. This confirms that learning what subset of entailments should be generalized across different typed entailment graphs (i.e., $\vec{\beta}$) is effective.

The untyped models yield a single large entailment graph. They contain (noisy) edges that are not found in smaller typed entailment graphs. For example, our corpus does not have many relations with types *medicine* or *disease*. Therefore, many entailments with those types might not be directly found in the typed entailment graphs. Despite the noise, untyped models for all three similarity measures still perform better than the typed ones in terms of AUC. However, they do worse in the high-precision range. For example, BInc_untyped is worse than BInc for precision > 0.85. The AVG models do surprisingly well (only about 0.5 to 3.5 below CG_PR in terms of AUC), but note that only a subset of the typed entailment graphs might have relations p and q of interest (usually not more than 10 typed entailment graphs out of 346 graphs). Therefore, the AVG models are generally expected to outperform the untyped ones, as typing has refined the entailments and averaging just improves the recall.

Comparison of CG_PR with CG models confirms that explicitly encouraging para-

phrase relations to have the same patterns of entailment is effective. It improves the results for BInc score, which is a directional similarity measure. We also tested applying the paraphrase resolution soft constraints alone, but the differences with the local scores were not statistically significant. This suggests that the paraphrase resolution is more helpful when similarities are transferred between graphs, as this can cause inconsistencies around the relations with transferred similarities, which are then resolved by the paraphrase resolution constraints.

The results of the distributional representations learned by ConvE are worse than most other methods. We attribute this outcome to the fact that a) while entailment rules are directional, these methods are symmetric; b) the learned embeddings are optimized for tasks other than entailment or paraphrase detection; and c) the embeddings are learned regardless of entity types. However, even the BInc_untyped baseline outperforms ConvE, showing that it is important to use a directional measure that directly models entailment. In chapter 4, we propose a model that does not have the the above limitations and show improved results.

3.6.2 Effect of Transitivity Constraints

Our largest graph has 53K nodes, we thus tested approximate methods instead of the ILP to close entailment rules under transitivity (section 3.2). The approximate TNF method of Berant et al. (2011) did not scale to the size of our graphs with moderate sparsity parameters. Berant et al. (2015) also present a heuristic method, High-To-Low Forest-Reducible Graph (HTL-FRG), which gets slightly better results than TNF on their dataset, and which scales to graphs of the size we work with.[15]

We applied the HTL-FRG method to the globally consistent similarity scores, i.e., BInc_CG_PR (shown as BInc_CG_PR_HTL) and changed the threshold on the scores to get a precision-recall curve. Figures 3.4C and 3.4D show the results of this method on Levy/Holt's and Berant's datasets. Our experiments show, in contrast to the results of Berant et al. (2015), that the HTL-FRG method leads to worse results when applied to our global scores. This result is caused because a) The use of heuristic methods in place of globally optimizing via ILP leads to suboptimal edge selection. Especially, committing to wrong edges at the early iterations of the HTL-FRG method could prevent the addition of many correct edges in the future iterations; and b) Many valid

[15]TNF did not converge after two weeks for threshold $\delta = .04$. For $\delta = .12$ (precisions higher than 80%), it converged, but with results slightly worse than HTL-FRG on both datasets.

Error type	Example
False Positive	
Spurious correlation (57%)	Microsoft **released** Internet Explorer → Internet Explorer **was developed by** Microsoft
Relation normalization (31%)	The pain **may be** relieved by aspirin → The pain can be treated with aspirin
Lemma based process & parsing (12%)	President Kennedy **came to** Texas → President Kennedy **came from** Texas
False Negative	
Sparsity (93%)	Cape town **lies at the foot of** mountains → Cape town is located near mountains
Wrong label & parsing (7%)	Horses are imported from Australia → Horses are native to Australia

Table 3.3: Examples of different error categories and their relative frequencies computed over 100 randomly selected false positive and false negative examples. The cause of errors is **boldfaced**.

edges could be removed since the FRG assumption is not correct for many real-world domains.

3.6.3 Error Analysis

We analyzed 100 false positive (FP) and 100 false negative (FN) randomly selected examples (using BInc_CG_ST results on Levy/Holt's dataset and at the precision level of Berant_ILP, i.e. 0.76). We present our findings in Table 3.3. Most of the FN errors are due to data sparsity, but a few errors are due to wrong labeling of the data and parsing errors. More than half of the FP errors are because of spurious correlations in the data that are captured by the similarity scores, but are not judged to constitute entailment by the human judges. About one third of the FP errors are because of the normalization we currently perform on the relations, e.g., we remove modals and auxiliaries. The remaining errors are mostly due to parsing and our use of Levy and Dagan's (2016) lemma based heuristic process.

Sentence Containing the Answer	Question
The board **hailed Romney** for his solid credentials.	Who **praised** Mitt **Romney**'s credentials?
Researchers announced this week that they've **found** a new **gene**, ALS6, which is responsible for ...	Which **gene** did the ALS association **discover** ?
One out of every 17 children under 3 years old in America **has** a **food allergy**, and some will outgrow their sensitivities.	How many Americans **suffer from food allergies**?
The reported compromise could itself **run afoul of** European labor **law**, opening the way for foreign workers ...	What **law** might the **deal break**?
... Barnes & Noble CEO William Lynch said as he **unveiled** his company 's **Nook Tablet** on Monday.	Who **launched** the **Nook Tablet**?
The report said opium has **accounted for** more than half of **Afghanistan 's gross domestic product** in 2007.	What **makes up** half of **Afghanistan's GDP**?

Table 3.4: Examples where explicit entailment rules improve the rankings. Correct sentences are selected because of the entailment rules from the relations in the answers (left column) to the relations in the questions (right column). The related words are **boldfaced**.

3.7 Extrinsic Evaluation

To further test the utility of explicit entailment rules, we evaluate the learned rules on an extrinsic task: answer selection for machine reading comprehension on NewsQA, a dataset that contains questions about CNN articles (Trischler et al., 2017). Machine reading comprehension is usually evaluated by posing questions about a text passage and then assessing the answers of a system (Trischler et al., 2017). The datasets that are used for this task are often in the form of (document,question,answer) triples, where answer is a short span of the document. Answer selection is an important task where the goal is to select the sentence(s) that contain the answer. We show improvements by adding knowledge from our learned entailments *without changing the graphs or tuning*

them to this task in any way.

Inverse sentence frequency (ISF) is a strong baseline for answer selection (Trischler et al., 2017). The ISF score between a sentence S_i and a question Q is defined as $\text{ISF}(S_i, Q) = \sum_{w \in S_i \cap Q} \text{IDF}(w)$, where $\text{IDF}(w)$ is the inverse document frequency of the word w by considering each sentence in the whole corpus as one document. The state-of-the-art methods for answer selection use ISF and by itself it already does quite well (Trischler et al., 2017; Narayan et al., 2018). We propose to extend the ISF score with entailment rules. We define a new score

$$\begin{aligned}\text{ISFEnt}(S_i, Q) &= \alpha \text{ISF}(S_i, Q) \\ &+ (1-\alpha)|\{p \in S_i, q \in Q : p \rightarrow q\}|,\end{aligned}$$

where $\alpha \in [0,1]$ is a hyper-parameter and p and q denote relations in the sentence and the question, respectively. The intuition is that if a sentence such as *"Luka Modric sustained a fracture to his right fibula"* is a paraphrase of or entails the answer of a question such as *"What does Luka Modric suffer from?"*, it will contain the answer span. We consider an entailment decision between two typed relations if their global similarity BInc_CG_PR is higher than a threshold δ.

We also considered entailments between unary relations (one entity) by leveraging our learned binary entailments. We split each binary entailment into two potential unary entailments. For example, the entailment $visit_{1,2}$*(:person,:location)* $\rightarrow$ $arrive_{1,in}$*(:person,:location)*, is split into $visit_1$*(: person)* $\rightarrow$ $arrive_1$*(: person)* and $visit_2$*(: location)* $\rightarrow$ $arrive_{in}$*(: location)*. We computed unary similarity scores by averaging over all related binary scores. This is particularly helpful when one entity is not present (e.g., adjuncts or *Wh* questions) or does not exactly match between the question and the answer.

We test the proposed answer selection score on NewsQA, a dataset that contains questions about CNN articles (Trischler et al., 2017). The dataset is collected in a way that encourages lexical and syntactic divergence between questions and documents. The crowdworkers who wrote questions saw only a news article headline and its summary points, but not the full article. This process encourages curiosity about the contents of the full article and prevents questions that are simple reformulations of article sentences (Trischler et al., 2017). This is a more realistic and suitable setting to test paraphrasing and entailment capabilities.

	ACC	MRR	MAP
ISF	36.18	48.99	48.57
ISFEnt	**37.61**	**50.06**	**49.63**

Table 3.5: Results (in percentage) for answer selection on the NewsQA dataset. We compare the original score (ISF) with the entailment-based score (ISFEnt).

We use the development set of the dataset (5165 samples) to tune α and δ [16] and report results on the test set (5124 examples) in Table 3.5. We observe about 1.4% improvement in accuracy (ACC) and 1% improvement in Mean Reciprocal Rank (MRR) and Mean Average Precision (MAP), confirming that entailment rules are helpful for answer selection.[17]

One of the main advantages of relational entailment graphs is their interpretability. In the sentence selection experiment, it is possible to explain why a correct sentence gets a high score when we use the entailment graphs. Table 3.4 shows some of the examples where ISFEnt ranks the correct sentences higher than ISF. These examples are very challenging for methods that do not have entailment and paraphrasing knowledge, and illustrate the semantic interpretability of the entailment graphs.

We also performed a similar evaluation on the Stanford Natural Language Inference dataset (SNLI; Bowman et al., 2015) and obtained 1% improvement over a basic neural network architecture that models sentences with an n-layered LSTM (Conneau et al., 2017). However, we did not get improvements over the state of the art results because only a few of the SNLI examples require external knowledge of relation entailments. Most examples require reasoning capabilities such as $A \wedge B \rightarrow B$ and simple lexical entailments such as *boy* $\rightarrow$ *person*, which are often present in the training set.

3.8 Conclusions

We have introduced a scalable framework to learn typed entailment graphs directly from text. We use global soft constraints to learn globally consistent entailment scores between relations. Our experiments show that generalizing in this way across differ-

[16] The selected hyper-parameters are $\alpha = .2$ and $\delta = .003$.

[17] The accuracy results of Narayan et al. (2018) are not consistent with their own MRR and MAP (ACC>MRR in come cases), as they break ties between ISF scores differently when computing ACC compared to MRR and MAP. See also `http://homepages.inf.ed.ac.uk/scohen/acl18external-errata.pdf`.

ent but related typed entailment graphs significantly improves performance over local similarity scores on two standard text-entailment datasets. We show around 100% increase in AUC on Levy/Holt's dataset and 30% on Berant's dataset. The method also outperforms PPDB and the prior state-of-the-art entailment graph-building approach due to Berant et al. (2011). Paraphrase Resolution further improves the results. We have in addition showed the utility of entailment rules on answer selection for machine reading comprehension.

Chapter 4

Duality of Link Prediction and Entailment Graph Induction

Link prediction and entailment graph induction are often treated as different problems. In this chapter, we show that these two problems are actually complementary. We train a link prediction model on a knowledge graph of assertions extracted from raw text. We propose an entailment score that exploits the new facts discovered by the link prediction model, and then form entailment graphs between relations. We further use the learned entailments to predict improved link prediction scores. Our results show that the two tasks can benefit from each other. The new entailment score outperforms prior state-of-the-art results on a standard entialment dataset and the new link prediction scores show improvements over the raw link prediction scores.

4.1 Introduction

Link prediction and entailment graph induction are often treated as different problems. The former (Figure 4.1A) is used to infer missing relations between entities in existing knowledge graphs (Socher et al., 2013; Bordes et al., 2013; Riedel et al., 2013). The latter (Figure 4.1B) constructs entailment graphs with relations as nodes and entailment rules as edges between them (Berant et al., 2011, 2015) for the task of answering questions from text. In this chapter, we show that these two problems are complementary by demonstrating how link prediction can help identify entailments and how discovered entailments can help predict missing links.

Methods to learn entailment graphs such as the work of Berant et al. (2011, 2015) and our work in chapter 3 process large text corpora to find *local* entailment scores

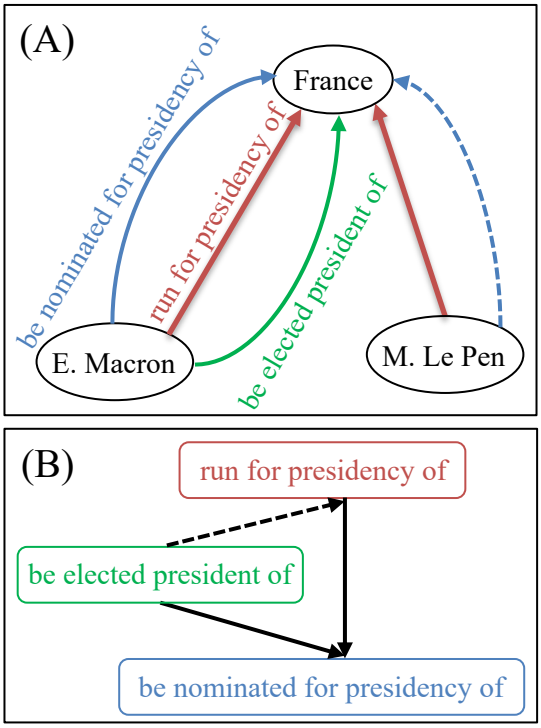


Figure 4.1: A link prediction knowledge graph (A) and an entailment graph (B) for entities of types *politician,country*. The solid lines are discovered correctly, but the dashed ones are missing. However, evidence from the link prediction model can be used to add the missing entailment rule in the entailment graph (B). Similarly, the entailment graph can be used to add the missing link in the knowledge graph (A).

between relations based on the Distributional Inclusion Hypothesis. They use types such as *person*, *location* and *time*, to disambiguate polysemous relations (e.g., *person born in location* and *person born in time*). Entailment graphs are then formed by imposing *global* constraints such as transitivity of the entailments proposed in (Berant et al., 2011) or global soft constraints proposed in chapter 3. The paraphrase[1] and entailment rules provide an interpretable resource that can be used to answer questions, when the answer is not explicitly stated in the text. For example, while we can find on the web the assertion *Loch Fyne lies at the foot of mountains*, we cannot find a sentence directly stating that *Loch Fyne is located near mountains* by querying Google as of 4th March 2019. Knowledge of the entailment rule between *lies at the foot of* and *is located near* can be used to answer such questions.

On the other hand, link prediction (or knowledge base completion) models are based on distributional methods and directly predict the source data. These models

[1]Relations that entail each other in both directions are regarded as paraphrases.

have received much attention in the recent years (Socher et al., 2013; Bordes et al., 2013; Riedel et al., 2013; Toutanova et al., 2016; Trouillon et al., 2016; Dettmers et al., 2018; Kazemi and Poole, 2018; Balazevic et al., 2019). The current methods learn embeddings for all entities and relations and a function to score any potential relation between the entities. One of the main capabilities of these models is that they implicitly exploit entailment rules such as *person born in country* entails *person be from country* (Riedel et al., 2013). However, entailment rules are not learned explicitly. For example, we cannot simply compute the cosine similarity of the vector representations of the two relations to detect the entailment between them, because cosine similarity is symmetric (section 4.5.1). These methods are usually applied to augment existing knowledge graphs such as Freebase (Bollacker et al., 2008), DBPedia (Lehmann et al., 2015) and Yago (Suchanek et al., 2007), but they can also be applied to assertions extracted from raw text.

