Machine Learning Approaches for Multi-hop Reasoning Over Relational Knowledge

関係知識上でのマルチホップ推論のための
機械学習アプローチ

Ryo Takahashi

Graduate School of Information Sciences
Tohoku University

This dissertation is submitted for the degree of
Doctor of Information Science

January 2021

Acknowledgements

I would like to express my sincere gratitude to my major advisor, Professor Kentaro Inui, for his support and guidance in carrying out this research and completing my dissertation. I would also like to express my sincere gratitude for his warm and patient observation during the six years since I was assigned to his laboratory in my third year of undergraduate studies, and for his great support and advice throughout my research life.

I would like to express my sincere gratitude to Professor Okatani Takayuki and Professor Masanori Hariyama for reviewing this paper as reviewers.

I would like to express my sincere gratitude to Professor Naoya Inoue for his guidance during the six years since I was assigned to the laboratory. I would like to express my deepest gratitude to him.

I would also like to express my deepest gratitude to the many people who have supported me in my research but whose names I have not mentioned here.

Finally, I would like to express my heartfelt gratitude to my mother, who have always supported and watched me warmly.

Abstract

The ability to reason over relational knowledge is central to general intellectual behavior. Relational knowledge here refers to the relationship between ”things” and ”things.” More specifically, one relational knowledge is represented by ⟨head entity, relation, tail entity⟩. For example, Wikidata, which is one of the databases, contains knowledge such as ⟨*TensorFlow*, *developer*, *Google Brain*⟩ and ⟨*Google Brain*, *field_of_work*, *Machine Learning*⟩. Inference on relational knowledge refers to using known relational knowledge to predict unknown relational knowledge. For example, it can be used to predict what Chainer will be used for. This kind of problem formulation, where we try to predict the missing entities, is specifically called Knowledge Graph Completion or Knowledge Base Completion.

In this research, we address the problem of how to learn multi-hop inference, especially in inference over relational knowledge. Multi-hop reasoning here refers to hopping over relational knowledge multiple times in order to find an answer. For example, from known relational knowledge, we can learn that the relation *used_for* is highly similar to the 2-hop path *developer/field_of_work*. In this case, if there are triples ⟨*Chainer*, *developer*, *PFN*⟩ and ⟨*PFN*, *field_of_work*, *Machine Learning*⟩, one can infer ⟨*Chainer*, *used_for*, *Machine Learning*⟩.

In this paper, we address two major problem settings for learning multi-hop reasoning. One is multi-hop reasoning in predicate logic, and the other is multi-hop reasoning in propositional logic. We propose frameworks and models for learning multi-hop reasoning over real-world knowledge bases, and explore their practical applicability.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

The ability to reason over relational knowledge is central to general intellectual behavior. Relational knowledge here refers to the relationship between "things" and "things." More specifically, one relational knowledge is represented by $\langle$head entity, relation, tail entity$\rangle$. For example, Wikidata, which is one of the databases, contains knowledge such as $\langle$*TensorFlow*, *developer*, *Google Brain*$\rangle$ and $\langle$*Google Brain*, *field_of_work*, *Machine Learning*$\rangle$. Inference on relational knowledge refers to using known relational knowledge to predict unknown relational knowledge. For example, it can be used to predict what Chainer will be used for. This kind of problem formulation, where we aim to predict the missing entities, is specifically called Knowledge Graph Completion or Knowledge Base Completion.

Most existing research models inference in a knowledge graph as an operation on a low-dimensional vector space, where the embeddings of entities and relations are learned from a set of triples (Wang et al., 2017). However, previous studies have shown that the performance on artificial benchmark datasets, where the locally dense part of the knowledge graph is extracted, is high, while the performance on sparse situations, such as those found in real-world knowledge graphs, is limited (Pujara et al., 2017).

Due to the background, there are ongoing researches to incorporate additional information during learning of knowledge graph embedding for more realistic situations. The one is to consider multiple sequences of relations (relation paths) to enable multi-hop reasoning. For example, to infer $\langle$*Tensorflow*, *used_for*, *Machine Learning*$\rangle$ from $\langle$*Tensorflow*, *developer*, *Google Brain*$\rangle$ and $\langle$*Google Brain*, *field_of_work*, *Machine Learning*$\rangle$, the system learns to make the embedding of the relational path *developer/field_of_work* closer to the embedding of *used_for*. The other is integration with textual information Toutanova et al. (2015). For example, if we can capture the fact that the relation *developer* is likely to appear as "*A is developed by B.*" in the text, we can infer $\langle$*Chainer*, *used_for*, *Machine Learning*$\rangle$ by combining it with the above triplet using the text "*Chainer is developed by PFN.*" as a clue, even if ⟨*Chainer*, *developer*, *PFN*⟩ is not stored in the knowledge graph. Although these two directions have been studied independently in previous research, they are complementary, and by combining the relational knowledge in the knowledge graph with the relational knowledge extracted from the text, there is a possibility of further improving the inference.

In this paper, we address two major problem settings for learning multi-hop reasoning. One is multi-hop reasoning in predicate logic, and the other is multi-hop reasoning in propositional logic. We propose frameworks and models for learning multi-hop reasoning over real-world knowledge bases, and explore their practical applicability.

The rest of this thesis is structured as follows.

- Chapter 2 discusses potential risk prediction in traffic scenes as an example of multi-hop reasoning over predicate logic.
- Chapter 3 introduces knowledge graph completion as an example of multi-hop reasoning in propositional logic.
- Chapter 4 focuses on the sparseness of knowledge graphs and discusses joint representation learning of texts and knowledge graphs.
- Chapter 5 summarizes our discussion and presents our future direction.

Chapter 2

Explaining Potential Risks in Traffic Scenes by Combining Logical Inference and Physics Simulation

The automatic recognition of risks in traffic scenes is a core technology of Advanced Driver Assistance Systems (ADASs). Most of the existing work on traffic risk recognition has been conducted in the context of motion prediction of vehicles. Thus, existing systems rely on directly observed information (e.g., velocity), whereas the exploitation of implicit information inferable from observed information (e.g., the intention of pedestrians) has rarely been explored. Our previous approach used abductive reasoning to infer implicit information from observation and jointly identify the most-likely risks in traffic scenes. However, abductive frameworks do not exploit quantitative information explicitly, which leads to a lack of grounds for physical quantities. In this work, we propose a novel risk recognition model combining first-order logical abduction-based symbolic reasoning with a simulation-based on physical quantities. We build a novel benchmark dataset of real-life traffic scenes that are potentially risky. Our evaluation demonstrates the potential of our approach. The developed dataset has been made publicly available for research purposes.

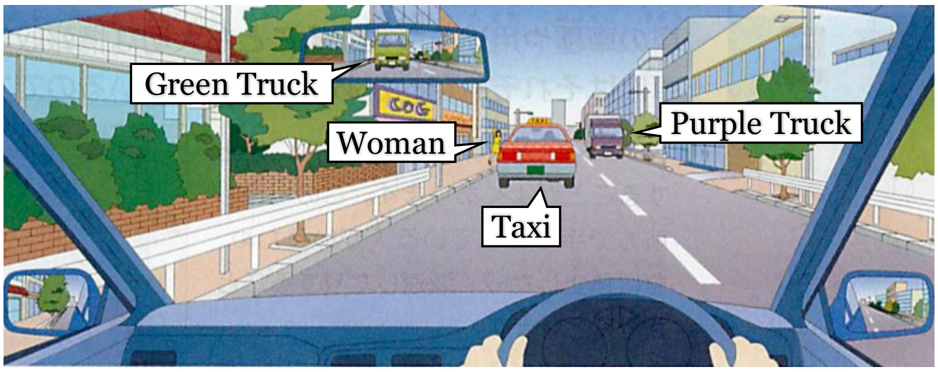
2.1 Introduction

In the field of automotive safety, technology for advanced driver assistance systems (ADASs) and automated driving systems has received much attention Bengler et al. (2014); Lefèvre et al. (2014); Rendon-Velez et al. (2009). One of the crucial, open issues in this field is how to make the system capable of making an early prediction of potential risks from every frame of a traffic scene.