In this chapter, we explore the synergies between the two tasks. Current entailment graphs suffer from sparsity and noise in the data. The link prediction methods discover new facts (triples) that can be used to alleviate the sparsity issue. In addition, they can remove noise by filtering facts that are not consistent with the other facts. We propose a new entailment score based on link prediction (section 4.3.1) which significantly improves over the previous state-of-the-art (our results in chapter 3) on a standard entailment detection dataset (4.5.1). For example, our method can discover that *be elected president of* entails *run for presidency of* by relying on the predicted links concerning the two relations (Figure 4.1). We use the new entailment score to compute local scores between the relations. We then use the local scores to build global entailment graphs using the global soft constraints discussed in chapter 3.

In addition, we show that the discovered entailments can be used to predict links in knowledge graphs (section 4.3.2). For example, knowing that *run for presidency of* entails *be nominated for presidency of* as well as the assertion *Le Pen ran for presidency of France*, we can infer that she also *was nominated for presidency of France*. In our experiments, we show improvements over a state-of-the-art link prediction model (section 4.4.2).[2]

The structure of this chapter is as follows. We discuss the background and introduce the notation in section 4.2 and propose our novel method in section 4.3. We describe the experimental setup in section 4.4 and discuss the results in section 4.5. We outline the related work in section 4.6 and conclude the chapter in section 4.7.

[2]Our code and data are available at `https://github.com/mjhosseini/linkpred_entgraph`.

4.2 Background and Notation

Let T denote the set of all types (e.g., *politician*), $\mathcal{E}(t)$ the set of entities with type t (e.g., *E. Macron*) and $R(t_1,t_2)$ the set of relations with types (t_1,t_2) or (t_2,t_1) (e.g., *be elected president of*). We denote by $\mathcal{E} = \bigcup_t \mathcal{E}(t)$ the set of all entities and by $R = \bigcup_{t_1,t_2} R(t_1,t_2)$ the set of all relations. Denote by $\mathcal{K}(t_1,t_2)$ the knowledge graph consisting of a set of correct triples (p,e_1,e_2), where $p \in R(t_1,t_2)$, $(e_1,e_2) \in \mathcal{E}^2(t_1,t_2)$ and $\mathcal{E}^2(t_1,t_2) = \big(\mathcal{E}(t_1) \times \mathcal{E}(t_2)\big) \cup \big(\mathcal{E}(t_2) \times \mathcal{E}(t_1)\big)$. We define $\mathcal{E}^2 = \bigcup_{t_1,t_2} \mathcal{E}^2(t_1,t_2)$ the set of all possible entity pairs. We denote by $\mathcal{K} = \bigcup_{t_1,t_2} \mathcal{K}(t_1,t_2)$ the knowledge graph consisting of all types. In practice, we have not observed all the correct triples, but instead have access to a noisy and incomplete knowledge graph. We define by X_{p,e_1,e_2} a binary random variable which is 1 if (p,e_1,e_2) is in the knowledge graph and 0, otherwise.

In the rest of this section, we introduce the problems of link prediction (section 4.2.1) and detecting entailment rules (section 4.2.2).

4.2.1 Link Prediction

As discussed in section 2.2, for each triple (p,e_1,e_2), a link prediction model defines a scoring function $f(p,e_1,e_2)$ of its plausibility (Socher et al., 2013; Bordes et al., 2013; Riedel et al., 2013; Toutanova et al., 2016; Trouillon et al., 2016; Dettmers et al., 2018). We use ConvE (Dettmers et al., 2018), a state-of-the-art and efficient model, in our experiments. The models then choose f such that the score $f(p,e_1,e_2)$ of a plausible triple $(p,e_1,e_2) \in \mathcal{K}$ is higher than the score $f(p',e_1',e_2')$ of an implausible triple $(p',e_1',e_2') \notin \mathcal{K}$ (Nguyen, 2020). The plausibility score $f(p,e_1,e_2)$ can optionally be mapped into a probability score S_{p,e_1,e_2}.[3] The probability score S_{p,e_1,e_2} is an estimate of $\mathrm{P}(X_{p,e_1,e_2}{=}1)$, i.e., the probability of the triple being correct. We denote by $S \in [0,1]^{|R|\times|\mathcal{E}^2|}$ the matrix containing triple probability scores. We define $S(t_1,t_2) \in [0,1]^{|R(t_1,t_2)|\times|\mathcal{E}^2(t_1,t_2)|}$ the submatrix of S with $R(t_1,t_2)$ as rows and $\mathcal{E}^2(t_1,t_2)$ as columns. We apply a link prediction model to a knowledge graph of triples extracted from text (section 4.4.2).

[3] For example by applying the Sigmoid function.

4.2.2 Entailment Prediction

The goal is to find entailment scores between all relations with the same types, where the entities can be in the same or opposite order (Berant et al., 2011; Lewis and Steedman, 2014b). We denote by $\mathbf{W}(t_1,t_2) \in [0,1]^{|R(t_1,t_2)|\times|R(t_1,t_2)|}$ the (sparse) matrix containing all similarity scores w_{pq} between relations $p,q \in R(t_1,t_2)$. We define $\mathbf{W}$ the (block diagonal) matrix consisting of all the similarity matrices $\mathbf{W}(t_1,t_2)$. For a $\delta > 0$, we define typed entailment graphs as $G_\delta(t_1,t_2) = \big(R(t_1,t_2), E_\delta(t_1,t_2)\big)$, where $R(t_1,t_2)$ are the nodes and $E_\delta(t_1,t_2) = \{(p,q)|p,q \in R(t_1,t_2), w_{pq} \geq \delta\}$ are the edges of the entailment graphs.

Existing entailment similarity measures for relational entailment detection such as Weeds (Weeds and Weir, 2003), Lin (Lin, 1998), and Balanced Inclusion (BInc; Szpektor and Dagan, 2008) are typically defined on feature vectors consisting of entity-pairs (e.g., *Obama-Hawaii*), where the values are frequencies or pointwise mutual information (PMI) between the relations and the features (Berant et al., 2011, 2012, 2015). While these methods had previous state-of-the-art results on relation entailment datasets (our work in chapter 3), they suffer from low recall because the feature vectors are usually sparse and do not have high overlap with each other. The link prediction models, on the other hand, can predict the probability of any triple being in the knowledge graph. Using predicted probability scores can hugely alleviate the sparsity problem by increasing the overlap between feature vectors (section 4.3.1).

4.3 Duality between Entailment Scores and Link Prediction

We discuss the relationship between link prediction scores $S(t_1,t_2)$ and entailment scores $\mathbf{W}(t_1,t_2)$. We claim that while these two tasks are usually treated separately, they are complementary. We propose a method to predict entailment scores by using link prediction scores. The proposed score estimates the probability of relations given one another. It exploits the strength of the link prediction models, i.e., predicting new facts as well as removing noise from the existing ones (section 4.3.1). We further show how we can improve link prediction scores by using predicted entailment scores. Having access to an entailment rule $p \rightarrow q$, we use the link prediction scores of p to refine the scores of q for any entity pairs (section 4.3.2). All the methods in this section are applied for each type pair separately; however, in the rest of this chapter, we drop

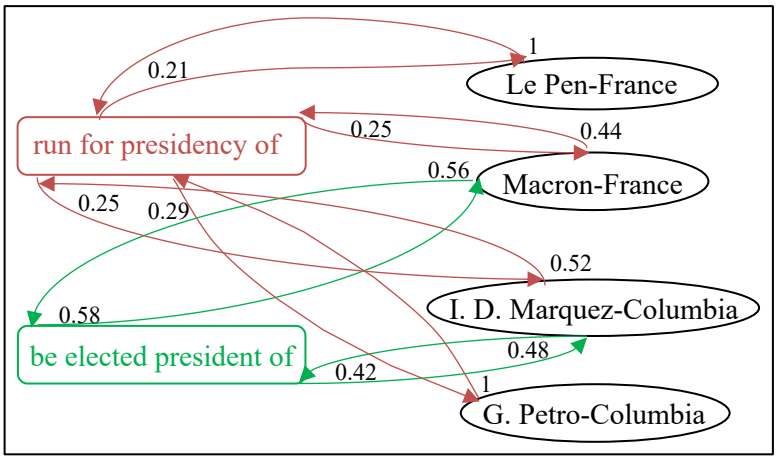


Figure 4.2: A small Markov chain with two relations (squares) and four entity-pairs (ovals). Directed edges connect each relation to its related entity-pairs, and vice versa. Transition probabilities are shown on each edge. The outgoing probabilities from each node sum to 1.

(t_1, t_2) for simplicity of the notation.

4.3.1 Entailment Scores From Link Prediction

In this section, we show how we can use link prediction scores to predict entailment scores. In order to compute the entailment scores, we apply a link prediction method on the knowledge graph $\mathcal{K}$. We define a new entailment score based on link prediction scores.

More specifically, We reform the knowledge graph representation into a Markov chain over a bipartite graph $M = (V_M, E_M)$, where $V_M = R \cup \mathcal{E}^2$ are the nodes of the graph, and E_M contains edges $(\langle p \rangle, \langle e_1, e_2 \rangle)$ and $(\langle e_1, e_2 \rangle, \langle p \rangle)$ iff $\mathrm{P}(X_{p,e_1,e_2}{=}1) > 0$. Figure 4.2 shows an example Markov chain with only two relations and four entity-pairs. The transition probabilities of the chain are:

$$\mathrm{P}(\langle e_1, e_2 \rangle \mid \langle p \rangle) = \frac{\mathrm{P}(X_{p,e_1,e_2}{=}1)}{\sum_{e_1,e_2 \in \mathcal{E}^2} \mathrm{P}(X_{p,e_1,e_2}{=}1)}$$

$$\mathrm{P}(\langle p \rangle \mid \langle e_1, e_2 \rangle) = \frac{\mathrm{P}(X_{p,e_1,e_2}{=}1)}{\sum_{r \in R} \mathrm{P}(X_{r,e_1,e_2}{=}1)}$$

For relations p and q, we define the entailment score $w_{pq} = \mathrm{P}(\langle q \rangle | \langle p \rangle)$, where we compute the probability by considering only the paths of length 2 between p and q that

pass through one entity-pair node.[4] We define:

$$\mathrm{P}(\langle q\rangle|\langle p\rangle) = \sum_{e_1,e_2\in\mathcal{E}^2} \mathrm{P}(\langle q\rangle|\langle e_1,e_2\rangle)\,\mathrm{P}(\langle e_1,e_2\rangle|\langle p\rangle). \tag{4.1}$$

We use S_{p,e_1,e_2} from the link prediction model as an estimate of $\mathrm{P}(X_{p,e_1,e_2}{=}1)$ to compute Equation 4.1. We can compute the scores for all $p,q \in R$ efficiently, by normalizing each row of the matrices S and $S^\top$ and multiplying them.[5] Note that building the matrix S for all possible triples make the computation of the scores intractable, especially for large number of relations (section 4.4.1). In our experiments, we consider any (p,e_1,e_2) seen in the corpus. In addition, we add a subset of high scoring triples not seen in the corpus (section 4.4.2).

4.3.2 Improving Link Prediction Scores using Entailment Scores

In the previous section, we demonstrated how we can use link prediction methods to learn entailment scores. In this section, we consider the inverse problem, i.e., we use the predicted entailment rules to improve link prediction scores. We assume the Distributional Inclusion Hypothesis (DIH) which states that a word (relation) p entails another word (relation) q if and only if in any context that p can be used, q can be used in its place (Dagan et al., 1999; Geffet and Dagan, 2005; Kartsaklis and Sadrzadeh, 2016). In particular, in a correct and complete knowledge graph, we have:

$$\begin{aligned} & p{\to}q \implies \forall(e_1,e_2)\in\mathcal{E}^2: \\ & X_{p,e_1,e_2} = 1 \to X_{q,e_1,e_2} = 1 \\ & \implies X_{p,e_1,e_2} \le X_{q,e_1,e_2}. \end{aligned} \tag{4.2}$$

Therefore when $p \to q$, it is reasonable to assume $\mathrm{P}(X_{p,e_1,e_2}=1) \le \mathrm{P}(X_{q,e_1,e_2}=1)$ for all entity pairs e_1,e_2. This would suggest we can define a new link prediction score based on entailment rules:

$$S^{ent}_{q,e_1,e_2} = \max_{p\in R:\, p\to q} S_{p,e_1,e_2}. \tag{4.3}$$

[4] Longer paths did not yield better performance in our experiments while increasing the memory and running time requirements.

[5] An alternative approach would be based on sampling paths over the Markov chain, but we compute the exact solution by performing matrix multiplication.

However, since we do not have access to the entailment rules and can only rely on the predictions, Equation 4.3 is likely to be very noisy. We smooth Equation 4.3 by using a weighted average of the scores of each entailment rule. We define:

$$S^{ent}_{q,e_1,e_2} = \max\left(S_{q,e_1,e_2}, \sum_{p \in R} w'_{pq} S_{p,e_1,e_2}\right),$$

where w'_{pq} is defined by normalizing the qth column of the matrix $\mathbf{W}$.

$$w'_{pq} = \frac{w_{pq}}{\sum_{p':p' \to q} w_{p'q}}.$$

4.4 Experimental Setup

In this section, we discuss the details of our experiments. We first describe the text corpus and extracted triples which are used as the input to our method (section 4.4.1). We then describe the details of the link prediction model (section 4.4.2), the datasets used to test the models (section 4.4.3) and the baseline systems (section 4.4.4).

4.4.1 Text Corpus

Link prediction models are often applied to existing knowledge graphs such as Freebase (Bollacker et al., 2008), DBPedia (Lehmann et al., 2015) and Yago (Suchanek et al., 2007); however, we chose to experiment on assertions extracted from raw text. This is because we can then evaluate the predicted entailments on existing entailment datasets with examples stated in natural language (section 4.4.3).

Similar to the experiments in chapter 3, We use the multiple-source NewsSpike corpus of Zhang and Weld (2013). As a reminder, the NewsSpike corpus includes 550K news articles and is well-suited for finding entailment and paraphrasing relations as it includes different articles from different sources describing identical news stories. We use the triples extracted in chapter 3 which were obtained by running the semantic parser of Reddy et al. (2014), GraphParser, to extract binary relations between a predicate and its arguments. GraphParser uses Combinatory Categorial Grammar (CCG) syntactic derivations by running EasyCCG (Lewis and Steedman, 2014a). The parser converts sentences to neo-Davisonian semantics, a first order logic that uses event identifiers and extracts one binary relation for each event and pair of arguments (Parsons, 1990). The entities are typed by first linking to Freebase (Bollacker et al., 2008) and then selecting the most notable type of the entity from Freebase and mapping it to

FIGER types (Ling and Weld, 2012) such as *building*, *disease* and *person*. We use the first level of the FIGER types hierarchy to assign one of the 49 types (out of 113 total types) to the entities.

The extractions contain 29M unique triples. Similar to chapter 3, we filter any relation that is seen with less than three unique entity-pairs, and any entity-pairs that is seen with less than three unique relations. The filtered corpus has 3.9M relations covering $|R|$=304K typed relations (101K untyped relations).

4.4.2 Link Prediction

We randomly split the corpus into training (95%), validation (4%) and test (1%) sets. We train the link prediction model on the training set and use the validation set for parameter tuning. We apply ConvE (Dettmers et al., 2018)[6], a state-of-the-art model for link prediction, on the training set. ConvE is an efficient multi-layer convolutional network model. Unlike most other link prediction models that take as input an entity pair and a relation as a triple (p, e_1, e_2) and score it (1-1 scoring), ConvE takes one (p, e_1) pair and scores it against all entities e_2 (1-N scoring). This improves the training time of ConvE, however more importantly, it is very fast at inference time as well. This is particularly important for our method as we apply the link prediction model exhaustively to predict new high-quality facts (section 4.4.4).

We learn 200-dimensional vectors for each entity and relation. We use the default parameter settings of the ConvE model as those parameters yielded good results on the validation set.[7] We run the model for 80 epochs where the model has converged (less than 10^{-5} change in training loss). We learn embeddings for each relation and its reverse to handle examples where the entity order of the two relations are different.

For evaluating on the entailment task, we calculate entailment scores by using the predictions of the link prediction model on the triples in train, development and test sets. This is because the other baselines have also access to the whole set of triples (section 4.4.4). However, for evaluating the link prediction model, we compute entailment scores by only considering the predictions in the training set. This is essential as the entailment scores will be used to predict improved link prediction scores on the test set. Therefore, the comparison will not be valid if the method has access to the test triples while computing entailment scores.

[6] Accessed from https://github.com/TimDettmers/ConvE.

[7] We experimented with changing the learning and dropout rates, but the results did not improve on the validation set.

4.4.3 Evaluation Datasets

We discuss the datasets that we use to test the proposed methods for the entailment detection and the link prediction tasks.

Entailment Detection Evaluation. For the entailment detection task, we evaluate on Levy/Holt's dataset (Levy and Dagan, 2016; Holt, 2018). As discussed in chapter 3. each example in the dataset contains a pair of triples where the entities are the same (possibly in the reverse order), but the relations are different. The label of the examples are either positive or negative, meaning that the first triple entails or does not entail the second triple. For example *Bartlett was interviewed on television*, entails *Bartlett appeared on television*, but the latter does not entail the former. The dataset contains 18,407 examples (3,916 positive and 14,491 negative). We use the same development (30%) and test sets (70%) split as in section 3.5.

Link Prediction Evaluation. For the link prediction task, we evaluate the models on the test set of the NewsSpike corpus (section 4.4.2) that has 40K triples. For each triple, we compare the link prediction score with the score of a corrupted triple by changing one of the entities in the triple.

4.4.4 Comparison

We compare the following entailment scores for evaluating on the entailment detection dataset.

MC is the entailment score based on the Markov chain (4.3.1), when the link prediction scores are computed only for the predicates we have seen in the corpus. While the link prediction method can assign scores to any possible triple, we report this results to check how the Markov chain model performs compared to the other scores that are directly computed for the triples in the corpus.

Aug MC is our novel entailment score that is based on the Markov chain, but augments the matrix S of the MC model with new entries. We use the link prediction method to compute scores on the original set of triples as well as new predicted triples. For each triple (p, e_1, e_2), we compute the score $S_{p,e_1,e_2'}$ for all candidate entities e_2' that have been seen with e_1 for any other relation p'. We augment the matrix S with the K highest scores. We similarly score S_{r,e_1',e_2} for all candidate entities e_1' and augment the matrix S with the K highest scores, accordingly. In our experiments, we used $K = 50$.[8]

[8]Higher values of K were not feasible on our machines. We performed our experiments on a 32-core 2.3 GHz machine with 256GB of RAM.

Cos is the cosine similarity of the embeddings of the relations if the cosine is positive, and 0 otherwise. We also compare to three Sparse Bag-of-Word (SBOW) methods: **Weeds** (Weeds and Weir, 2003), **Lin** (Lin, 1998), and Balanced Inclusion (**BInc**; Szpektor and Dagan, 2008). These similarities check the set of entity-pairs for each relation pair and compute how much one set is included in the other, and/or how much they overlap. Similar to chapter 3, we have computed these scores based on the Pointwise Mutual Information (PMI) between the relations and the entity pairs.

Berant's ILP is the method of Berant et al. (2011). It computes local similarities and then learns global entailment graphs satisfying transitivity constraints by solving an Integer Linear Programming. We downloaded Berant et al. (2011)'s entailment graphs and tested it on the Levy/Holt's dataset.[9]

For all the above similarities, we report results both in the **local** setting, where the similarities are computed for each relation pair independent of the others and the **global** setting, where we apply the global soft constraints introduced in chapter 3. We apply two sets of global soft constraints: a) Cross Graph which transfers similarities between relations in different, but related typed graphs; and b) Paraphrase Resolution which encourages paraphrase relations to have the same patterns of entailment. We tune the parameters of the global soft constraints on the development set of the Levy/Holt's dataset.

For the link prediction task, we compare the ConvE model with our proposed link prediction score. We test how MC and Aug MC entailment scores can improve the link prediction scores in both local and global settings.

4.5 Results and Discussion

We first compare our proposed entailment score with the previous state-of-the-art results (section 4.5.1) and then show that we can use entailment decisions to improve the link prediction task (section 4.5.2).

4.5.1 Entailment Scores based on Link Prediction

In this section, we compare the variants of our method to the previous state-of-the-art results on the Levy/Holt's dataset. We compute similarity scores and report precision-recall curve by changing the threshold for entailment between 0 and 1. In order to have

[9]The entailment graphs of Berant et al. (2015) yield similar results.

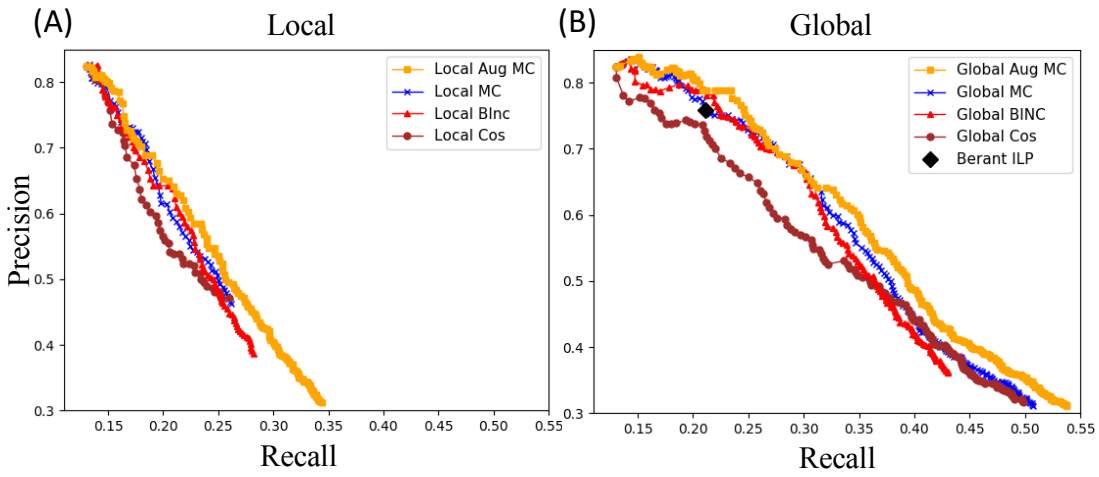


Figure 4.3: Comparison of the Markov chain (MC) and Augmented Markov chain (Aug MC) models to the BInc score (SBOW model) on Levy/Holt's dataset in local (A) and global (B) settings.

a fair comparison with Berant's ILP method, we first test a set of rule-based constraints proposed by them (Berant et al., 2011). We also apply the lemma baseline heuristic process of Levy and Dagan (2016) before testing the methods.

Figure 4.3 shows the precision-recall curve of all the methods in both local (A) and global (B) settings. From the SBOW methods, we only show the BInc score in the graphs as it got the best results on the development set. For Berant's ILP method, we only have one point of precision and recall, as we had access to their entailment graphs for only one sparsity level. In both settings, Aug MC works better than all the other methods. This confirms that the link prediction method is indeed useful for finding entailment rules. Aug MC consistently outperforms MC suggesting that adding the missing entries before forming the Markov chain alleviates the sparsity problem inherent to the entailment task.

Interestingly, while the MC model has access to the same set of entity-pairs as the BInc score, it outperforms it in most of the recall range (especially in the high recall range). Note that the link prediction method might still assign a low score to a triple (p, e_1, e_2) in the corpus if it is not consistent with the other facts. This is especially important when the input triples are noisy. Therefore, an accurate link prediction model can successfully remove the noise from the input triples. For triples extracted directly from text, the noise might come from various sources such as the relation extraction components (e.g, parsing and named entity linking) or fake or inconsistent news. The cos similarity is worse than the other methods. This is mainly attributed to the fact that

cos is symmetric, while the entailment relation is directional.

We also report area under the precision-recall curve. Because the different methods cover different ranges of precision and recall values, we compute area under the precision-recall curve for the precision range $[0.5, 1]$, as it is covered by all the baselines and the precision values higher than random are more important for end applications such as semantic parsing or summarization. Table 4.1 shows the area under precision-recall curves for all the methods. In the global setting, Aug MC shows about 13% improvement relative to the best result of the methods based on SBOW vectors (.187 vs .165). In addition, it is 25% higher relative to the cos score (15%). Similar patterns can be seen in the local setting.

4.5.2 Effect of Entailment Scores for Improving Link Prediction

We now test the proposed method for improving the link prediction score. Each triple (p, e_1, e_2) in the test set is corrupted by either replacing its first or second entity by any possible entities. The candidate entities are then ranked in descending order based on their plausibility score. The original entity is then ranked among all the other entities. We report results using a filtered setting, i.e., we rank test triples against all other triples not appearing in the training, validation or test sets (Bordes et al., 2013). We report Hits@1 (the proportion of the test triples for which the correct entity was ranked as the first prediction), Hits@10, Mean Rank (MR) and Mean Reciprocal Rank (MRR).

Table 4.2 shows the results of link prediction. We report the results for all entities as well as infrequent entities, where in the latter case we have removed any triple with an entity in the top 20 most frequent entities. In each setting, the first row is the plain ConvE model. We then test how the different variants of our entailment scores change the results. We observe that adding the entailment scores improve the rankings of the correct triples. The value of MRR, Hits@1 and Hits@10 have increased after applying any of the methods for learning entailment scores.

It is interesting to see that the improvements obtained by the different entailment scores are generally consistent with the results on the entailment detection task, i.e., the scores with better results on the Levy/Holt's dataset, show more improvements on this task as well. The change of the mean rank (MR) is more apparent. For example, MR has decreased about 50% when we apply our best method (Global Aug MC) to re-rank the link prediction scores. This means using entailment rules is more useful to improve the link prediction for harder examples. The results of all methods for

	SBOW			Link Prediction		
	Weeds	Lin	BInc	Cos	MC	Aug MC
Local	.073	.074	.076	.067	.079	**.085**
Global	.147	.149	.165	.150	.174	**.187**

Table 4.1: Area under precision-recall curve (for precision > 0.5) on Levy/Holt's dataset.

	Hits@1	Hits@10	MR	MRR
	ALL entities			
ConvE	20.36	47.93	1999.29	29.58
+ Local MC	20.66	48.64	1157.33	30.03
+ Local Aug MC	**20.68**	48.90	1018.37	30.12
+ Global MC	**20.68**	49.13	1012.54	**30.19**
+ Global Aug MC	20.64	**49.16**	**987.13**	**30.19**
	INFREQUENT entities			
ConvE	19.05	45.59	2124.71	27.94
+ Local MC	19.26	46.10	1303.56	28.25
+ Local Aug MC	**19.30**	46.36	1154.06	28.33
+ Global MC	19.29	46.60	1154.28	28.41
+ Global Aug MC	19.28	**46.66**	**1118.09**	**28.43**

Table 4.2: Link prediction results on the test set of NewsSpike for all entities (top) and infrequent entities (below). We test the effect of refining ConvE scores with entailment rules.

infrequent entities are worse than the results on all entities; however, we observe the same trends among the different methods.

Note that the amount of the data that is used for all the methods is the same. In particular, we have only used the triples from the NewsSpike corpus for both link prediction and entailment detection tasks and the gain in performance of the both tasks is merely because the two tasks learn complementary information.

Our model can provide explanations for getting the improvements in link prediction when we use the entailment graphs. Table 4.3 shows a few examples where entailment rules improve the link prediction scores. The target triples are extractions from the development set of NewsSpike (section 4.4.2), but have low link prediction scores ($<$0.05). Their scores are increased because alternative triples that entail them or are paraphrase of them have high link prediction scores ($>$0.95).

Target Triple	Alternative Triple
John Kerry **nominee for secretary of** state	John Kerry **confirmed as secretary of** state
Lady Gaga **canceled performance** in Hamilton	Lady Gaga **canceled show** in Hamilton
Dave Toub **considers anyone from** Jon Gruden	Dave Toub **considers everyone from** Jon Gruden
Zeke Spruill **traded in exchange for** Justin Upton	Justin Upton **sent in return for** Zeke Spruill

Table 4.3: Examples where entailment relations improve the scores of correct triples. The relations are **boldfaced**. In each row, the target triple has a low link prediction score (<0.05), but its score is increased because an alternative triple with high score (>0.95) entails the first triple or is a paraphrase of it. For each target triple, only one alternative triple is shown.

4.6 Related Work

Link Prediction. In recent years, many link prediction models have been proposed that learn vector or matrix representations for relations and entities (Socher et al., 2013; Bordes et al., 2013; Riedel et al., 2013; Wang et al., 2014; Lin et al., 2015b; Toutanova et al., 2016; Nguyen et al., 2016; Trouillon et al., 2016; Dettmers et al., 2018; Schlichtkrull et al., 2018; Nguyen et al., 2019; Balazevic et al., 2019). In our experiments we have used ConvE (Dettmers et al., 2018), however, our proposed score can be computed based on any link prediction model and the discovered entailment rules might be useful for improving any link prediction model. See section 2.2 for a detailed discussion on the link prediction literature.

Entailment Graph Induction. Entailment graphs are learned by imposing global constraints on local entailment decisions. Berant et al. (2011, 2012, 2015) have used transitivity constraints and applied Integer Linear Programming (ILP) or approximation methods to learn entailment graphs. In chapter 3, we have used two sets of global soft constraints to: (a) transfer similarities between different but related typed entailment graphs; and (b) encourage paraphrase relations to have the same patterns of entailments. Our method in this chapter, in contrast, learns a new entailment score to improve local decisions, which in turn improves the global entailment graphs.

Entailment Rule Injection for link prediction. There are some attempts in re-

cent years to improve link prediction by injecting entailment rules. Wang et al. (2015a) incorporate various sets of heuristic rules, including entailment rules, into embedding models for knowledge base completion. They formulate inference as an ILP problem, with the objective function generated from embedding models and the constraints translated from the rules. Guo et al. (2016) extend the TransE model by defining plausibility scores for grounded logical rules as well as triples and learning entity and relation embeddings that score positive examples higher than negative ones. Guo et al. (2018) take an iterative approach where in each iteration a set of unseen triples are scored according to the current link prediction model and a small set of precomputed logical rules. The new triples and their scores are then used to update the current link prediction model.