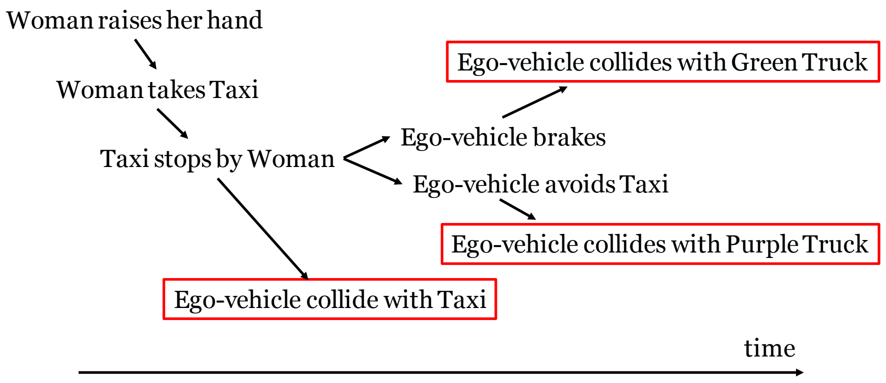
Let us consider the traffic scene illustrated in Figure 2.1a, where an individual is driving on a street and a red taxi is driving ahead of them. From this scene, the woman in yellow may signal for a taxi ride, and the taxi driver may suddenly stop to pick up the woman. This can be a potential risk for the individual if they suddenly brake and the green truck hits them from behind. Furthermore, this can be a risk because the individual cannot overtake the taxi due to the purple truck driving in the opposite lane.

A key to avoid such risks is to predict them as early as possible. However, early prediction of potential risks is not straightforward because it requires prediction of the behavior or intentions of pedestrians and vehicles (e.g. stopping a taxi) in a given traffic scene. Furthermore, as illustrated in Figure 2.1b, it requires reasoning about their plausible results through chains of causalities. Another requirement is that it is necessary to consider physical quantities for reasoning with such chains. In Figure 2.1, the likelihood of reasoning that the individual might be hit by the green truck by braking suddenly is dependent on the distance and the relative speed between the individual and the truck. While these requirements provide an intriguing application-oriented instance of the task of commonsense reasoning over the human-physical world, few studies regarding early prediction of potential risks exist in the field of automated driving and ADASs.

Regarding early prediction of potential risks, there are some studies on inference-based approaches Armand et al. (2014); Mohammad et al. (2015); Zhao et al. (2015). However, the inference engines used in these studies are deductive. As described below, this cannot handle the uncertainty of observation. On the other hand, in our previous work Inoue et al. (2015), we proposed a context-aware risk prediction model that exploits first-order logic-based abductive reasoning. An abductive framework allows us to predict long-range movements of traffic objects by using implicit contextual information and simultaneously provides deeper explanations as to why a traffic scene has a risk. However, abductive frameworks do not exploit quantitative information explicitly, which leads to a lack of grounds for physical quantities. Our previous work also pointed out that the majority of the erroneous traffic risks are derived via unreasonable inference rules, which are caused by a lack of physical information such as the precise positions and directions of traffic objects. For example, the system needed to understand that if a bus was currently stopped at a bus stop, and a man



(a) A risky traffic scene. (This illustration was cited from kik (1999).)


Woman raises her hand
Woman takes Taxi
Taxi stops by Woman
Ego-vehicle collides with Green Truck
Ego-vehicle brakes
Ego-vehicle avoids Taxi
Ego-vehicle collides with Purple Truck
Ego-vehicle collide with Taxi
time

(b) A causality chains of above traffic scene. The red rectangles denote a potential risk.

Fig. 2.1 What is dangerous about this traffic scene?

across the street appeared to be interested in crossing the street to ride the bus, then the man may suddenly cross the street to catch the bus.

In this study, we integrate a symbolic inference-based approach and a physics simulation. We rebuild our previous knowledge-base in order to connect it with physics simulation. We expect our approach to perform well on risky traffic scenes that are needed to exploit quantitative information.

To evaluate our risk recognition system, we build a novel benchmark dataset of real-life traffic scenes that have a potential risk from the existing near-miss database. To the best of our knowledge, it is the largest dataset (over 3,000 scenes) of risk prediction based on real-life data. We conducted a corpus study on the dataset to select scenes that are needed to predict risks. Our preliminary evaluation results on a subset of the corpus suggest that the proposed integrated architecture provides rich informations for early prediction of potential risks in real-life traffic scenes.

2.2 Background

2.2.1 Related Work

A majority of studies concerning inner-city risk assessment are based on detecting possible conflicts of future trajectories Lefèvre et al. (2014). Broadhurst et al. Broadhurst et al. (2005) used the so-called Monte Carlo method to generate a probability distribution for the possible future motion of every vehicle in the scene to avoid danger. Althoff et al. Althoff et al. (2009) have proposed a stochastic approach to detect forthcoming collisions. These works, as well as other works, do not address the explicit interaction among traffic participants, although its importance has been indicated in Rendon-Velez et al. (2009).

Several symbolic inference-based approaches have been proposed for understanding situations in an inner-city context. Armand et al. Armand et al. (2014) formulated an ontology for inner-city traffic situation analysis and created rules which enable reasoning on the traffic participants' future behavior. Furthermore, they showed that the ontology makes it possible to understand which are the key entities that a driver should consider. Mohammad et al. Mohammad et al. (2015) proposed an ontology-based framework for assessing the degree of risk in a road scene in which there are vehicles or pedestrians and indicated that the framework is capable of assessing risk with high accuracy. Zhao et al. Zhao et al. (2015) proposed an ontology-based knowledge base and a decision-making system capable of making safe decisions on uncontrolled intersections and two-way narrow roads.

These approaches take into account the characteristics of the environment and the interactions among them. The advantage of using a symbolic inference-based approach is the transparency of the system: the prediction result is represented by a combination of prediction rules. The rules can be used for explaining the reason of the prediction, which has recently become an important research topic of ADASs. However, the inference engines employed in previous work is *deductive*, which cannot handle the uncertainty of observations. Moreover, symbolic inference-based approaches occasionally overgeneralize the physical world, as the prediction is not based on precise physical prediction.

Grounding technologies, including image/motion recognizers and radars, are considered to recently becoming advanced Ambardekar et al. (2014). For object recognizers, a number of benchmark datasets are publicly available Dollár et al. (2009); Enzweiler and Gavrila (2009); Ess et al. (2008); Geiger et al. (2012), and they have been extensively studied over the years. Zhang et al. Zhang et al. (2016) compare around 10 pedestrian detectors on the Caltech-USA pedestrian benchmark Dollár et al. (2012) and report that the best method Checkerboards achieve an 18.47 % miss-rate. In fact, these technologies have already been applied to traffic scene understanding Souza and Santos (2011). Regarding other grounding technologies, such as radars and vision cameras, extensive research has also been done (see Bengler et al. Bengler et al. (2014) for a detailed overview). However, the accuracy is not always perfect: handling uncertainty of observations is important.

To the best of our knowledge, there is no previous work that focuses on integrating logical inference that makes maximum use of symbolic information and simulation that exploits quantitative data.

2.2.2 Abduction

Abduction is inference to the best explanation and is widely used for knowledge-based symbolic inference systems such as diagnosis systems or natural language understanding Hobbs et al. (1993) in artificial intelligence research. Formally, first-order logical abduction is defined as follows:

Given: Background knowledge B, and observations O, where B is a set of first-order logical formulae, and O is a set of literals or substitutions,

Find: An hypothesis H such that $H \cup B \models O$, $H \cup B \not\models \perp$, where H is a set of literals or substitutions.