The above models need grounding of logical rules. A few recent works do not need grounding and are more space and time efficient (Demeester et al., 2016; Ding et al., 2018). They incorporate logical rules into distributed representations of relations. These models constrain entity or entity-pair vector representations to be non-negative. They encourage partial ordering over relation embeddings based on implication rules; however, their methods can be only applied to (multi-)linear link prediction models such as ComplEx (Trouillon et al., 2016). In contrast, our method can be applied to any type of link prediction model.

All these methods require entailment rules as their input. In most cases (Wang et al., 2015a; Demeester et al., 2016; Guo et al., 2016), the entailment rules are constructed manually, or selected from lexical resources such as WordNet (Miller, 1995). Therefore, the improvement of such methods come from out-of-domain knowledge (manually built lexical resources or expert knowledge), while our entailment rules come from in-domain knowledge, i.e., the same data which is used for link prediction. The number of entailment rules in all the previous models is very small because of scalability issues (at most a few hundred rules in Ding et al. (2018)). In contrast, our method can incorporate millions of automatically discovered entailment rules.

Modeling Reasoning Chains for Link Prediction. As discussed in section 2.2.5, the methods that explicitly model reasoning chains for link prediction are more transparent than the triple-based link prediction methods. For example, they can explain that (*starred in, Leonardo DiCaprio, The Revenant*) gets a high score based on the observations that (*played, Leonardo DiCaprio, Hugh Glass*) and (*character in , Hugh Glass, The Revenant*) have high scores or are simply observed in the original knowledge graph. Our method can similarly provide explanation for high scoring triples. For

example, it can be used to inform the user that (*played in, Leonardo DiCaprio, The Revenant*) gets a high score because (*starred in, Leonardo DiCaprio, The Revenant*) has a high score. While the two approach are similar, they have two main differences.

First, in our method, the supporting triples are between exactly the same entity pairs, e.g., (*Leonardo DiCaprio, The Revenant)*. However, in the previous reasoning chains approaches, the supporting triples form a path that could potentially pass through other related entities (e.g., *Hugh Glass*). Therefore, those methods cover a wider range of entailments that include first-order Horn clauses (Schoenmackers et al., 2010) such as (*starred in, A, B*) and (*character in, B, C*) entails (*played in, A, C*). It is therefore reasonable to extend our method to handle these types of entailments. In addition, the entailments hold between more than two entity types (e.g., *actor*, *movie* and *role*). Therefore, they can potentially share information better between multiple entity types, e.g., transferring entailments from (t_1, t_2, t_3) to (t'_1, t'_2, t'_3). This covers a broader set of entailment transfers than our method in chapter 3 that only handles two entity types in each graph , i.e., transferring entailments from (t_1, t_2) to (t'_1, t'_2).

Second, our method learns the general entailment rules explicitly. For example, the current approaches that model reasoning chains do not learn an explicit rule such as (*starred in, A, B*) entails (*played in, A, B*), but rather model these rules with specific grounded entities[10]. The advantage of general rules is that they apply to many entities. In addition, modeling the general rules explicitly makes it possible to apply learning algorithms such as the global method described in chapter 3 to improve the set of rules.

Future work might consider a single model that learns entailment graphs and link prediction jointly instead of having a separate model for the two tasks. The model could capture a more diverse set of entailments such as the ones in the reasoning chains methods, but it can also benefit from capturing general entailment rules explicitly.

4.7 Conclusions

We have shown that link prediction and entailment graph induction are complementary tasks. We have introduced a new score for entailment detection by performing link prediction on triples extracted from text. We reform the normal knowledge graph representation into a Markov chain with relations and entity-pairs as its states. The score is computed by estimating transition probabilities between the relation states. Our experiments show that the entailment graphs built by our proposed score outper-

[10]Our approach uses types such as *actor* and *movie* to disambiguate the context

form previous state-of-the-art results because link prediction is effective in filtering noise and adding new facts. We have additionally considered the reverse problem, i.e., using the learned entailment graphs to improve link prediction. Our results show that the two tasks can benefit from each other.

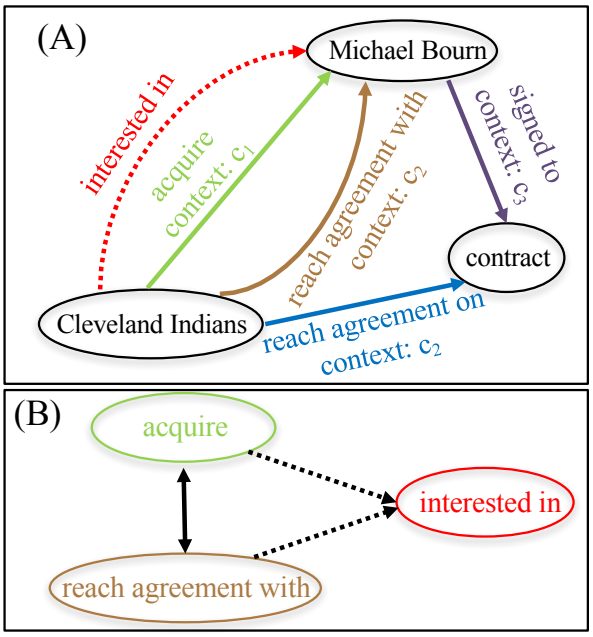
Chapter 5

Contextual Link Prediction to Learn Relational Entailment Graphs

Relational entailment graphs capture entailment rules between natural language predicates. The methods first process large text corpora to extract entity-entity relationships in the form of triples, and then compute entailment scores by measuring the inclusion of the entity pairs of one relation in another. Existing approaches suffer from the sparsity of the data, since many correct triples are not stated in the text. In chapter 4, we performed link prediction to predict new relations between the entities; however, the link prediction only uses the knowledge graph of the extracted triples and ignores the actual contexts of the extractions. In this chapter, we instead propose to perform contextual link prediction, i.e., predicting new relations given triples in textual context. We fine-tune existing pre-trained language models with an unsupervised contextual link prediction objective. We define a new entailment score that leverages the predicted triples. Our experiments show that the contextual link prediction extracts high-quality novel triples that yields new state-of-the-art entailment graphs. In addition, we show that the learned entailment graphs can improve the contextual link prediction task when tested on text with unseen entity pairs.

5.1 Introduction

Most existing approaches to learn relational entailments (Lin and Pantel, 2001; Szpektor et al., 2004; Berant et al., 2011, 2012, 2015) and the method proposed in chapter 3 use triples such as (*acquire, Cleveland Indians, Michael Bourn*) extracted from text to compute local entailment scores between the relations. The entailment scores are



c_1: Big-spending Cleveland Indians continued their bold off-season by **acquiring** free agent outfielder Michael Bourn.

c_2: The Indians **reached agreement with** Bourn on a four-year, $48 million contract on Monday night.

c_3: The Indians **signed** free agent center fielder Michael Bourn **to** a four-year, $48 million contract.

Figure 5.1: A) Contextual link prediction on a knowledge graph of triples extracted from text, where each triple is accompanied with its context(s). The relation tokens are boldfaced. B) An entailment graph between relations with entity types *sports_team* and *athlete*. The contextual link prediction model predicts the relation *interested in* given *acquire* in the context c_1 which is used in turn to add the link *acquire* $\rightarrow$ *interested in* to the entailment graph.

computed based on the Distributional Inclusion Hypothesis (Dagan et al., 1999; Geffet and Dagan, 2005). Learning global relational entailment graphs (Figure 5.1B) with relations as nodes and entailment rules as edges improves the local entailments by taking the dependencies between the entailment rules into account. This includes hard transitivity constraints (Berant et al., 2011, 2015) as well as the global soft constraints proposed in chapter 3. However, these approaches suffer from the sparsity inherent to the extractions since many correct triples are not mentioned in the text (Levy and Dagan, 2016).

In chapter 4, we performed link prediction on the assertions extracted from text to discover new relations between the entities. While using the new relations alleviates

the sparsity issue, the link prediction only uses the knowledge graph of the extracted triples and ignores the actual context of the extractions. In this chapter, we instead propose to perform contextual link prediction: Given a triple (p, e_1, e_2) in context between the relation p and entities e_1 and e_2, we predict all the other relations that hold between the two entities. In contextual link prediction, in addition to the knowledge graph structure, we have access to the context(s) that each triple is extracted from. After learning a contextual link prediction model, we can add missing links to the knowledge graph by considering each triple in context and adding predicted links between its two entities.

Figure 5.1A shows an example of contextual link prediction. Given the triple (*acquire, Cleveland Indians, Michael Bourn*) extracted from the sentence *Big-spending Cleveland Indians continued their bold off-season by acquiring free agent outfielder Michael Bourn.*[1], we predict all the relations that hold between the entity pair (*Cleveland Indians, Michael Bourn*). In addition to the graph structure, contextual link prediction has access to other clues such as *Big-spending*, *bold off-season*, and *free agent* in predicting the relations between (*Cleveland Indians, Michael Bourn*). We use the predicted relations (e.g., *interested in*) given each relation in context (e.g., *acquire*) to build relational entailment graph. Figure 5.1B shows a fragment of an entailment graph between relations with entity types *sports_team* and *athlete*. The contextual link prediction model assigns a high score to the relation *interested in* which will be used in turn to predict the link *acquire* $\rightarrow$ *interested in* in the entailment graph.

In this chapter, we define the contextual link prediction problem and present a model to use the contextualized embeddings of the relations to predict new relations that hold between the entities (section 5.4). The embeddings are initialized with BERT pre-trained model (Devlin et al., 2019), and are fine-tuned according to a contextual link prediction objective (section 5.5). We then propose a new entailment score based on the predicted relations in context (section 5.6). Our experiments show that the contextual link prediction model effectively predicts new relations that can be used to reduce the sparsity issues of the entailment graphs. Our model gets the state-of-the-art results on a challenging relational entailment dataset (section 5.8).

The structure of this chapter is as follows. We first introduce the notation in section 5.2 and review the related work in section 5.3. We define the contextual link prediction problem and propose our model in section 5.4. We discuss the training procedure in section 5.5 and propose our new entailment score in section 5.6. We discuss the details of the experiments in section 5.7 and analyze the results in section 5.8. We conclude

[1]We assume having access to a named entity linked corpus.

the chapter in section 5.9.

5.2 Notation

We define entities, types, and relations similar to chapter 4. In particular, let $\mathcal{E}$ denote the set of all entities (e.g., *Michael Bourn*) and T denote the set of all types (e.g., *athlete*). We consider binary relations and use the types of their two entities to disambiguate their meanings. Let $R(t_1,t_2)$ denote the set of all relations with types t_1,t_2, or t_2,t_1 (e.g., *acquire(sports_team,athlete)*). We define R as the set of all relations and $R(e_1,e_2)$ as the set of observed relations between the entity pair (e_1,e_2).

The relations can entail each other with the same entity order or the reverse order, e.g., *acquire(sports_team,athlete)* entails *reach agreement with(sports_team,athlete)* as well as *be player of(athlete,sports_team)*. When the two types are the same, we keep two copies of the typed relations and their triples so that we can model entailments with reverse entity orders, e.g., *buy player from(sports_team$_1$,sports_team$_2$)* entails *sell player to(sports_team$_2$,sports_team$_1$)*. In the second triple, we consider the reverse entity orders as the context, e.g., for the original triple *(buy player from(sports_team$_1$, sports_team$_2$),Cleveland Indians,San Diego Padres)*, we keep its reverse version as well, i.e., *(buy player from(sports_team$_2$, sports_team$_1$), San Diego Padres, Cleveland Indians)*. We specify the entity order by $o(p) \in \{0,1\}$. For all relations with unequal types, we set $o(p) = 0$. For relations with identical types, we set $o(p) = 0$ if the entities are in the original order and $o(p) = 1$, otherwise.

A triple mention is a predicate-argument structure extracted from text in addition to its textual context. We define a triple mention as a tuple $m = (p,e_1,e_2,c,s)$, where $p \in R$ is a relation and $e_1,e_2 \in \mathcal{E}$ are entities. The token sequence $c = [c_0,\ldots,c_n]$ is the textual context of the triple[2], and $s = (s_0,\ldots,s_k)$, are indices of the relation tokens. Figure 5.1A shows multiple triple mentions such as *(reach agreement with, Cleveland Indians, Michael Bourn, c_1,[3,4,5])*.[3] We denote by $\mathcal{D} = \left[(r^i,e_1^i,e_2^i,c^i,s^i)\right]_{i\in\{1,\ldots,N\}}$, where N is the total number of triples in context, the contextual knowledge graph consisting of all triple mentions. We define the (decontextual) knowledge graph consisting of all triples as $\mathcal{K} = \{(p,e_1,e_2) | \exists c,s : (p,e_1,e_2,c,s) \in \mathcal{D}\}$. We define $\mathcal{D}(p) = \{(p,e_1,e_2,c,s) | (p,e_1,e_2,c,s) \in \mathcal{D}\}$ as the set of all triple mentions of the relation p.

[2] c_0=[CLS] and c_n=[SEP] are special start and end tokens.

[3] we have omitted the types from the relation name for brevity. The indices of the relations could be shifted to the right if a word is divided into subwords during tokenization.

5.3 Related Work

Relational Entailment Graphs. The *local* learning approach to learn relational entailments predict entailment or paraphrase rules independently from each other. The work by Berant et al. (2010, 2011, 2012, 2015) and our method in chapter 3 take a *global* learning approach where the dependencies between the entailment rules are taken into account. These methods first compute local entailment scores and then build global entailment graphs that satisfy hard transitivity constraints or global soft constraints that consider both the structures across typed entailment graphs and inside each graph. In this chapter, we improve the local entailment scores that in turn improve the global entailment graphs.

Link Prediction. The existing link prediction models operate on a set of triples in a knowledge graph $\mathcal{K}$. These models are trained to assign high scores to plausible triples $(p, e_1, e_2) \in \mathcal{K}$ and low scores to implausible triples $(p', e'_1, e'_2) \notin \mathcal{K}$. While many models in recent years have been proposed for the link prediction problem (Nickel et al., 2011; Bordes et al., 2013; Trouillon et al., 2016; Dettmers et al., 2018; Kazemi and Poole, 2018; Balazevic et al., 2019), they have been mainly applied to existing knowledge graphs such as Freebase (Bollacker et al., 2008) or DBPedia (Lehmann et al., 2015). Our method in chapter 4 and the recent work by Broscheit et al. (2020) have performed link prediction on triples extracted from raw text, but the existing models only use the knowledge graph structure, and are not able to take advantage of the triple contexts. In this chapter, we introduce the contextual link prediction problem and propose a model to consider the textual contexts around the extracted triples as well as the knowledge graph structure. KG-BERT (Yao et al., 2019) uses contextual representation for knowledge base completion, however, they form synthetic token sequences by concatenating entity descriptions and relation tokens. In our model, we consider the natural text associated with the triples.

Fill in the blanks with Contextual Language Models. Recently, there has been an increasing interest in extracting factual knowledge, i.e., relations between entities, from pre-trained Language Models (LMs). As we discussed in section 2.3.2, these works form a prompt where an entity is missing (e.g., *Cleveland Indians acquire* __), and ask the language models to predict the missing entity (Petroni et al., 2019; Jiang et al., 2020). In contrast, our contextual link prediction model predicts the relations that hold between the two entities. The predicted relations are then used to compute improved entailment scores.

As described in section 2.3.1, Baldini Soares et al. (2019) propose the matching-the-blank (MTB) model to learn relation embeddings based on Lin and Pantel (2001)'s extension of Harris (1954)'s distributional hypothesis for relations. Their model encourages relations that share the same entity-pair to have similar embeddings. This is similar to our contextual link prediction training (section 5.5), but has three main differences: a) our model works on parser-based extracted triples (section 5.7.1), while MTB considers the text between two entities as a relation; b) we learn a directional score between relations in context and query relations, while MTB learns a symmetric score; and c) we can predict a score for any query relation while MTB needs the query relation with the same entity pair in another sentence in the corpus. Hence, our model can generalize to unseen relations which proves to be useful for detecting relational entailments (section 5.8.1). In addition, all the above models are tested on manually defined knowledge bases, but we test our contextual link prediction model on extractions from text (section 5.8.3).

5.4 Contextual Link Prediction Model

Given a triple mention $m = (p, e_1, e_2, c, s)$, we compute the probability that a query relation $q \in R$ holds between the entity pair (e_1, e_2), or in other words the probability that $(q, e_1, e_2) \in \mathcal{K}$. We consider two different types of embeddings for each relation.

First, the relation p has a contextualized embedding when it is observed in a triple mention $m = (p, e_1, e_2, c, s)$. This type of embedding is encoded by the contextualized embedding vector $\vec{m} \in \mathbb{R}^d$ of the triple mention, where d is the number of dimensions. Let $[\vec{h}_0, \ldots, \vec{h}_n]$ be the contextualized embeddings of c, where n is the number of tokens and $\vec{h}_i \in \mathbb{R}^d$. In our experiments, we use the contextualized embeddings of the relation's token(s) as the embedding vector of the triple mention. For multi-token relations, we use the average embedding vectors of the start and end tokens, i.e., $\vec{m} = (\vec{h}_{s_0} + \vec{h}_{s_k})/2$. We multiply the contextualized embedding with a matrix $A_0 \in \mathbb{R}^{d \times d}$, if the entities are in the original order ($o(p) = 0$), and $A_1 \in \mathbb{R}^{d \times d}$, if they are in the reverse order ($o(p) = 1$). This allows us to disentangle the embeddings of relations with original and reverse entity orders which in turn enables our model to capture entailments between relations that have reverse entity order (section 5.6).

Second, each relation has an out-of-context embedding retrieved from an embedding weight matrix. We use the out-of-context embedding $\vec{q} \in \mathbb{R}^d$ to encode the query relation. We define the contextual link prediction score as:

$$\mathrm{P}\big(q|m=(p,e_1,e_2,c,s)\big) = \sigma(\vec{m}A_{o(p)} \cdot \vec{q})$$
$$= \frac{1}{1+\exp(-\vec{m}A_{o(p)} \cdot \vec{q})}.$$

We encode the query relations with out-of-context embeddings rather than contextualized embeddings for two reasons: a) The model can be applied to relations $q \notin R(e_1,e_2)$, i.e., relations that are not observed with the same entity pair in any context. This is useful since we can find a set of novel relations that are likely to hold between the entity pair, but are not observed because of data sparsity. We show that augmenting the data with these relations improves the entailment scores (section 5.8.1). b) While the above score uses the inner product between embedding vectors, it is still asymmetric as it uses embeddings from different spaces for the relations p and q. In particular, if our data also contains a triple mention $m' = (q,e_1,e_2,c',s')$, the probability $\mathrm{P}(p|m')$ could be different from $\mathrm{P}(q|m)$. This is a desired property while computing entailment scores between relations (section 5.6).

5.5 Training

Given observed triple mentions $m = (p,e_1,e_2,c,s) \in \mathcal{D}$, we train a model to assign high contextual link prediction scores to relations q that hold between the entity pairs ($q \in R(e_1,e_2)$), and low scores to relations q' that do not hold between the entity pairs ($q' \notin R(e_1,e_2)$). This can be seen as a multi-label classification task with $|R(e_1,e_2)|$ positive examples, and $|R(t_1,t_2)| - |R(e_1,e_2)|$ negative examples, where t_1 and t_2 are the types of the entities e_1 and e_2. We do not consider negative relations with types other than (t_1,t_2) since we only model entailments between relations of the same types. We fine-tune the contextualized embeddings and learn the out-of-context relation embeddings by minimizing the following loss:

$$\mathcal{L} = -\sum_{m=(p,e_1,e_2,c,s)\in\mathcal{D}} \Big[\sum_{q\in R(e_1,e_2)} \log \mathrm{P}\big(q|m\big) + \sum_{q'\in(R(t_1,t_2)\setminus R(e_1,e_2))} \log\big(1-\mathrm{P}\big(q'|m\big)\big)\Big]. \tag{5.1}$$

In our experiments, the contextualized embeddings are initialized with BERT pre-trained embeddings (Devlin et al., 2019) and the out-of-context embeddings and the matrices A_0 and A_1 are initialized randomly.

5.6 Entailment Score between Relations

In this section, we discuss how to use the contextual link prediction model to compute entailment scores. We compute entailment scores w_{pq} between relations p and q. For a threshold $\delta > 0$, we build entailment graphs where nodes are relations $p \in R$, and edges are $(p,q) \in R \times R$ when $w_{pq} \geq \delta$.[4]

We define an entailment score similar to the Markov chain (MC) model. In chapter 4, we formed a bipartite graph between relations and entity pairs. We defined an MC with relation states $\langle p \rangle$ and entity pair states $\langle e_1, e_2 \rangle$, and edges $(\langle p \rangle, \langle e_1, e_2 \rangle)$ and $(\langle e_1, e_2 \rangle, \langle p \rangle)$ that connect each relation to its corresponding entity-pairs, and vice versa. The transition probabilities of the chain are normalized probability scores of triples (p, e_1, e_2) so that the outgoing edge weights from each state sum to 1. We defined the MC entailment score as:

$$w_{pq} = \mathrm{P}(\langle q \rangle | \langle p \rangle) = \sum_{e_1, e_2 \in \mathcal{E} \times \mathcal{E}} \mathrm{P}(\langle e_1, e_2 \rangle | \langle p \rangle) \, \mathrm{P}(\langle q \rangle | \langle e_1, e_2 \rangle).$$

In this chapter, we similarly define a Markov chain with relation states $\langle p \rangle$ and triple mention states $\langle m \rangle$. Each relation state $\langle p \rangle$ has directed edges to the mention states $\langle m \rangle$ such that $m \in \mathcal{D}(p)$, where $\mathcal{D}(p)$ is the set of all triple mentions of p. On the other hand, each mention $m = (p, e_1, e_2, c, s)$ has directed edges to a set of relations $R(m) = R(e_1, e_2) \cup R'(e_1, e_2)$, where $R'(e_1, e_2) \subseteq R \setminus R(e_1, e_2)$ contains a (small) set of novel relations with high contextual link prediction scores (section 5.7.3). The relations in $R'(e_1, e_2)$ have not been observed with the entity pair (e_1, e_2), but are likely to be valid since their contextual link prediction scores are high. Figure 5.2 shows an example Markov chain. We assign the transition probabilities from relations to mentions uniformly, and define the transition probabilities from mentions to relations as normalized contextual link prediction scores:

$$\mathrm{P}(\langle m{=}(p, e_1, e_2, c, s) \rangle | \langle p \rangle) = \frac{1}{|\mathcal{D}(p)|}$$

$$\mathrm{P}(\langle q \rangle | \langle m \rangle) = \frac{\mathrm{P}(q|m)}{\sum_{p \in R(m)} \mathrm{P}(p|m)}$$

[4] We learn a separate typed entailment graph for each type pair, but we have dropped the types in this section for the simplicity of the notation.

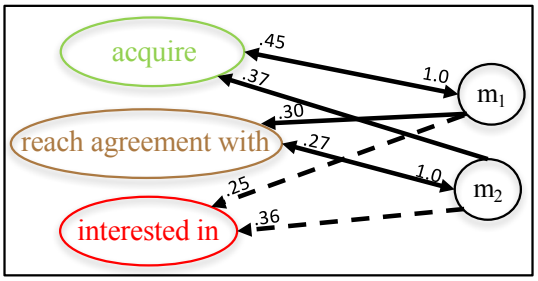


Figure 5.2: A small Markov chain between relations *acquire*, *reach agreement with*, and *interested in* (p_1, p_2 and p_3) on the left side, and triple mentions $m_1 = (p_1, e_1, e_2, c_1, s_1)$ and $m_2 = (p_2, e_1, e_2, c_2, s_2)$ on the other side, where e_1 and e_2 are *Cleveland Indians* and *Michael Bourn*, c_1 and c_2 are the same as the contexts in Figure 5.1, and s_1 and s_2 indicate the relation tokens. The edge weights show transition probabilities between the states. The dotted lines correspond to novel relations discovered by the contextual link prediction model.

We define the contextual Markov chain entailment score as:

$$\mathrm{P}(\langle q \rangle | \langle p \rangle) = \sum_{m \in \mathcal{D}(p)} \mathrm{P}(\langle q \rangle | \langle m \rangle) \, \mathrm{P}(\langle m \rangle | \langle p \rangle). \tag{5.2}$$

This score corresponds to the probability that a random walk with length 2 from the relation state $\langle p \rangle$ ends in the relation state $\langle q \rangle$.

5.7 Experimental Setup

In this section, we discuss the details of the corpus we have used (section 5.7.1), training (section 5.7.2) and entailment graph building (section 5.7.3)

5.7.1 Text Corpus with Triple Mentions

Similar to chapters 3 and 4, We perform our experiments on the NewsSpike corpus that contains 550K news articles from various news sources (Zhang and Weld, 2013). In chapter 3, we processed the corpus with a Combinatory Categorial Grammar (CCG; Steedman, 2000) semantic parser and extracted typed and linked triples, where each typed relation (entity-pair) has occurred with at least 3 unique entity-pairs (relations). The entities are linked to Freebase and assigned one of the 49 first-level FIGER types (Ling and Weld, 2012). We built 346 typed entailment graphs with a total of $|R| =$

304K typed relations. There are 7 graphs with more than 10K relations, where the largest graph has 53K relations.

In this chapter, we adapt the procedure in chapter 3 to couple each triple with its mentions in the text so that we can identify the tokens of the relations and compute embeddings in context (section 5.4). This yields $|\mathcal{D}| = 8.5$M triple mentions for $|\mathcal{K}| =$ 3.9M unique triples.

5.7.2 Training

We use the Hugging Face transformers library (Wolf et al., 2019) and train our model by minimizing the loss function introduced in Section 5.5. We randomly split the triple mentions into training (95%), development (2.5%) and test (2.5%) sets.We start with BERT-base pretrained embeddings for triple mentions and randomly initialize the out-of-context relation embeddings.[5] We process batches with the same entity-pair types to speed up the training by accessing the out-of-context embeddings of the same set of typed relations in each batch. We use batches of size 64 as this was the largest that we could process with our GPU machines.

In the news corpus, the number of positive examples (typically less than 100) for training is much less than the number of negative examples, since for each triple we only consider relations that occur with the same entity-pair as positive examples ($R(e_1, e_2)$), and all other relations with the same types are considered to be negative ($R(t_1, t_2) \setminus R(e_1, e_2)$). This causes a class imbalance problem, especially for type-pairs with many relations (up to 53K). To alleviate this problem, we consider a small subset of all relations as candidates for contextual link prediction: a) the union of all in-batch positive relations; b) a random subset of other relations up to the number of the in-batch positive relations. For each triple mention, the relations in the subset that are not positive are considered to be negative relations.[6] This leaves the first term in Equation 5.1 unchanged for each triple mention. But the second term is computed on a smaller subset of all negative relations. This still gives us a relatively high number of negative

[5] We also tried RoBERTa-base (Liu et al., 2019), but the results were similar. We could not use BERT-large or RoBERTa-large in our experiments because of memory limitations. We performed the neural network training and inference experiments on NVIDIA P102 GPU machines with 11GB of memory.

[6] We also tried a combination of random negative relations and hard negative relations, where the relations share exactly one entity in the mention. However, the results were similar and we report the results of the simpler method.

examples (around a few hundred negative examples in average). In our experiments, we also tried using up to total of 1000 negative relations per batch, but the results stayed similar. Using all possible negative relations yielded poor results.

We used Adam (Kingma and Ba, 2015) with linear decay of learning rates for optimization. We optimize a multi-label classification task. For each triple mention embedding ($\vec{m}$), we have many (in practice around 100) out-of-context embeddings. Therefore, each contextualized embedding will be updated around 100 times the number of updates for the out-of-context embeddings. In practice, we found that this will be problematic as the contextualized embeddings will be hugely affected by the out-of-context embeddings and the model does not converge to a good solution. Therefore, we used different initial learning rates for the two sets of embeddings.

We tuned hyperparameters by maximizing the mean average precision (MAP) of contextual link prediction in the development set. To compute the MAP, we consider each triple mention as a query, and rank the set of positive and negative relations according to their contextual link prediction score. Having positive relations at the top of the ranked list leads to higher MAP values. We tuned initial learning rates to 10^{-6} for contextualized embeddings and 10^{-4} for out-of-context embeddings. We found that 40 tokens (up to 20 tokens at each side of the relations) are sufficient for the context, where the context can cross multiple sentences. We optimized the model for 10 epochs over the whole corpus.

5.7.3 Building Entailment Graphs

We compute entailment scores and build local entailment graphs. We then apply the cross graph and paraphrase resolution soft constraints (chapter 3) to build globally consistent entailment graphs.

We compute the new entailment scores using all the NewsSpike triple mentions and based on the Markov chain defined in Section 5.6. In the Markov chain, the relations are connected to their mentions in the corpus. Each mention $m = (p, e_1, e_2, c, s)$ is connected to $R(m)$ that contains both the relations that appear with the same entity-pair (e_1, e_2) in the corpus as well as a small set of relations that have the same types. We first compute contextual entailment scores for a candidate set of relations. We then connect the mentions to the relations with the highest contextual link prediction scores. Adding more connections to the Markov chain is useful to alleviate the sparsity issues inherent to the data.

In our experiments, the mentions that have fewer than K=100 relations with the same entity pairs, are connected to 100 relations. If the total number of such relations were less than 100, the mention is connected to all relations with the same types as its entities. While the candidate set could be the whole set of relations, we considered negative relations in the batch at the inference time. We build batches of data based on a stream of news stories, it is therefore reasonable to assume that most of the relations that entail each other or are paraphrases, are more likely to appear around the same time in the news text (Zhang and Weld, 2013; Zhang et al., 2015). If we use larger batch sizes, we have more candidate relations; however, computing entailment scores for the candidate relations takes more time. We used the batch size of 512 that gives us a relatively high number of candidate relations, while it keeps the entailment score computation still tractable.[7]

Finally, we multiplied the contextual link prediction scores of the new connected relations by a factor $\alpha \in (0, 1]$ before computing the chain probabilities and the entailment scores. This guides the entailment score to rely more on the original connections and is useful to improve the precision of the graphs. We tuned $\alpha = 0.5$ based on the development set of Levy/Holt's entailment dataset (section 5.8.1).

5.8 Results and Discussion

In this section, we first compare our new entailment score with the previous state-of-the-art models (section 5.8.1). We then evaluate the directionality of the entailment scores (section 5.8.2). We finally evaluate the contextual link prediction task (section 5.8.3).

5.8.1 Evaluating Entailment Graphs

We evaluate the entailment graphs on the Levy/Holt's entailment dataset (Levy and Dagan, 2016; Holt, 2018) which is discussed in section 3.5.2.