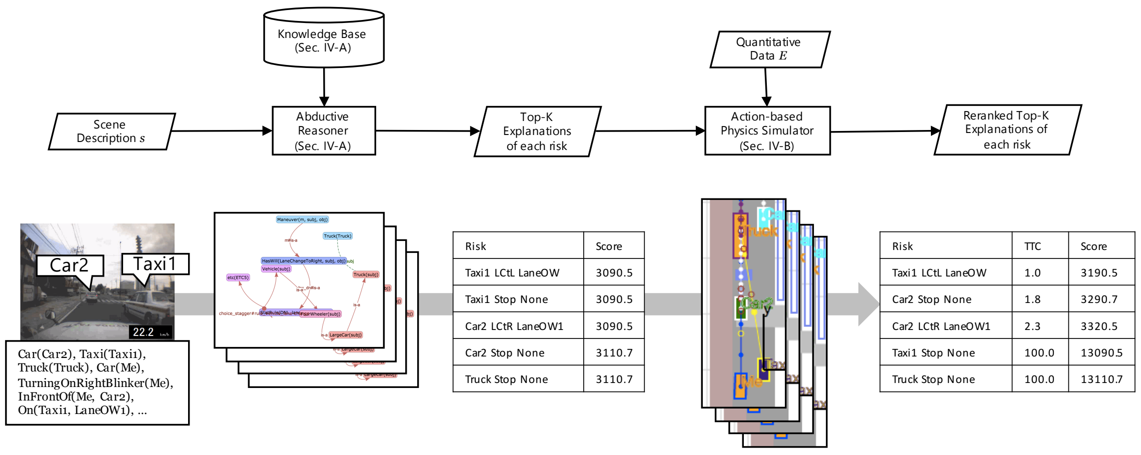
Each of hypothesis H that satisfies the condition is called a *candidate hypothesis*, and denote a set of candidate hypotheses as $\mathscr{H}$. The goal of abduction is to find the best explanation[1] among candidate hypotheses by a specific evaluation measure. In this paper, we formulate abduction as the task of finding the minimum-cost explanation $\hat{H}$ among $\mathscr{H}$. Formally, we find $\hat{H} = \arg\min_{H \in \mathscr{H}} Cost(H)$, where, $Cost$ is a function that maps each $H \in \mathscr{H}$ to a real number, which is called the *abductive cost function*. We elaborate our cost function in Section 2.4.1.

2.3 Task Definition

In this paper, we formalize the problem of traffic risk recognition as follows:

Given: A scene description s of a traffic scene which has potential risks with respect to the ego-vehicle, and quantitative data E of each entity in the scene, where s is a set of literals in first-order predicate logic following the knowledge representation described in Sec. 2.4.1, and E is a set of triples of the form $(shape, position, velocity)$,

[1]In the context of abduction, the term *explanation* and *hypothesis* are used interchangeably.


Knowledge Base (Sec. IV-A)
Quantitative Data E
Scene Description s
Abductive Reasoner (Sec. IV-A)
Top-K Explanations of each risk
Action-based Physics Simulator (Sec. IV-B)
Reranked Top-K Explanations of each risk
Car2
Taxi1
22.2 km/h
Car(Car2), Taxi(Taxi1), Truck(Truck), Car(Me), TurningOnRightBlinker(Me), InFrontOf(Me, Car2), On(Taxi1, LaneOW1), ...
Maneuver(m, subj, obj)
Truck(Truck)
HasWill(LaneChangeToRight, subj, obj)
Vehicle(subj)
etc(ETC5)
Truck(subj)
FourWheeler(subj)
LargeCar(subj)
Truck
Car2
Me

Risk	Score
Taxi1 LCtL LaneOW	3090.5
Taxi1 Stop None	3090.5
Car2 LCtR LaneOW1	3090.5
Car2 Stop None	3110.7
Truck Stop None	3110.7

Risk	TTC	Score
Taxi1 LCtL LaneOW	1.0	3190.5
Car2 Stop None	1.8	3290.7
Car2 LCtR LaneOW1	2.3	3320.5
Taxi1 Stop None	100.0	13090.5
Truck Stop None	100.0	13110.7

Fig. 2.2 A diagram of our system and an example. LCtL and LCtR in the table denote “lane change to left” and “lane change to right” respectively. The lower the score is, the more risky action is.

Find: The best explanation of the risk: a set R of potential risks, where each potential risk r consists of an entity-action tuple (e, a).

This study assumes that s and E are constructed from the outputs of perception systems such as Light Detection and Rangings (LIDARs) and object recognition technologies.

2.4 Proposed Approach

As mentioned in Sec. I, the prediction of the behavior or intentions of traffic agents is crucial in the task of traffic risk prediction. Some prior studies, including our previous work Inoue et al. (2015), proposed symbolic inference-based approaches to predict such information Armand et al. (2014); Mohammad et al. (2015); Zhao et al. (2015). We employ an abduction-based approach proposed in our previous work Inoue et al. (2015) as the starting point of this study because our previous work can handle the uncertainty of observation, which is considered to be more advantageous for practical situations. We then overcome the significant drawback of the previous work: the previous work did not exploit the quantitative information such as the shape, position, velocity of traffic agents. To solve this problem, we propose a method to plug a physics simulator into a symbolic inference engine. More specifically, we rebuild the knowledge base in previous work so that it can predict richer information which is usable for a physics simulator, and show how to combine the symbolic inference with a physics simulator.

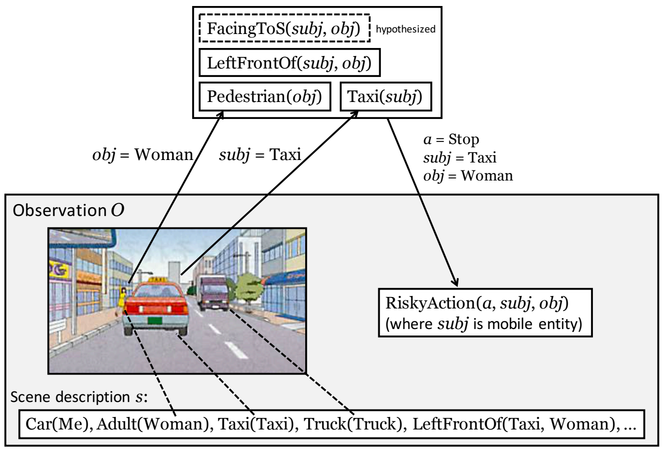


Fig. 2.3 Working example of Action Recognition as Abduction.

We give a brief overview of our approach. Our overall traffic risk recognition architecture is shown in Figure 2.2. Firstly, an abductive reasoner predicts multiple risky entities and their actions from the point of view the ego-vehicle with the scores, using s and a *qualitative* knowledge base as an input. Note that this module does not use precise quantitative information: the distance between ego-vehicle and mobile entities, velocity, and so on. Secondly, an action-based physics simulator simulates the former prediction exploiting quantitative data E on a virtual space, and then outputs metricies such as time-to-collision (TTC). We expect this module to determine whether the former prediction is indeed risky for ego-vehicle.

For example, in the traffic scene illustrated in Figure 2.2, an abductive reasoner predicts that it is most dangerous that Taxi1, on the neighboring lane, might stop suddenly. However, our system notices that it is not necessarily dangerous because ego-vehicle should not collide with the taxi by simulating the situation, so it ranks lower in the scene. In the rest of this section, the further detail of each component is described.

2.4.1 Action Recognition as Abduction

Following our previous work Inoue et al. (2015), we formulate the risk prediction problem as the problem of abductive inference: the problem of explaining why an observed traffic scene and a hypothetical observation "the observed scene has some risk." are observed, using a knowledge base. The knowledge base contains two types of axioms: (i) conceptual hierarchy and (ii) knowledge about relation between intention and its implied situation.

We describe the overall framework using the working example illusrated in Figure 2.3. The observation O includes the logical forms of the observed scene (i.e., *Car*(*Me*), *Adult*(*Woman*), ...) and the hypothetical observation of the existence of a risky entity (i.e., *RiskyAction*(a, $subj$, obj)).

The literal *RiskyAction*(a, $subj$, obj) indicates that a traffic agent $subj$ will take a risky action a to a target obj. We then feed this observation to the abductive inference system to obtain the best explanation, which will contain the variable binding of a, $subj$ and obj. In this example, what happens is that: (i) *RiskyAction*(a, $subj$, obj) is explained by *FacingToS*($subj$, obj) $\wedge$ *LeftFrontOf*($subj$, obj) $\wedge$ *Pedestrian*(obj) $\wedge$ *Taxi*($subj$), and (ii) *Pedestrian*(obj) and *Taxi*($subj$) are explained by the observed scene, assuming $subj$ = *Woman* and obj = *Taxi*. As a result, we can identify that the risky factor of this scene is the sudden stop of the taxi besides the woman.

The main advantage of using abduction is characterized as its declarative nature. We can abstract away from the process of inferences, concentrating on creating a sophisticated knowledge base in the declarative fashion.

The second advantage is that by combining several types of knowledge bases (e.g., causality and ontological knowledge), our model can abductively infer implicit information from observed information and jointly identify the most-likely risks in traffic scenes. For example, in Figure 2.3, FacingToS(Taxi, Woman) is implicit inferred information that is not captured by observations. Abduction-based modeling allows us to predict long-range movements of traffic objects by using implicit contextual information, and simultaneously provide deeper explanations as to why the traffic scene has a risk.

Knowledge Base

We now elaborate on the knowledge base used in this study. We first describe the knowledge representation about a traffic scene. In principle, our representation consists of the following concepts:

- Type of object: e.g., *Vehicle*(x), *Pedestrian*(x);
- Poperty of traffic objects (e.g., whether right blinker is turned on): e.g., *TurningOnLeftBlinker*(x);
- Relation between traffic objects (e.g., relative position): e.g., *InFrontOf*(x, y), *RightOf*(x, y);
- The definition of possible action of pedestrian and vehicle (e.g., turning right), which are represented by constants: *Stop*, *GoLeft*.

Based on this knowledge representation, we constructed the following two types of axioms:

1. Conceptual hierarchy: This represents the hierarchical (a.k.a IS-A) structure of concepts. For example, the knowledge "a taxi is one kind of a car" is represented by the logical form $\forall x.$ Taxi(x) $\Rightarrow$ Car(x). This knowledge allows us to perform reasoning on various levels of abstraction.

2. Intention-situation axiom: The axiom describes the causal relation between an action and the situation where the action is likely to be taken. For example, the knowledge "a vehicle v is likely to overtake a large vehicle c_l which belongs to v's lane l and is in front of v" is represented by the logical form $\forall v, c_l, l.\ \mathit{Vehicle}(v) \wedge \mathit{LargeCar}(c_l) \wedge \mathit{InFrontOf}(v, c_l) \wedge \mathit{On}(v, l) \wedge \mathit{On}(c_l, l) \Rightarrow \mathit{RiskyAction}(\mathit{Overtake}, v, c_l)$.

As described in Sec. 2.4.2, a physics simulator requires an entity-action-object tuple as an input. The knowledge base in our previous work Inoue et al. (2015) is tailored for predicting an risky entity-action tuple, which is insufficient to integrate a physics simulator with symbolic inference. Popular machine learning approaches for classification or ranking also suffers from predicting this kind of richer information. By using first-order logic as a representation, we can easily handle a richer information structure.

Cost function

We employ the cost function of Weighted Abduction Hobbs et al. (1993) as the abductive cost function. In weighted abduction, observation O and hypothesis H are represented by the conjunction of existentially quantified literals. Each literal has a positive real-valued *cost* (henceforth, referred to as $l^{\$100}$). The cost of observation handles the uncertainty of observations. Background knowledge B is a set of Horn clauses. Each literal in the body of Horn clauses is assigned a positive real-valued *weight* (referred to as $l_1^{0.6} \wedge l_2^{0.6} \Rightarrow l_3$).

We then describe the cost function. Let $nonexp(H)$ be a set of non-explained literals in H. In weighted abduction, the cost of a hypothesis H is defined by the sum of non-explained literals:

$$Cost_{WA}(H) = \sum_{h \in nonexp(H)} cost(h), \tag{2.1}$$

where $cost(h)$ is a cost of a literal h. If h is a non-observed literal, $cost(h)$ is calculated by $cost(obs(h)) \cdot \prod_{a \in axioms(h)} weight(a)$, where $axioms(h)$ is the set of axioms used for deriving h, and $obs(h)$ is the observed literal backchained on to hypothesize h. If h is an observed literal, $cost(h)$ is simply a real-valued cost attached to h in the input. See Hobbs et al. (1993) for further details. In this study, we extend equation (2.1) in two ways.

Learning reliabiity of axioms Hobbs et al. (1993) did not provide a method to learn the parameters of cost function. Following our previous work Inoue et al. (2015), in order to learn the reliability of axioms, we extend equation (2.1) as follows:

$$Cost(H) = Cost_{WA}(H) + \mathbf{w_a} \cdot \Phi_a(H), \tag{2.2}$$

where $\Phi_a(H)$ is a feature vector that is constructed from the set of axioms used for deriving H, and $\mathbf{w_a}$ is a real-valued weight vector.

Context-dependency According to the cost function, the goodness of hypothesis depends on the plausibility of axioms used for deriving H and the uncertainty of observations, but *not* on the observed context. However, this is not suitable for traffic risk prediction. To incorporate the context-dependency, we further extend equation (2.2) as follows:

$$Cost(H) = Cost_{WA}(H) + \mathbf{w_a} \cdot \Phi_a(H) + \mathbf{w_c} \cdot \Phi_c(H, O), \tag{2.3}$$

where $\Phi_c(H, O)$ is a feature function that returns a d-dimensinal vector which is determined by a given hypothesis H and an observation O, and $\mathbf{w_c}$ is a real-valued weight vector.

In this study, we take a two-step supervised learning approach to learn $\mathbf{w_c}$ and $\mathbf{w_a}$. We first learn $\mathbf{w_c}$ by using Ranking SVM Joachims (2002), where all the features are binary features encoding (i) literals describing a ranked object and action (prefixed with "obj" and "action_") and (ii) literals describing the other traffic objects in a traffic scene (prefixed with "context_"). Since the combinations of features are considered important for risk prediction, we used a polynomial kernel of degree 2. We then use a latent structured perceptron approach Sun et al. (2009) to learn $\mathbf{w_a}$, fixing $\mathbf{w_c}$ and using the binary feature function that returns a 0-1 vector where the value of i-th element is 1 if i-th axiom is used in H; 0 otherwise as Φ_a. In our experiment, we use Phillip (Yamamoto et al., 2015) as an abductive inference engine[2].

To make inference tractable, we extend the cost function of Weighted Abduction as follows: (1) the cost of unification between hypothesized literals is ∞, (2) the cost of backward inference on observed literals except `action/3` is ∞. This amounts to performing a best-proof search for a literal `action/3` using $O \cup B$ as a background knowledge base.

2.4.2 Action-based Physics Simulations

In addition to qualitative inference described so far, we use physics simulation to rerank the risk prediction results based on the physical information such as location, distance, and velocity. We assume the input and output of a physics simulator as follows:

Input: (i) a road structure, (ii) the quantitative information of traffic agents (i.e., position, direction, velocity), and (iii) the intention of each object represented by a first-order

[2]https://github.com/naoya-i/phillip

logic literal (e.g., *RiskyAction*(*Stop*, *Car*, *YellowSignal*) for "Car will stop before YellowSignal.");

Output: (i) the predicted future trajectories of each traffic agent, and (ii) information about collision between traffic agents (e.g. time-to-collision (TTC)).