We chose to perform experiments on the Levy/Holt's dataset for two reasons: 1) It has resolved a major issue in other datasets such as Zeichner's (Zeichner et al., 2012) and SherLIiC (Schmitt and Schütze, 2019), where candidate positive and negative en-

[7] It takes around 10 days to compute the entailment scores. We also experimented with lower values of K (40) and smaller batch sizes (256 and 128), and the results were slightly worse, with the batch size being more important than K.

tailments have been collected from the top-scoring entailments according to an existing entailment score. Therefore, those datasets do not contain long-tail challenging examples that cannot be directly extracted from text corpora.[8] 2) The Levy/Holt's dataset contains the reverse triple pair for any positive example. This makes the dataset more suitable for checking whether the models capture directionality of entailments compared to just predicting whether the two relations are similar.

We compare the following models: **Aug Contextual MC** is our novel model, where the entailment scores are computed after augmenting the Markov chain with new connections from triple mentions to relations with high contextual link prediction scores. **Contextual MC** is our model without the new connections. **Aug MC** is the proposed model in chapter 4, where we computed the entailment scores based on a Markov chain between relations and entity-pairs. We augmented the Markov chain based on scores of a link prediction model. **MC** is similar to Aug MC, but without any additional links between relations and entity-pairs. Balanced Inclusion (**BInc**) score is a Sparse Bag-of-Word (SBOW) model, where each relation is represented using a vector of entity-pairs (Szpektor and Dagan, 2008).

Similar to the previous chapters, we first apply a lemma baseline (Levy and Dagan, 2016) and a set of constraints (Berant et al., 2011) before applying any of the models. We then use the entailment score between the relations of each example, and predict positive (negative) entailment if the score is greater than or equal to (less than) a threshold. We report precision-recall curves by changing the threshold between $[0, 1]$. Figure 5.3 shows precision-recall curves in (A) local and (B) global settings.[9] We report the area under the precision-recall curves for precision > 0.5. Table 5.1 (left) shows the results for all models.

The MC model and the contextual MC model are defined in a similar way. They both only use the extractions from the text corpus. Their overall results are close. The Aug MC model, and our novel model, Aug Contextual MC, augment the data with additional links. The former uses a link prediction model to add new links, while the latter does that based on the contextual link prediction. They both improve the results compared to the models without augmentation. However, Aug Contextual MC outperforms Aug MC, i.e., the previous state of the art model, in both local and global

[8] Our preleminary experiments on the SherLIiC dataset showed that entailment graphs learned by simple count-based models such as Balanced Inclusion (Szpektor and Dagan, 2008) outperform the ones learned with embedding-based models.

[9] We have not shown the MC model in the plots for more clarity.

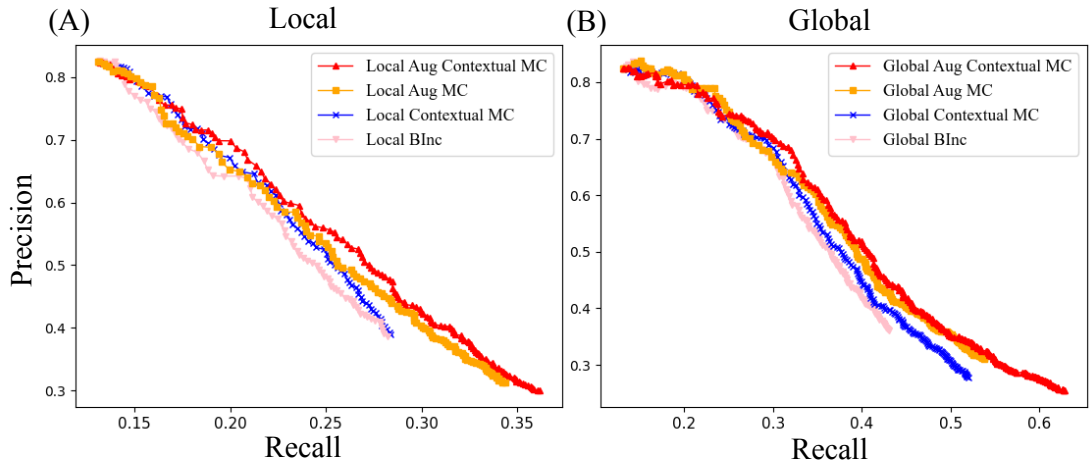


Figure 5.3: Comparison of the Aug Contextual MC and Contextual MC models with models based on only triples (BInc and Aug MC) on Levy/Holt's dataset in (A) local and (B) global settings.

	ALL		DIR
	Local	Global	Global
BInc	.076	.165	.155
MC	.079	.174	.159
Aug MC	.085	.187	.163
Contextual MC	.084	.176	.159
Aug Contextual MC	**.096**	**.195**	**.165**

Table 5.1: Area under the precision-recall curve on a) left: the Levy/Holt's dataset (precision > 0.5) and b) right: the directional subset of the Levy/Holt's dataset (recall ≤ 0.33).

settings.

The improvement in the entailment detection task is mainly because the contextual link prediction model discovers new high-quality relations beyond the original extractions. Each triple mention in the MC (Figure 5.2) gets linked to the predicted relations. Computing the entailment score on the augmented MC alleviates some of the sparsity issues of the relational entailment problem. This can be seen by comparing the Aug Contextual MC model with the Contextual MC model. For instance, in Levy/Holt's development set, the entailment score of *give rise to* $\rightarrow$ *cause* is 0 for all models except the Aug Contextual MC model. This is because while there are no shared entity pairs between the two relations in the text corpus, the contextual link prediction model predicts *cause* given the triple mentions with the relation *give rise to*.

5.8.2 Evaluating Directionality of Entailment Graphs

We evaluate all models on the directional portion of the Levy/Holt's dataset. This portion is a subset of the main dataset and contains 2414 examples (630 in dev and 1784 in test). For any triple pair in this portion, the reverse of the pair is also present. The entailment is correct in one direction and incorrect in the other. For example, *Printing press was invented by Gutenberg* entails *Gutenberg developed the printing press*, however, the entailment is not correct in the opposite direction.[10] This makes the task much harder than the one of the original dataset. Even a perfect paraphrasing model (two-way entailment) gets the precision of exactly 0.50. Therefore, the model needs to specifically score the entailment in one direction above the other direction. For the original dataset, a symmetric score such as Lin score (Lin, 1998), that is only aware of relatedness between relations but cannot distinguish the directions, can still solve many examples correctly and yield high precision values (chapter 3).

Figure 5.4 shows the precision-recall curves for global models. We report the area under the curves in Table 5.1 (right col) for recall ≤ 0.33 that are covered by all models. In order to have a fair comparison between the Aug Contextual MC and the Aug MC models, we also computed the area under the curve for recall ≤ 0.48 that is covered by both models: the area under the curves are 0.251 and 0.250, respectively. The results show that defining the entailment scores on a Markov chain as the probability that a path (of length 2) from one relation ends in another relation is an effective way to predict directional entailments. Augmenting the Markov chains with additional links further improves the results. Note that while the two models with augmentation get better overall results, the precisions for all models are still relatively low (≤ 0.60). In addition, the precision is not high even for low recalls meaning that the models cannot separate the directionality of the entailments well even if the entailment scores are very high. This calls for more research on finding the direction of the relational entailments.

5.8.3 Evaluating Contextual Link Prediction

In this section, we evaluate the contextual link prediction model itself. We compute the MAP of predicting relations q that hold between the entity pairs in a triple mention $m = (p, e_1, e_2, c, s)$ as discussed in section 5.7.2. We compare different approaches to this task. The first approach is to use the entailment graphs and score relations q

[10] During the data annotation, one of the arguments is masked with its type so that world knowledge does not bias the data (Levy and Dagan, 2016).

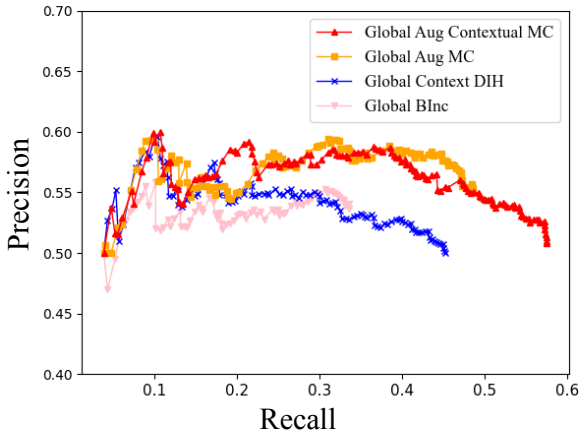


Figure 5.4: Comparison of models on the directional portion of the Levy/Holt's dataset.

	Split Type	
	Random	Entity Pair
Contextual Link Pred	.618	.205
Contextual MC	.423	.316
Aug Contextual MC	.421	.327
Contextual Link Pred + Contextual MC	**.708**	.354
Contextual Link Pred + Aug Contextual MC	.704	**.357**

Table 5.2: MAP of relation prediction on the NewsSpike test set for Contextual Link Prediction compared to Entailment Graphs as well as the combination of the two approaches.

based on the entailment score $p \rightarrow q$. This approach only uses the types of entities from the context, but does not take the whole context into account. It instead relies on the entailment score w_{pq}, where the entailment score is a weighted average of the contextual link prediction scores of q given mentions of p (Equation 5.2). The second approach is to use the contextual entailment score $\mathrm{P}(q|m)$. In addition, we combine the two approaches to test whether entailment graphs and contextual link prediction learn complementary information. We consider the linear summation $\lambda w_{pq} + (1-\lambda)\mathrm{P}(q|m)$, where $\lambda \in [0,1]$ is a hyper-parameter. We tuned λ using the triple mentions in the development set.[11]

We compute the MAP on the test portion of the NewsSpike corpus in two settings: a) the triple mentions in the corpus are randomly split into training, development, and test sets. This is the setting that we used to build the entailment graphs (section 5.7.2). b) the triple mentions are split in a way that each entity pair (e_1, e_2) and its reverse

[11] In all of our settings in this section, $\lambda = 0.95$ performed the best on the development set.

(e_2, e_1) are present in either training, development, or test sets. The second split is more suitable to test whether the model generalizes well to unseen text. This is because in the first setting, a model that perfectly memorizes $R(e_1, e_2)$, i.e., all the relations q where (q, e_1, e_2) is a valid triple in the corpus, can get MAP close to 1 (MAP becomes exactly 1 if at least one mention of each entity pair appears in the training set).

For testing on the random split, we use our trained model in section 5.7.2 and build entailment graphs using only the training portion of the triple mentions. For testing on the entity-pair split, we train a new model and build new entailment graphs on the training portion of the entity-pair split. Table 5.2 shows the results, where we compare the contextualized link prediction score against the entailment graphs with Contextual MC score and Aug Contextual MC score as well as the combination of the contextual link prediction score and the entailment graphs. With the random split, the contextualized link prediction gets better results than the entailment graphs. However, with entity-pair split, the MAP of the contextualized link prediction model drops significantly compared to the entailment graphs. This confirms that the entailment graphs are more robust while testing on unseen text. Moreover, higher MAP with Aug Contextual MC score compared to Contextual MC score while testing on the entity-pair split, means that the discovered relations improve the generalizability of the entailment graphs when tested on the contextual link prediction task. This is consistent with the results of the entailment dataset. Contextual MC gets better results than Aug Contextual MC in the random split setting. This could be because in that setting, many triple mentions are present in both training and test splits. Therefore, Contextual MC entailment graphs that are only based on the triples in the training set work better on the test set compared to Aug Contextual MC that predicts new potentially correct, but novel relations.

When we combine the embedding-based contextual link prediction model with the entailment graphs (last two rows of Table 5.2), we get improvements in both random split and entity-pair split settings. These results confirm that the two approaches in fact benefit from each other. Table 5.3 shows an example from the development set of the entity-pair split where the contextual link prediction model improves the results of the entailment graphs. The extracted triple from the context is *Microsoft is committed to success*. The entailment graphs predict high scores for some wrong relations, e.g., *Microsoft builds success*. The reason for this prediction is that the typing system we have used has assigned the general type *thing* to the entity *success*[12]. The type *thing* is

[12]The type *thing* is assigned to entities that are not linked to any entity in Freebase or their Freebase

Context	Microsoft **is committed to** the long term success of the entire PC ecosystem.	
Triple	Microsoft **is committed to** success	
Types	(Organization,thing)	
Predictions	Microsoft **builds** success	↓
	Microsoft **switches to** success	↓
	Microsoft **'s** success	↑
	Microsoft **achieves** success	↑
	Microsoft **hopes for** success	↑

Table 5.3: An example where contextual link prediction improves (indicated by downward or upward arrows) entailment graph predictions. The relation tokens are **boldfaced**.

Context	Apple **is working on** a high-tech watch	
Triple	Apple **is working on** watch	
Types	(Organization,thing)	
Predictions	Watch **falls on** Apple	↓
	Apple **'s** watch	↑
	Apple **has** watch	↑
	Apple **launches** watch	↑
	Apple **tests** watch	↑

Table 5.4: An example where entailment graphs improve (indicated by downward or upward arrows) the results of contextual link prediction. The relation tokens are **boldfaced**.

assigned to many other entities such as *system*. The entailment signal comes from extractions such as *Apple is committed to system* and *Apple builds system*. The extracted relations used in the entailment graphs conflate different senses of the relation *build* leading to wrong entailments. However, the contextual link prediction model which is based on BERT can disambiguate the context because it embodies a latent type structure that captures the difference between entities such as *success* and *system*. The table also shows some correct relations that their scores are increased by the contextual link prediction model. In addition, Table 5.4 shows an example where entailment graphs

types do not have a mapping to FIGER types.

perform better than the contextual link prediction model. For example, the embeddings of some infrequent relations such as *falls on* have not been learned well and they get a high contextual score, but the entailment graphs do not contain these wrong entailments.

5.9 Conclusions

We have introduced the contextual link prediction problem and have used it to improve the relational entailment detection task. We presented a model based on the in-context embeddings of relations and out-of-context embeddings of query relations. We trained our model on a corpus of triple mentions and used it to assign scores to both observed and unobserved triples. We defined a new entailment score based on the predicted triple scores. Our empirical evaluations show that we can use high-quality novel triples to reduce the sparsity issues of the entailment graphs. As in the previous chapter, we also showed that the learned entailment graphs can improve the contextual link prediction task.

Chapter 6

Conclusions

In this thesis, we tackled the problem of unsupervised learning of relational entailment graphs. We derived entailment graphs by processing large amounts of multiple-source news text and finding consistent entailment patterns between the events. We extracted predicate-argument structures in the form of binary relations by applying a fast Combinatory Categorial Grammar (CCG) parser to news text. We linked the entities to Freebase by running an entity linker on the text and assigned fine-grained types to them. We computed local entailment scores between typed relations based on the Distributional Inclusion Hypothesis for relations. The entailment graphs suffer from the noise and sparsity inherent to the data. We introduced a series of models for building large typed entailment graphs that improve the results by either imposing global constraints on the graphs, or improving the local scores by neural learning of relation and entity representations. We showed that the entailment graphs can be used to improve question answering and link prediction tasks and also explain the improvements.