As illustrated in Figure 2.2, the physics simulator receives an output from the abductive reasoner, which contains a risky entity-action-object tuple, which is required by the physics simulator input. Using this information, there could be some possible ways to combine physics simulation with symbolic inference. In this chapter, we introduce a simple, pipeline reranking model as a preliminary study for this new challenge. More advanced and complicated combination methods will be explored in future work.

After running physics simulation, we simply rerank risky entity-action-object tuples based on the TTC with the ego-vehicle because we want to detect an entity-action-object tuple which brings a risk *to the ego-vehicle*. The re-ranking score function for a risk $r = (e, a)$ after n seconds is then defined as follows:

$$\begin{aligned} Score_{\mathrm{TTC}}(r, n) = {} & Cost(H_r) \\ & + w \cdot \begin{cases} |n - TTC| & \text{if } e \text{ collided with ego-vehicle} \\ \alpha & \text{otherwise,} \end{cases} \end{aligned}$$

where H_r is a hypothesis associated with r. We set $w = 10$ as a result of manual adjustment on a development set. We empirically set α as 10.

To implement the physics simulator, we use an action-based motion model using prototype trajectories Lefèvre et al. (2014). Given an input, a physics simulator generates a trajectory of each traffic object based on a predefined set of prototype trajectories, where a prototype trajectory contains landmarks and a parametric trajectory represented by a set of points and acceleration. We combine several prototype trajectories to generate a final trajectory, transforming these prototype trajectories by scale and an angle. We use 14 prototype trajectories as a preliminary study.

2.5 Evaluation

Our evaluation is two-folded. The purpose of the first evaluation is to check if the proposed model can enrich prediction results without hurting a simple statistical ranking model. The second evaluation aims at seeing the effectiveness of physics simulation integration. Our previous work Inoue et al. (2015) evaluated the model on a non-realistic small dataset (i.e.,

Table 2.1 Classification of cause of braking.

Label	Cause	#	Freq. (%)
Rule	Traffic rules (e.g., red light traffic signals)	209	20.9
Avoidance	To avoid imminent collisions (e.g., a leading vehicle brakes suddenly)	544	54.5
Prediction	To be on the safe side (e.g., overtaking a bicycle)	166	16.6
Other	Changing lane, entering road, etc.	49	4.9
Unknown	Cannot judge cause	31	3.1

Table 2.2 Classification of behavior of the ego-vehicle.

Label	#	Freq. (%)
Direct	379	53.4
Change	281	39.6
Other	50	7.0

illustrations from a textbook), but we evaluate our model on a large database of near-miss recordings which is collected by the taxis' drive recorders.

2.5.1 Task Setting

Given a traffic scene two seconds before the actual near-miss, our task is to identify a risky action-object tuple that causes the near-miss. In this experiment, we assume the input to our task is a 2-dimensional bird-view map that represents the traffic scene two seconds before the actual near-miss.

The 2-dimensional map contains information about the road structure, traffic objects (position and its direction). The physics simulator uses this information to perform physics simulation. The logical representation of each traffic scene is automatically generated from the map according to Sec. 2.4.1. In this experiment, we assume that the accuracy of sensor technologies is perfect, in order to focus on exploring the methodology of risk prediction.

We evaluate our results in the framework of ranking task. We use $Acc@k$ (Accuracy at k) as a metric, which is the fraction of problems where the correct prediction is made within rank k.

Table 2.3 Accuracy of risk prediction models.

	Validation			Test		
Model	Acc@1	Acc@3	Acc@5	Acc@1	Acc@3	Acc@5
BASELINE	**52.8 (38/72)**	80.6 (58/72)	**90.3 (65/72)**	55.6 (40/72)	**77.8 (56/72)**	**93.1 (67/72)**
INFERENCE	51.4 (37/72)	80.6 (58/72)	**90.3 (65/72)**	**58.3 (42/72)**	**77.8 (56/72)**	91.7 (66/72)
INF+PHYSIM	51.4 (37/72)	**81.9 (59/72)**	**90.3 (65/72)**	**58.3 (42/72)**	**77.8 (56/72)**	91.7 (66/72)

2.5.2 Dataset

We use a database of near-miss accidents published by Tokyo University of Agriculture and Technology. The dataset consists of over 100,000 video recordings of near-miss accidents recorded by a taxi's drive recorders[3]. For evaluation, we use over 3,000 videos recorded in 2014.

The dataset includes traffic scenes that has no potential risks indeed, since it is collecded based on existence of sudden braking. We conducted a corpus study on the dataset to classify scenes that are needed to predict risk truly based on cause of braking. The classification is entrusted to a person who is not involved in the development of the dataset. We classified about 1,000 scenes that the risky object is visible from the point of view of the ego-vehicle in two seconds before the actual near-miss. Table 2.1 shows the classification result. The scenes labeled Avoidance and Prediction are considered to be potentially risky, so we further classified them.

Next classification is based on whether the ego-vehicle changed the trajectory or not from two seconds before the actual near-miss to it. We collect the scenes that ego-vehicle did not change the trajectory in order to make task setting “What potential risks are there when the ego-vehicle keeps the speed?” Table 2.2 shows the classification result. Direct label denotes tha scene that ego-vehicle did not change the trajectory, and Change label denotes the scene that it did. Finally, we use Direct labeled 379 scenes as evaluation data, divided the dataset into a training set, test set, validation set with the ratio of 3:1:1. The data corresponding above labels and ID on the database is publicly available[4].

To create a benchmark dataset, we manually created a 2-dimensional bird-view map for each video.

2.5.3 Models

In the evaluation, we compare the following three models.

[3]http://web.tuat.ac.jp/~smrc/drcenter.html (in Japanese)
[4]`https://github.com/reiyw/traffic-scene-understanding`

- BASELINE: predicts a risk by an abductive reasoner with the cost function $Cost(H) = \mathbf{w_c} \cdot \Phi_c(H, O)$ (referred to as BASELINE). This baseline directly models a mapping between observed information and a risky entity-action tuple. As mentioned in Sec. 2.4.1, the weight vector $\mathbf{w_c}$ is trained by Ranking SVM Joachims (2002); therefore, the result of this baseline indicates the basic performance of a statistical machine learning-based ranker.

- INFERENCE: predicts a risk by an abductive reasoner with the full cost function desribed in Sec. 2.4.1. It does not perform physics simulation: the model uses only symbolic information for the risk prediction.

- INF+PHYSIM: the proposed model, which predicts a risk by an abductive reasoner combined with the physics simulator desribed in Sec. 2.4.2.

2.5.4 Evaluation Results

The results are shown in Table 2.3. By comparing BASELINE and INFERENCE, we observe that adding abductive reasoning does not hurt the performance: the output is successfully enriched so that physics simulation can be performed. Furthermore, we observed the performance improvement on the test set. Manual inspection reveals that some mistakes made by the baseline model were corrected by an inference-based prediction.

We show the example improvement in Figure 2.4. The risk is that *Car* will change the lane to avoid *ParkingCar*, which might lead to a collision between the ego-vehicle (or *Me*) and *Car*. While BASELINE predicted that *Car* will stop, INFERENCE predicted that *Car* will change the lane *with the richer information* that the target lane is the lane next to the current lane.

Using the predicted rich information, the physics simulator predicted the future trajectories of each traffic agent. As illustrated in Figure 2.4, the physics simulator correctly predicted the future trajectories and also infer that *Me* will collide with *Car* 3.6 seconds after. This indicates that our logical inference framework successfully connects the world of symbolic inference to the physical world. For a simple machine learning-based ranker or classifier, it is relatively harder to predict this kind of richer explanations.

Finally, we compare INFERENCE with INF+PHYSIM. The results indicate that physics simulation did not improve the results of qualitative inference. In future work, we will conduct a deeper analysis on the results of physics simulation and refine the entire framework to improve the results.

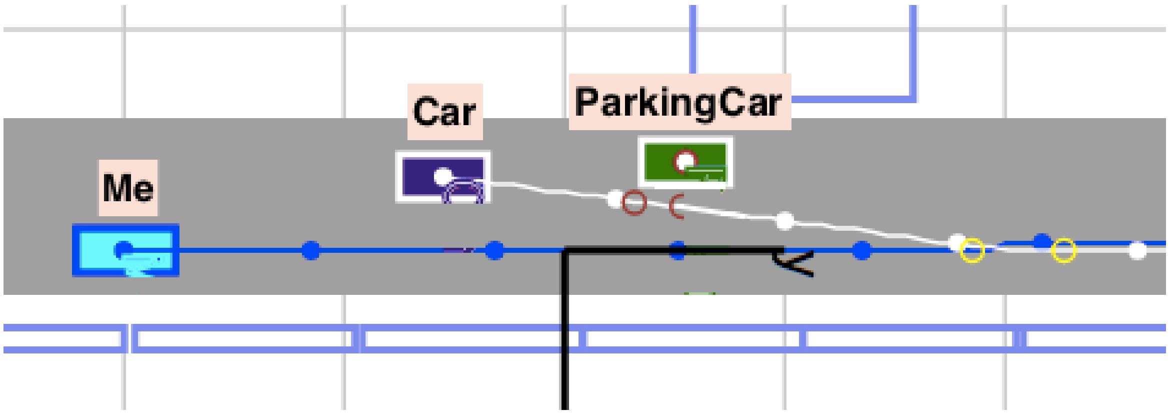


Fig. 2.4 Example trajectories our physics simulator output. Scene ID is 1397 on the database. The boxes represent vehicles, and the lines that drawn from vehicles represent trajectories.

2.6 Conclusions

We have developd an Advanced Driver Assistance System (ADAS) that can recognize potential risks in traffic scenes and provide the reasoning for its prediction. We have extended our previous qualitative risk prediction model with physics simulation to overcome the weakness of qualitative inference. Our evaluation on a real-life traffic incident database demonstrates the potentiality of our approach.

In future work, we will refine the task setting for more practical evaluation. Currently, the task setting requires us to predict a risk *exactly two seconds after* the input scene; however, in practice, predicting *any* risks after the input scene will be beneficial. Another future work will include evaluating the quality of produced explanations.

ACKNOWLEDGMENT

This work was supported by JSPS KAKENHI Grant Number 15H01702. The authors would like to thank DENSO CORPORATION for funding this research.

Chapter 3

Interpretable and Compositional Relation Learning by Joint Training with an Autoencoder

Embedding models for entities and relations are extremely useful for recovering missing facts in a knowledge base. Intuitively, a relation can be modeled by a matrix mapping entity vectors. However, relations reside on low dimension sub-manifolds in the parameter space of arbitrary matrices – for one reason, composition of two relations M_1, M_2 may match a third M_3 (e.g. composition of relations `currency_of_country` and `country_of_film` usually matches `currency_of_film_budget`), which imposes compositional constraints to be satisfied by the parameters (i.e. $M_1 \cdot M_2 \approx M_3$). In this chapter we investigate a dimension reduction technique by training relations jointly with an autoencoder, which is expected to better capture compositional constraints. We achieve state-of-the-art on Knowledge Base Completion tasks with strongly improved Mean Rank, and show that joint training with an autoencoder leads to interpretable sparse codings of relations, helps discovering compositional constraints and benefits from compositional training. Our source code is released at github.com/tianran/glimvec.

3.1 Introduction

Broad-coverage knowledge bases (KBs) such as Freebase (Bollacker et al., 2008) and DBPedia (Auer et al., 2007) store a large amount of facts in the form of ⟨head entity, relation, tail entity⟩ triples (e.g. ⟨*The Matrix*, `country_of_film`, *Australia*⟩), which could support

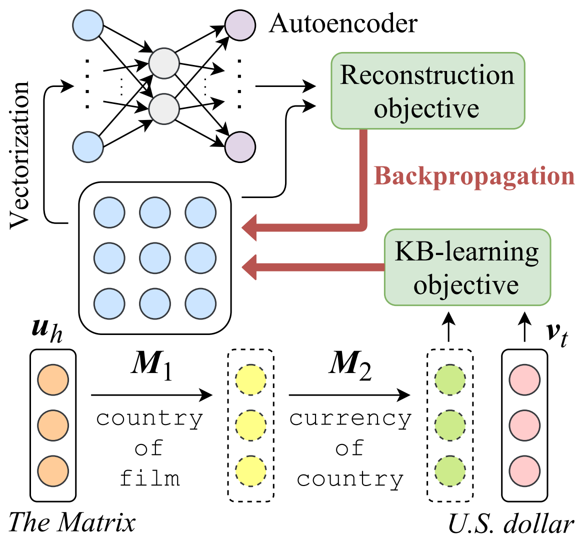


Fig. 3.1 In joint training, relation parameters (e.g. M_1) receive updates from both a *KB-learning objective*, trying to predict entities in the KB; and a *reconstruction objective* from an autoencoder, trying to recover relations from low dimension codings.

a wide range of reasoning and question answering applications. The Knowledge Base Completion (KBC) task aims to predict the missing part of an incomplete triple, such as ⟨*Finding Nemo*, `country_of_film`, ?⟩, by reasoning from known facts stored in the KB.

As a most common approach (Wang et al., 2017), modeling entities and relations to operate in a low dimension vector space helps KBC, for three conceivable reasons. First, when dimension is low, entities modeled as vectors are forced to share parameters, so "similar" entities which participate in many relations in common get close to each other (e.g. *Australia* close to *US*). This could imply that an entity (e.g. *US*) "type matches" a relation such as `country_of_film`. Second, relations may share parameters as well, which could transfer facts from one relation to other similar relations, for example from ⟨x, `award_winner`, y⟩ to ⟨x, `award_nominated`, y⟩. Third, spatial positions might be used to implement *composition* of relations, as relations can be regarded as mappings from head to tail entities, and the composition of two maps can match a third (e.g. the composition of `currency_of_country` and `country_of_film` matches the relation `currency_of_film_budget`), which could be captured by modeling composition in a space.

However, modeling relations as mappings naturally requires more parameters – a general linear map between d-dimension vectors is represented by a matrix of d^2 parameters – which are less likely to be shared, impeding transfers of facts between similar relations. Thus, it is desired to reduce dimensionality of relations; furthermore, the existence of a composition of two relations (assumed to be modeled by matrices M_1, M_2) matching a third (M_3) also justifies dimension reduction, because it implies a *compositional constraint* $\boldsymbol{M}_1 \cdot \boldsymbol{M}_2 \approx \boldsymbol{M}_3$ that can be satisfied only by a lower dimension sub-manifold in the parameter space[1].

Previous approaches reduce dimensionality of relations by imposing pre-designed hard constraints on the parameter space, such as constraining that relations are translations (Bordes et al., 2013a) or diagonal matrices (Yang et al., 2015), or assuming they are linear combinations of a small number of prototypes (Xie et al., 2017). However, pre-designed hard constraints do not seem to cope well with compositional constraints, because it is difficult to know *a priori* which two relations compose to which third relation, hence difficult to choose a pre-design; and compositional constraints are not always exact (e.g. the composition of `currency_of_country` and `headquarter_location` usually matches `business_operation_currency` but not always), so hard constraints are less suited.

In this chapter, we investigate an alternative approach by training relation parameters jointly with an autoencoder (Figure 3.1). During training, the autoencoder tries to reconstruct relations from low dimension codings, with the reconstruction objective back-propagating to relation parameters as well. We show this novel technique promotes parameter sharing between different relations, and drives them toward low dimension manifolds (Sec.3.6.2). Besides, we expect the technique to cope better with compositional constraints, because it discovers low dimension manifolds posteriorly from data, and it does not impose any explicit hard constraints.