In chapter 3, we proposed two sets of global soft constraints on the structure of entailment graphs. We introduced a method to learn globally consistent entailment scores that are used to build global entailment graphs. The idea of imposing global constraints have been studied in the literature before; however, the previous works have mainly focused on hard transitivity constraints. The hard constraints are intuitive and have been proven to be effective; however, their exact Integer Linear Programming (ILP) solution does not scale to more than a few hundred relations and their approximate solutions are based on assumptions concerning the graph structure that are not correct for many real work entailments. Our method based on new global soft constraints scale to more than 100K relations and shows large improvements over previous state-of-the-art constraints.

We defined the cross graph soft constraints on different typed entailment graphs and the paraphrase resolution constraints inside each graph. The cross graph constraints encourage transferring of entailments between typed entailment graphs. The transfer can be done only if the typed relations in two graphs should have similar entailments. We proposed a method to infer transfer coefficients based on existing similarities between the entailments of typed relations in different graphs. The paraphrase resolution constraints encourage paraphrase relations, i.e., the relations that entail each other in both directions, to have similar patterns of entailment. We modeled the problem as inference in a Markov network, with entailment rules as nodes, and the dependencies imposed by the constraints as edges. Our method based on the Block Coordinate Descent method iteratively learns the global entailment scores and the transfer coefficients.

In chapter 4, we showed the duality of the entailment graph construction task and the link prediction task. The main reason for the sparsity of entailment graphs is that many correct assertions are not stated directly in the text. Therefore, inferring the unknown relations between entity pairs given the known ones yields data with higher coverage that is useful for building the graphs. We trained a link prediction method on the knowledge graph of triples extracted from text. We then predicted missing links in the knowledge graph. In order to compute local entailment scores, we reformed the knowledge graph into a bipartite graph between the relations on one side and the entity pairs on the other side. We formed a Markov chain on the bipartite graph with transition probabilities based on the link prediction scores.

We defined a new local entailment score, i.e, the Markov chain (MC) score, between two relations as the probability that a random walk (with a length of exactly 2) from the first relation ends in the second one. We observed that augmenting the MC with predicted assertions from the same text that the entailment scores were computed on, improves the entailment scores in both local and global settings. We then defined a new link prediction score based on the learned entailment scores as well as the original link prediction score. We showed that the new link prediction score predicts the missing links more accurately than the original score, with the augmented MC score yielding more improvement than the MC score. Our empirical evaluation confirms that the two tasks are indeed complementary.

We introduced the contextual link prediction task in chapter 5, where given a triple mention in context, we predict other relations that hold between the entity pair of the triple. We proposed a model based on the embeddings of relations in context as well as out-of-context embeddings of query relations. We fine-tuned the embeddings based

on a multi-label classification task, where each triple mention is considered as an example, and each correct (incorrect) query relation is considered as a positive (negative) label. The contextual link prediction model has access to more context than the normal link prediction model and can predict new relations between entity pairs more reliably. We defined the contextual MC score similar to the MC score on a Markov chain on relations and triple mentions as nodes and transition probabilities proportional to the contextual link prediction scores. Our results showed that when we add novel predicted relations to the computation of the contextual MC score, we get improved results compared to the augmented MC score. Moreover, we showed that the entailment graphs perform well on the intrinsic task of contextual link prediction. Our results demonstrate that embeddings-based contextual link prediction and the entailment graphs learn complementary information.

6.1 Future Work

We briefly discuss possible future directions on learning entailment graphs and their applications.

6.1.1 Temporal Information

The articles in the NewsSpike corpus as well as most other news corpora such as News Crawl[1] have associated timestamps. Zhang and Weld (2013) have shown that using temporal information is very effective in learning paraphrase clusters. Future work can extend our learning algorithms to incorporate temporal information. For example, distributional methods often confuse synonyms with antonyms (e.g., *win* entails *lose*) because antonym events can occur with the same entity pairs at different times. However, different articles describing identical news stories that have the same timestamps, as well as the natural order of the news, are good indicators for resolving such issues. Our preliminary experiments show that article timestamps can be used effectively to both remove such noisy entailments and reduce the size of the entailment graphs.

In addition, labeling the edges of the entailment graphs with the temporal ordering between events is useful for making more fine-grained inferences such as finding *precondition* and *consequence* relationships (Lewis and Steedman, 2014b). The temporal labeling can be done after the structures of the entailment graphs are decided, or can

[1] https://www.statmt.org/wiki/?n=Corpora.NewsCrawl

be learned jointly with the entailment graph induction task.

6.1.2 Iterative Learning of Entailment Graph Building and Link Prediction

Our results in chapters 4 and 5 show that (contextual) link prediction and entailment graph induction are complementary. We have used the outcome of each task to improve the other one. Future work might look into joint learning of the two tasks to improve both of them. Building the entailment graphs take time and it is not straightforward to completely update the graphs as the embeddings of the relations and entities get updated; however, it is possible to iterate between the two tasks to get improved results. Future work can also look at having a single model to perform both tasks simultaneously instead of iterating between the tasks. One possible approach is to extend the methods that explicitly model inference chains (section 2.2.5). The consistent patterns in the inference chains could be used to form and (partially) update entailment graphs and the entailment graphs could be used in turn to guide the models to choose inference chains that end in the correct entity nodes.

6.1.3 Learning Entailment Graphs with Other Arities

In this thesis, we only considered entailment graphs between binary relations, i.e., each relation in our entailment graphs has exactly two entities. However, there are many useful entailments with arities different from 2. This includes entailments between unary relations, e.g., *person walks* entails *person moves*, or relations with arities greater than 2, e.g., *person buys thing from location* entails *location sells thing to person*. In addition, some entailments hold between relations with different arities, e.g., $person_1$ *kills* $person_2$ entails $person_2$ *dies*. In the future, the methods in this thesis can be extended to entailment graphs with arities other than 2.

6.1.4 Learning Structured Relation Embeddings

Our work has resolved some of the sparsity issues of the entailment graphs: a) The soft constraints are used to share information between different typed entailment graphs. They also add more edges by detecting paraphrase relations; b) The link prediction and its contextualized version are used to augment the context vector of relations with new entity pairs, which reduce sparsity. However, the graphs still suffer from low recall

since in many cases that an entailment relation exists, the context vectors of the two relations do not have any overlaps, or there is no path in the Markov Chains introduced in chapters 4 and 5 between the two relations. Our embeddings based models add more paths to the Markov Chains and cover more entailment rules, but they do not cover all possible entailment rules because for some of them there is no evident signal in the text even with the use of embeddings.

An alternative approach is to develop structured embedding methods that capture directional relations. In these approaches, the entailment rules can be read off the embeddings themselves. Since the embeddings are dense and in low dimensions, they can introduce more overlap and reduce the sparsity. We performed some preliminary experiments with the box embeddings model (Vilnis et al., 2018; Li et al., 2019), where each relation was assigned a box represented by two vectors and each entity-pair was assigned a vector. However, the results were not positive because of the noise that was added to the graphs due to the high overlap of the embeddings of unrelated relations. Nevertheless, previous works on hypernymy detection have shown relatively positive results (Vendrov et al., 2015; Chang et al., 2018; Vilnis et al., 2018; Li et al., 2019) and it is possible that these methods could be adapted for detecting relational entailments.

6.1.5 Building Form-Independent Semantics and Applications of Entailment Graphs

As discussed in chapters 2 and 3, the cliques of the entailment graphs can be collapsed into paraphrase clusters with a single relation identifier. Future work can replace the form-dependent lexical semantics of the CCG parser with the form-independent relations (Lewis and Steedman, 2013a, 2014b). The resulting semantics can be used in tasks such as question answering, relation extraction, or summarization. In particular, relational entailment graphs can be used to perform improved open-domain factual question answering by a) using the CCG parser to extract relations in the question; b) using the entailment graphs to generalize the surface form of both the question and the text containing the answer; and c) using the form-independent semantics to handle complex language phenomena such as negation and composition.

In addition, future work can learn entailment graphs for other languages and align them together to obtain a form and language-independent semantics. The resulting semantics can be used to parse textual data and build a knowledge graph with relations corresponding to the paraphrase clusters in the entailment graphs.

6.1.6 Entailment Aware Language Models

In chapter 5, we fine-tuned pre-trained LMs for the contextual link prediction task. Future work might consider adding relational entailment knowledge into the pre-trained LMs. One approach would be to extend our contextual link prediction model to predict masked entities and relations by performing masked language modeling jointly with the contextual link prediction. The resulting LMs can be used to query for relational knowledge similar to the work of Petroni et al. (2019) and Jiang et al. (2020). They can also be used for downstream tasks such as relation extraction.

Bibliography

Omri Abend, Shay B. Cohen, and Mark Steedman. 2014. Lexical Inference over Multi-Word Predicates: A Distributional Approach. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 644–654, Baltimore, Maryland, USA.

Gabor Angeli, Melvin Johnson Premkumar, and Christopher D. Manning. 2015. Leveraging Linguistic Structure for Open Domain Information Extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 344–354, Beijing, China.

Yoav Artzi and Luke Zettlemoyer. 2011. Bootstrapping Semantic Parsers from Conversations. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 421–432.

Yoav Artzi and Luke Zettlemoyer. 2013. Weakly Supervised Learning of Semantic Parsers for Mapping Instructions to Actions. *Transactions of the Association for Computational Linguistics*, 1:49–62.

Ivana Balazevic, Carl Allen, and Timothy Hospedales. 2019. TuckER: Tensor Factorization for Knowledge Graph Completion. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5188–5197, Hong Kong, China.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. 2019. Matching the Blanks: Distributional Similarity for Relation Learning. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2895–2905, Florence, Italy.

Jonathan Berant, Noga Alon, Ido Dagan, and Jacob Goldberger. 2015. Efficient Global Learning of Entailment Graphs. *Computational Linguistics*, 42:221–263.

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, Seattle, Washington, USA.

Jonathan Berant, Ido Dagan, Meni Adler, and Jacob Goldberger. 2012. Efficient Tree-Based Approximation for Entailment Graph Learning. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 117–125, Jeju, Korea.

Jonathan Berant, Ido Dagan, and Jacob Goldberger. 2010. Global Learning of Focused Entailment Graphs. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1220–1229, Uppsala, Sweden.

Jonathan Berant, Jacob Goldberger, and Ido Dagan. 2011. Global Learning of Typed Entailment Rules. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 610–619, Edinburgh, Scotland, UK.

Rahul Bhagat, Patrick Pantel, and Eduard Hovy. 2007. LEDIR: An Unsupervised Algorithm for Learning Directionality of Inference Rules. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 161–170, Prague, Czech Republic.

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet Allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

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 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250, Vancouver, Canada.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating Embeddings for Modeling Multi-Relational Data. In *Advances in Neural Information Processing Systems*, pages 2787–2795, Lake Tahoe, Nevada, USA.

Johan Bos. 2008. Wide-Coverage Semantic Analysis with Boxer. In *Proceedings of the Conference on Semantics in Text Processing (STEP)*, Research in Computational Semantics, pages 277–286.

Johan Bos and Katja Markert. 2005. Recognising Textual Entailment with Logical Inference. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 628–635, Vancouver, Canada.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A Large Annotated Corpus for Learning Natural Language Inference. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal.

Samuel Broscheit, Kiril Gashteovski, Yanjie Wang, and Rainer Gemulla. 2020. Can We Predict New Facts with Open Knowledge Graph Embeddings? A Benchmark for Open Link Prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2296–2308, Online.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, and Burr Settles. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 1306–1313, Atlanta, Georgia, USA.

Haw-Shiuan Chang, Ziyun Wang, Luke Vilnis, and Andrew McCallum. 2018. Distributional Inclusion Vector Embedding for Unsupervised Hypernymy Detection. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 485–495, New Orleans, Louisiana, USA.

Wenhu Chen, Wenhan Xiong, Xifeng Yan, and William Yang Wang. 2018. Variational Knowledge Graph Reasoning. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1823–1832, New Orleans, Louisiana, USA.

Stephen Clark, Julia Hockenmaier, and Mark Steedman. 2002. Building deep dependency structures using a wide-coverage CCG parser. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 327–334, Philadelphia, Pennsylvania, USA.

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised Learning of Universal Sentence Representations from Natural

Language Inference Data. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark.

Gene Ontology Consortium. 2015. Gene ontology consortium: going forward. *Nucleic acids research*, 43(D1):D1049–D1056.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL Recognising Textual Entailment Challenge. In *Machine Learning Challenges Workshop*, pages 177–190.

Ido Dagan, Lillian Lee, and Fernando C.N. Pereira. 1999. Similarity-Based Models of Word Cooccurrence Probabilities. *Machine learning*, 34(1-3):43–69.

Ido Dagan, Dan Roth, Mark Sammons, and Fabio Massimo Zanzotto. 2013. Recognizing Textual Entailment: Models and Applications. *Synthesis Lectures on Human Language Technologies*, 6(4):1–220.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2018. Go for a Walk and Arrive at the Answer: Reasoning Over Paths in Knowledge Bases using Reinforcement Learning. In *Proceedings of the International Conference on Learning Representations*, Vancouver, Canada.

Rajarshi Das, Arvind Neelakantan, David Belanger, and Andrew McCallum. 2017a. 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*, pages 132–141, Valencia, Spain.

Rajarshi Das, Manzil Zaheer, Siva Reddy, and Andrew McCallum. 2017b. 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, Vancouver, Canada.

Donald Davidson. 1967. The Logical Form of Action Sentences. *Essays on Actions and Events*, 1(9):105–149.

Thomas Demeester, Tim Rocktäschel, and Sebastian Riedel. 2016. Lifted Rule Injection for Relation Embeddings. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1389–1399, Austin, Texas, USA.

Tim Dettmers, Minervini Pasquale, Stenetorp Pontus, and Sebastian Riedel. 2018. Convolutional 2D Knowledge Graph Embeddings. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 1811–1818, Honolulu, Hawaii, USA.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4171–4186, Minneapolis, Minnesota.

Boyang Ding, Quan Wang, Bin Wang, and Li Guo. 2018. Improving Knowledge Graph Embedding Using Simple Constraints. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 110–121, Melbourne, Australia.

Li Dong, Jonathan Mallinson, Siva Reddy, and Mirella Lapata. 2017. Learning to Paraphrase for Question Answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 875–886, Copenhagen, Denmark.

Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 601–610, New York, New York, USA.

Sourav Dutta and Gerhard Weikum. 2015. Cross-Document Co-Reference Resolution using Sample-Based Clustering with Knowledge Enrichment. *Transactions of the Association for Computational Linguistics*, 3:15–28.

Takuma Ebisu and Ryutaro Ichise. 2018. TorusE: Knowledge Graph Embedding on a Lie Group. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 1819–1826, New Orleans, Louisiana, USA.

Bradley Efron and Robert Tibshirani. 1985. The Bootstrap Method for Assessing Statistical Accuracy. *Behaviormetrika*, 12(17):1–35.

Hady Elsahar, Pavlos Vougiouklis, Arslen Remaci, Christophe Gravier, Jonathon Hare, Frederique Laforest, and Elena Simperl. 2018. T-REx: A Large Scale Alignment of

Natural Language with Knowledge Base Triples. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation*, pages 3448–3452, Miyazaki, Japan.

Oren Etzioni, Anthony Fader, Janara Christensen, Stephen Soderland, and Mausam Mausam. 2011. Open Information Extraction: The Second Generation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence*, pages 3–10, Barcelona, Spain.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2014. Open Question Answering over Curated and Extracted Knowledge Bases. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1156–1165, New York, New York, USA.

DA Ferrucci. 2012. Introduction to" This is Watson". *IBM Journal of Research and Development*, 56(3):235–249.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The Paraphrase Database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 758–764, Atlanta, Georgia, USA.