Yet, joint training with an autoencoder is not simple; one has to keep a subtle balance between gradients of the reconstruction and KB-learning objectives throughout the training process. We are not aware of any theoretical principles directly addressing this problem; but we found some important settings after extensive pre-experiments (Sec.3.4). We evaluate our system using standard KBC datasets, achieving state-of-the-art on several of them (Sec.3.6.1), with strongly improved Mean Rank. We discuss detailed settings that lead to the performance (Sec.3.4.1), and we show that joint training with an autoencoder indeed helps discovering compositional constraints (Sec.3.6.2) and benefits from compositional training (Sec.3.6.3).

3.2 Base Model

A knowledge base (KB) is a set $\mathscr{T}$ of triples of the form $\langle h, r, t\rangle$, where $h, t \in \mathscr{E}$ are entities and $r \in \mathscr{R}$ is a relation (e.g. $\langle$*The Matrix*, `country_of_film`, *Australia*$\rangle$). A relation r has its inverse $r^{-1} \in \mathscr{R}$ so that for every $\langle h, r, t\rangle \in \mathscr{T}$, we regard $\langle t, r^{-1}, h\rangle$ as also in the

[1]It is noteworthy that similar compositional constraints apply to most modeling schemes of relations, not just matrices.

KB. Under this assumption and given $\mathcal{T}$ as training data, we consider the Knowledge Base Completion (KBC) task that predicts candidates for a missing tail entity in an incomplete $\langle h, r, ?\rangle$ triple.

Most approaches tackle this problem by training a *score function* measuring the plausibility of triples being facts. The model we implement in this work represents entities h, t as d-dimension vectors $\boldsymbol{u}_h, \boldsymbol{v}_t$ respectively, and relation r as a $d \times d$ matrix $\boldsymbol{M}_r$. If $\boldsymbol{u}_h, \boldsymbol{v}_t$ are one-hot vectors with dimension $d = |\mathcal{E}|$ corresponding to each entity, one can take $\boldsymbol{M}_r$ as the adjacency matrix of entities joined by relation r, so the set of tail entities filling into $\langle h, r, ?\rangle$ is calculated by $\boldsymbol{u}_h^\top \boldsymbol{M}_r$ (with each nonzero entry corresponds to an answer). Thus, we have $\boldsymbol{u}_h^\top \boldsymbol{M}_r \boldsymbol{v}_t > 0$ if and only if $\langle h, r, t\rangle \in \mathcal{T}$. This motivates us to use $\boldsymbol{u}_h^\top \boldsymbol{M}_r \boldsymbol{v}_t$ as a natural parameter to model plausibility of $\langle h, r, t\rangle$, even in a low dimension space with $d \ll |\mathcal{E}|$. Thus, we define the score function as

$$s(h, r, t) := \exp(\boldsymbol{u}_h^\top \boldsymbol{M}_r \boldsymbol{v}_t) \tag{3.1}$$

for the basic model. This is similar to the bilinear model of Nickel et al. (2011), except that we distinguish $\boldsymbol{u}_h$ (the vector for head entities) from $\boldsymbol{v}_t$ (the vector for tail entities). It has also been proposed in Tian et al. (2016), but for modeling dependency trees rather than KBs.

More generally, we consider *composition* of relations $r_1/\ldots/r_l$ to model *path*s in a KB (Guu et al., 2015a), as defined by $r_1, \ldots, r_l$ participating in a sequence of facts such that the head entity of each fact coincides with the tail of its previous. For example, a sequence of two facts $\langle$*The Matrix*, `country_of_film`, *Australia*$\rangle$ and $\langle$*Australia*, `currency_of_country`, *Australian Dollar*$\rangle$ form a path of composition `country_of_film`/`currency_of_country`, because the head of the second fact (i.e. *Australia*) coincides with the tail of the first. Using the previous $d = |\mathcal{E}|$ analogue, one can verify that composition of relations is represented by multiplication of adjacency matrices, so we accordingly define

$$s(h, r_1/\ldots/r_l, t) := \exp(\boldsymbol{u}_h^\top \boldsymbol{M}_{r_1} \cdots \boldsymbol{M}_{r_l} \boldsymbol{v}_t)$$

to measure the plausibility of a path. It is explored in Guu et al. (2015a) to learn a score function not only for single facts but also for paths. This *compositional training* scheme is shown to bring valuable information about the structure of the KB and may help KBC. In this work, we conduct experiments both with and without compositional training.

In order to learn parameters $\boldsymbol{u}_h, \boldsymbol{v}_t, \boldsymbol{M}_r$ of the score function, we follow Tian et al. (2016) using a Noise Contrastive Estimation (NCE) (Gutmann and Hyvärinen, 2012) objective. For each path (or triple) $\langle h, r_1/\ldots, t\rangle$ taken from the KB, we generate negative samples by replacing the tail entity t with some random noise t^*. Then, we maximize

$$\mathscr{L}_1 := \sum_{\text{path}} \ln \frac{s(h, r_1/\ldots, t)}{k + s(h, r_1/\ldots, t)} + \sum_{\text{noise}} \ln \frac{k}{k + s(h, r_1/\ldots, t^*)}$$

as our *KB-learning objective*. Here, k is the number of noises generated for each path. When the score function is regarded as probability, $\mathscr{L}_1$ represents the log-likelihood of "$\langle h, r_1/\ldots, t\rangle$ being actual path and $\langle h, r_1/\ldots, t^*\rangle$ being noise". Maximizing $\mathscr{L}_1$ increases the scores of actual paths and decreases the scores of noises.

3.3 Joint Training with an Autoencoder

Autoencoders learn efficient codings of high-dimensional data while trying to reconstruct the original data from the coding. By joint training relation matrices with an autoencoder, we also expect it to help reducing the dimensionality of the original data (i.e. relation matrices).

Formally, we define a *vectorization* $\boldsymbol{m}_r$ for each relation matrix $\boldsymbol{M}_r$, and use it as input to the autoencoder. $\boldsymbol{m}_r$ is defined as a reshape of $\boldsymbol{M}_r$ flattened into a d^2-dimension vector, and normalized such that $\|\boldsymbol{m}_r\| = \sqrt{d}$. We define

$$\boldsymbol{c}_r := \mathrm{ReLU}(\boldsymbol{A}\boldsymbol{m}_r) \tag{3.2}$$

as the coding. Here $\boldsymbol{A}$ is a $c \times d^2$ matrix with $c \ll d^2$, and ReLU is the Rectified Linear Unit function (Nair and Hinton, 2010). We reconstruct the input from $\boldsymbol{c}_r$ by multiplying a $d^2 \times c$ matrix $\boldsymbol{B}$. We want $\boldsymbol{B}\boldsymbol{c}_r$ to be more similar to $\boldsymbol{m}_r$ than other relations. For this purpose, we define a similarity

$$g(r_1, r_2) := \exp\left(\frac{1}{\sqrt{dc}} \boldsymbol{m}_{r_1}^\top \boldsymbol{B}\boldsymbol{c}_{r_2}\right), \tag{3.3}$$

which measures the length of $\boldsymbol{B}\boldsymbol{c}_{r_2}$ projected to the direction of $\boldsymbol{m}_{r_1}$. In order to learn the parameters $\boldsymbol{A}, \boldsymbol{B}$, we adopt the Noise Contrastive Estimation scheme as in Sec.3.2, generate random noises r^* for each relation r and maximize

$$\mathscr{L}_2 := \sum_{r \in \mathscr{R}} \ln \frac{g(r, r)}{k + g(r, r)} + \sum_{r^* \sim \mathscr{R}} \ln \frac{k}{k + g(r, r^*)}$$

as our *reconstruction objective*. Maximizing $\mathscr{L}_2$ increases $\boldsymbol{m}_r$'s similarity with $\boldsymbol{B}\boldsymbol{c}_r$, and decreases it with $\boldsymbol{B}\boldsymbol{c}_{r^*}$.