Maayan Geffet and Ido Dagan. 2005. The Distributional Inclusion Hypotheses and Lexical Entailment. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 107–114, Ann Arbor, Michigan, USA.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2016. Jointly Embedding Knowledge Graphs and Logical Rules. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 192–202, Austin, Texas, USA.

Shu Guo, Quan Wang, Lihong Wang, Bin Wang, and Li Guo. 2018. Knowledge Graph Embedding with Iterative Guidance from Soft Rules. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 4816–4823, Honolulu, Hawaii, USA.

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, Lisbon, Portugal.

Zellig S Harris. 1954. Distributional Structure. *Word*, 10(2-3):146–162.

Aurélie Herbelot and Mohan Ganesalingam. 2013. Measuring Semantic Content in Distributional Vectors. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 440–445, Sofia, Bulgaria.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Xavier R. Holt. 2018. Probabilistic Models of Relational Implication. Master's thesis, Macquarie University.

Mohammad Javad Hosseini, Nathanael Chambers, Siva Reddy, Xavier Holt, Shay Cohen, Mark Johnson, and Mark Steedman. 2018. Learning Typed Entailment Graphs with Global Soft Constraints. *Transactions of the Association for Computational Linguistics*, 6:703–717.

Mohammad Javad Hosseini, Shay B Cohen, Mark Johnson, and Mark Steedman. 2019. Duality of Link Prediction and Entailment Graph Induction. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4736–4746, Florence, Italy.

Mohammad Javad Hosseini, Shay B Cohen, Mark Johnson, and Mark Steedman. 2021. Contextual Link Prediction to Learn Relational Entailment Graphs. Under Review.

Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. 2015. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*, pages 687–696, Beijing, China.

Guoliang Ji, Kang Liu, Shizhu He, and Jun Zhao. 2016. Knowledge Graph Completion with Adaptive Sparse Transfer Matrix. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 985–991, Phoenix, Arizona, USA.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How Can We Know What Language Models Know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Aishwarya Kamath and Rajarshi Das. 2019. A Survey on Semantic Parsing. In *Automated Knowledge Base Construction (AKBC)*, Amherst, Massachusetts, USA.

Dimitri Kartsaklis and Mehrnoosh Sadrzadeh. 2016. Distributional Inclusion Hypothesis for Tensor-based Composition. In *Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2849–2860, Osaka, Japan.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *Advances in Neural Information Processing Systems*, pages 4284–4295, Montreal, Canada.

Ross Kindermann and J Laurie Snell. 1980. *Markov Random Fields and their Applications*, volume 1. American Mathematical Society.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the 3rd International Conference on Learning Representations*, San Diego, California, USA.

Philipp Koehn. 2004. Statistical Significance Tests for Machine Translation Evaluation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 388–395, Barcelona, Spain.

Thomas Kollar, Danielle Berry, Lauren Stuart, Karolina Owczarzak, Tagyoung Chung, Lambert Mathias, Michael Kayser, Bradford Snow, and Spyros Matsoukas. 2018. The Alexa Meaning Representation Language. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers)*, pages 177–184, New Orleans, Louisiana, USA.

Daphne Koller, Nir Friedman, Sašo Džeroski, Charles Sutton, Andrew McCallum, Avi Pfeffer, Pieter Abbeel, Ming-Fai Wong, David Heckerman, Chris Meek, et al. 2007. *Introduction to Statistical Relational Learning*. MIT press.

Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. 2017. Neural Semantic Parsing with Type Constraints for Semi-Structured Tables. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1516–1526, Long Beach, California, United States.

Jayant Krishnamurthy and Tom Mitchell. 2012. Weakly Supervised Training of Semantic Parsers. In *Proceedings of the 2012 Joint Conference on Empirical Methods*

in Natural Language Processing and Computational Natural Language Learning, pages 754–765, Jeju Island, Korea.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *Advances in Neural Information Processing Systems*, pages 7059–7069, Vancouver, Canada.

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.

Omer Levy and Ido Dagan. 2016. Annotating Relation Inference in Context via Question Answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 249–255, Berlin, Germany.

Mike Lewis. 2014. *Combined Distributional and Logical Semantics*. Ph.D. thesis, University of Edinburgh.

Mike Lewis and Mark Steedman. 2013a. Combined Distributional and Logical Semantics. *Transactions of the Association for Computational Linguistics*, 1:179–192.

Mike Lewis and Mark Steedman. 2013b. Unsupervised Induction of Cross-Lingual Semantic Relations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 681–692, Seattle, Washington, USA.

Mike Lewis and Mark Steedman. 2014a. A^* CCG Parsing with a Supertag-factored Model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 990–1000, Doha, Qatar.

Mike Lewis and Mark Steedman. 2014b. Combining Formal and Distributional Models of Temporal and Intensional Semantics. In *Proceedings of the ACL Workshop on Semantic Parsing*, pages 28–32, Baltimore, Maryland, USA.

Xiang Li, Luke Vilnis, Dongxu Zhang, Michael Boratko, and Andrew McCallum. 2019. Smoothing the Geometry of Probabilistic Box Embeddings. In *Proceedings of the International Conference on Learning Representations*, New Orleans, Louisiana, USA.

Dekang Lin. 1998. Automatic Retrieval and Clustering of Similar Words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics*, pages 768–774, Montreal, Canada.

Dekang Lin and Patrick Pantel. 2001. Discovery of Inference Rules for Question-Answering. *Natural Language Engineering*, pages 343–360.

Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015a. Modeling Relation Paths for Representation Learning of Knowledge Bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 705–714, Lisbon, Portugal.

Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015b. Learning Entity and Relation Embeddings for Knowledge Graph Completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2181–2187, Austin, Texas, USA.

Xiao Ling and Daniel S. Weld. 2012. Fine-Grained Entity Recognition. In *Proceedings of the National Conference of the Association for Advancement of Artificial Intelligence*, pages 94–100, Toronto, Canada.

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

Yuanfei Luo, Quan Wang, Bin Wang, and Li Guo. 2015. Context-Dependent Knowledge Graph Embedding. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1656–1661, Lisbon, Portugal.

Bill MacCartney and Christopher D Manning. 2009. An Extended Model of Natural Logic. In *Proceedings of the Eight International Conference on Computational Semantics*, pages 140–156, Tilburg, The Netherlands.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119, Lake Tahoe, Nevada, USA.

George A Miller. 1995. WordNet: a Lexical Database for English. *Communications of the ACM*, 38(11):39–41.

George A Miller. 1998. *WordNet: An Alectronic Lexical Database*. MIT Press.

Shashi Narayan, Ronald Cardenas, Nikos Papasarantopoulos, Shay B. Cohen, Mirella Lapata, Jiangsheng Yu, and Yi Chang. 2018. Document Modeling with External Attention For Sentence Extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 2020–2030, Melbourne, Australia.

Dai Quoc Nguyen, Thanh Vu, Tu Dinh Nguyen, Dat Quoc Nguyen, and Dinh Phung. 2019. A Capsule Network-based Embedding Model for Knowledge Graph Completion and Search Personalization. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2180–2189, Minneapolis, Minnesota.

Dat Ba Nguyen, Johannes Hoffart, Martin Theobald, and Gerhard Weikum. 2014. AIDA-light: High-Throughput Named-Entity Disambiguation. In *Workshop on Linked Data on the Web*, pages 1–10, Seoul, Korea.

Dat Quoc Nguyen. 2020. A Survey of Embedding Models of Entities and Relationships for Knowledge Graph Completion. *arXiv preprint arXiv:1703.08098*.

Dat Quoc Nguyen, Kairit Sirts, Lizhen Qu, and Mark Johnson. 2016. STransE: a Novel Embedding Model of Entities and Relationships in Knowledge Bases. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 460–466, San Diego, California, USA.

Tu Dinh Nguyen, Dat Quoc Nguyen, Dinh Phung, et al. 2018. A Novel Embedding Model for Knowledge Base Completion Based on Convolutional Neural Network. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 327–333, New Orleans, Louisiana, USA.

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*, pages 809–816, Bellevue, Washington, USA.

Terence Parsons. 1990. *Events in the Semantics of English: A Study in Subatomic Semantics*. MIT Press, Cambridge, MA.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better Paraphrase Ranking, Fine-Grained Entailment Relations, Word Embeddings, and Style Classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 425–430, Beijing, China.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1532–1543, Doha, Qatar.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, USA.

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

Alexandrin Popescul and Lyle H Ungar. 2003. Statistical Relational Learning for Link Prediction. In *Proceedings of the Workshop on Learning Statistical Models from Relational Data at IJCAI*, pages 109–115, Acapulco, Mexico.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding with unsupervised learning. *Technical report, OpenAI*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas.

Siva Reddy, Mirella Lapata, and Mark Steedman. 2014. Large-Scale Semantic Parsing without Question-Answer Pairs. *Transactions of the Association for Computational Linguistics*, 2:377–392.

Sebastian Riedel and James Clarke. 2006. Incremental Integer Linear Programming for Non-projective Dependency Parsing. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, pages 129–137, Sydney, Australia.

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin M. Marlin. 2013. Relation Extraction with Matrix Factorization and Universal Schemas. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84, Atlanta, Georgia, USA.

Amrita Saha, Vardaan Pahuja, Mitesh M. Khapra, Karthik Sankaranarayanan, and Sarath Chandar. 2018. Complex Sequential Question Answering: Towards Learning to Converse Over Linked Question Answer Pairs with a Knowledge Graph. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 705–713, Honolulu, Hawaii, USA.

Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling Relational Data with Graph Convolutional Networks. In *Proceedings of the European Semantic Web Conference*, pages 593–607, Heraklion, Crete, Greece.

Martin Schmitt and Hinrich Schütze. 2019. SherLIiC: A Typed Event-Focused Lexical Inference Benchmark for Evaluating Natural Language Inference. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 902–914, Florence, Italy.

Stefan Schoenmackers, Oren Etzioni, Daniel S. Weld, and Jesse Davis. 2010. Learning First-Order Horn Clauses From Web Text. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1088–1098, Cambridge, Massachusetts, USA.

Chao Shang, Yun Tang, Jing Huang, Jinbo Bi, Xiaodong He, and Bowen Zhou. 2019. End-to-End Structure-Aware Convolutional Networks for Knowledge Base Completion. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, volume 33, pages 3060–3067, Honolulu, Hawaii, USA.

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, Montreal, Canada.

Baoxu Shi and Tim Weninger. 2017. ProjE: embedding projection for knowledge graph completion. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, pages 1236–1242, San Francisco, California, USA.

Richard Socher, Danqi Chen, Christopher D. Manning, and Andrew Ng. 2013. Reasoning with Neural Tensor Networks for Knowledge Base Completion. In *Advances in Neural Information Processing Systems*, pages 926–934, Lake Tahoe, Nevada, USA.

Robyn Speer and Catherine Havasi. 2012. Representing General Relational Knowledge in ConceptNet 5. In *Proceedings of the Eighth International Conference on Language Resources and sEvaluation*, pages 3679–3686, Istanbul, Turkey.

Mark Steedman. 2000. *The Syntactic Process*. MIT Press, Cambridge, MA.

Mark Steedman and Jason Baldridge. 2011. Combinatory categorial grammar. *Non-Transformational Syntax: Formal and Explicit Models of Grammar*, pages 181–224.

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, Banff, Canada.

Idan Szpektor and Ido Dagan. 2008. Learning Entailment Rules for Unary Templates. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 849–856, Manchester, UK.

Idan Szpektor, Hristo Tanev, Ido Dagan, and Bonaventura Coppola. 2004. Scaling Web-based Acquisition of Entailment Relations. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 41–48, Barcelona, Spain.

Kristina Toutanova, 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*, pages 1434–1444, Berlin, Germany.

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, Vancouver, Canada.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex Embeddings for Simple Link Prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning*, pages 2071–2080, New York City, New York, USA.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, Long Beach, California, USA.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. 2015. Order-Embeddings of Images and Language. *arXiv preprint arXiv:1511.06361*.

Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. 2018. Probabilistic Embedding of Knowledge Graphs with Box Lattice Measures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 263–272, Melbourne, Australia.

Quan Wang, Bin Wang, and Li Guo. 2015a. Knowledge Base Completion using Embeddings and Rules. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, pages 1859–1865, Buenos Aires, Argentina.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015b. Building a Semantic Parser Overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics*, pages 1332–1342, Beijing, China.

Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge Graph Embedding by Translating on Hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, pages 1112–1119, Quebec City, Canada.

Julie Weeds and David Weir. 2003. A General Framework for Distributional Similarity. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 81–88, Sapporo, Japan.

Robert West, Evgeniy Gabrilovich, Kevin Murphy, Shaohua Sun, Rahul Gupta, and Dekang Lin. 2014. Knowledge Base Completion via Search-based Question Answering. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 515–526, Milan, Italy.

Ronald J Williams. 1992. Simple Statistical Gradient-following Algorithms for Connectionist Reinforcement Learning. *Machine learning*, 8(3-4):229–256.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing. *ArXiv*, pages arXiv–1910.

Yangyang Xu and Wotao Yin. 2013. A Block Coordinate Descent Method for Regularized Multiconvex Optimization with Applications to Nonnegative Tensor Factorization and Completion. *SIAM Journal on Imaging Sciences*, 6(3):1758–1789.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2015. Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *Proceedings of the International Conference on Learning Representations*, San Diego, California, USA.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized Autoregressive Pretraining for Language Understanding. In *Advances in Neural Information Processing Systems*, pages 5753–5763, Vancouver, Canada.

Mihalis Yannakakis. 1978. Node-and Edge-Deletion NP-Complete Problems. In *Proceedings of the Tenth Annual ACM Symposium on Theory of Computing*, pages 253–264, New York City, New York, USA.

Liang Yao, Chengsheng Mao, and Y. Luo. 2019. KG-BERT: BERT for Knowledge Graph Completion. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence*, pages 2901–2908, New York City, New York, USA.

Xuchen Yao and Benjamin Van Durme. 2014. Information Extraction over Structured Data: Question Answering with Freebase. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 956–966, Baltimore, Maryland, USA.

Alexander Yates and Oren Etzioni. 2009. Unsupervised Methods for Determining Object and Relation Synonyms on the Web. *Journal of Artificial Intelligence Research*, 34:255–296.

Pengcheng Yin, Nan Duan, Ben Kao, Junwei Bao, and Ming Zhou. 2015. Answering Questions with Complex Semantic Constraints on Open Knowledge Bases. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pages 1301–1310, Melbourne, Australia.

Naomi Zeichner, Jonathan Berant, and Ido Dagan. 2012. Crowdsourcing Inference-Rule Evaluation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 156–160, Jeju, Korea.

Luke Zettlemoyer and Michael Collins. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. In *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, pages 658–666, Edinburgh, Scotland.

Congle Zhang, Stephen Soderland, and Daniel S Weld. 2015. Exploiting Parallel News Streams for Unsupervised Event Extraction. *Transactions of the Association for Computational Linguistics*, 3:117–129.

Congle Zhang and Daniel S. Weld. 2013. Harvesting Parallel News Streams to Generate Paraphrases of Event Relations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1776–1786, Seattle, Washington, USA.

Hongming Zhang, Yan Song, Yangqiu Song, and Dong Yu. 2019. Knowledge-aware Pronoun Coreference Resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 867–876, Florence, Italy.