During joint training, both $\mathscr{L}_1$ and $\mathscr{L}_2$ are simultaneously maximized, and the gradient $\nabla\mathscr{L}_2$ propagates to relation matrices as well. Since $\nabla\mathscr{L}_2$ depends on $\boldsymbol{A}$ and $\boldsymbol{B}$, and $\boldsymbol{A}, \boldsymbol{B}$

interact with all relations, they promote indirect parameter sharing between different relation matrices. In Sec.3.6.2, we further show that joint training drives relations toward a low dimension manifold.

3.4 Optimization Tricks

Joint training with an autoencoder is not simple. Relation matrices receive updates from both $\nabla\mathscr{L}_1$ and $\nabla\mathscr{L}_2$, but if they update $\nabla\mathscr{L}_1$ too much, the autoencoder has no effect; conversely, if they update $\nabla\mathscr{L}_2$ too often, all relation matrices crush into one cluster. Furthermore, an autoencoder should learn from genuine patterns of relation matrices that emerge from fitting the KB, but not the reverse – in which the autoencoder imposes arbitrary patterns to relation matrices according to random initialization. Therefore, it is not surprising that a naive optimization of $\mathscr{L}_1 + \mathscr{L}_2$ does not work.

After extensive pre-experiments, we have found some crucial settings for successful training. The most important "magic" is the scaling factor $\frac{1}{\sqrt{dc}}$ in definition of the similarity function (3.3), perhaps being combined with other settings as we discuss below. We have tried different factors 1, $\frac{1}{\sqrt{d}}$, $\frac{1}{\sqrt{c}}$ and $\frac{1}{dc}$ instead, with various combinations of d and c; but the autoencoder failed to learn meaningful codings in other settings. When the scaling factor is too small (e.g. $\frac{1}{dc}$), all relations get almost the same coding; conversely if the factor is too large (e.g. 1), all codings get very close to 0.

The next important rule is to keep a balance between the updates coming from $\nabla\mathscr{L}_1$ and $\nabla\mathscr{L}_2$. We use Stochastic Gradient Descent (SGD) for optimization, and the common practice (Bottou, 2012) is to set the learning rate as

$$\alpha(\tau) := \frac{\eta}{1 + \eta\lambda\tau}. \tag{3.4}$$

Here, η, λ are hyper-parameters and τ is a counter of processed data points. In this work, in order to control the updates in detail to keep a balance, we modify (3.4) to use a a step counter τ_r for each relation r, counting "number of updates" instead of data points[2]. That is, whenever $\boldsymbol{M}_r$ gets a nonzero update from a gradient calculation, τ_r increases by 1. Furthermore, we use different hyper-parameters for different "types of updates", namely η_1, λ_1 for updates coming from $\nabla\mathscr{L}_1$, and η_2, λ_2 for updates coming from $\nabla\mathscr{L}_2$. Thus, let Δ_1 be the partial gradient of $\nabla\mathscr{L}_1$, and Δ_2 the partial gradient of $\nabla\mathscr{L}_2$, we update $\boldsymbol{M}_r$ by $\alpha_1(\tau_r)\Delta_1 + \alpha_2(\tau_r)\Delta_2$

[2]Similarly, we set separate step counters for all head and tail entities, and the autoencoder as well.

at each step, where

$$\alpha_1(\tau_r) := \frac{\eta_1}{1+\eta_1\lambda_1\tau_r}, \quad \alpha_2(\tau_r) := \frac{\eta_2}{1+\eta_2\lambda_2\tau_r}.$$

The rule for setting η_1, λ_1 and η_2, λ_2 is that, η_2 should be much smaller than η_1, because η_1, η_2 control the magnitude of learning rates at the early stage of training, with the autoencoder still largely random and Δ_2 not making much sense; on the other hand, one has to choose λ_1 and λ_2 such that $\|\Delta_1\|/\lambda_1$ and $\|\Delta_2\|/\lambda_2$ are at the same scale, because the learning rates approach $1/(\lambda_1\tau_r)$ and $1/(\lambda_2\tau_r)$ respectively, as the training proceeds. In this way, the autoencoder will not impose random patterns to relation matrices according to its initialization at the early stage, and a balance is kept between $\alpha_1(\tau_r)\Delta_1$ and $\alpha_2(\tau_r)\Delta_2$ later.

But how to estimate $\|\Delta_1\|$ and $\|\Delta_2\|$? It seems that we can approximately calculate their scales from initialization. In this work, we use i.i.d. Gaussians of variance $1/d$ to initialize parameters, so the initial Euclidean norms are $\|\boldsymbol{u}_h\| \approx 1$, $\|\boldsymbol{v}_t\| \approx 1$, $\|\boldsymbol{M}_r\| \approx \sqrt{d}$, and $\|\boldsymbol{BAm}_r\| \approx \sqrt{dc}$. Thus, by calculating $\nabla\mathscr{L}_1$ and $\nabla\mathscr{L}_2$ using (3.1) and (3.3), we have approximately

$$\|\Delta_1\| \approx \|\boldsymbol{u}_h\boldsymbol{v}_t^\top\| \approx 1, \quad \text{and} \tag{3.5}$$

$$\|\Delta_2\| \approx \|\frac{1}{\sqrt{dc}}\boldsymbol{Bc}_r\| \approx \frac{1}{\sqrt{dc}}\|\boldsymbol{BAm}_r\| \approx 1. \tag{3.6}$$

It suggests that, because of the scaling factor $\frac{1}{\sqrt{dc}}$ in (3.3), we have $\|\Delta_1\|$ and $\|\Delta_2\|$ at the same scale, so we can set $\lambda_1 = \lambda_2$. This might not be a mere coincidence.

3.4.1 Training the Base Model

Besides the tricks for joint training, we also found settings that significantly improve the base model on KBC, as briefly discussed below. In Sec.3.6.3, we will show performance gains by these settings using the FB15k-237 validation set.

Normalization It is better to normalize relation matrices to $\|\boldsymbol{M}_r\| = \sqrt{d}$ during training. This might reduce fluctuations in entity vector updates.

Regularizer It is better to minimize $\|\boldsymbol{M}_r^\top\boldsymbol{M}_r - \frac{1}{d}\operatorname{tr}(\boldsymbol{M}_r^\top\boldsymbol{M}_r)I\|$ during training. This regularizer drives $\boldsymbol{M}_r$ toward an orthogonal matrix (Tian et al., 2016) and might reduce fluctuations in entity vector updates. As a result, all relation matrices trained in this work are very close to orthogonal.

Initialization Instead of pure Gaussian, it is better to initialize matrices as $(I+G)/2$, where G is random. The identity matrix I helps passing information from head to tail (Tian et al., 2016).

Negative Sampling Instead of a unigram distribution, it is better to use a *uniform* distribution for generating noises. This is somehow counter-intuitive compared to training word embeddings.

3.5 Related Works

KBs have a wide range of applications (Berant et al., 2013; Hixon et al., 2015; Nickel et al., 2016a) and KBC has inspired a huge amount of research (Bordes et al., 2013a; Das et al., 2017a; Hayashi and Shimbo, 2017; Nguyen et al., 2016; Riedel et al., 2013; Socher et al., 2013; Toutanova et al., 2016a; Wang et al., 2014a,c; Xiao et al., 2016).

Among the previous works, TransE Bordes et al. (2013a) is the classic method which represents a relation as a translation of the entity vector space, and is partially inspired by Mikolov et al. (2013)'s vector arithmetic method of solving word analogy tasks. Although competitive in KBC, it is speculated that this method is well-suited for 1-to-1 relations but might be too simple to represent N-to-N relations accuratelyWang et al. (2017). Thus, extensions such as TransR Lin et al. (2015c) and STransE Nguyen et al. (2016) are proposed to map entities into a relation-specific vector space before translation. The ITransF model Xie et al. (2017) further enhances this approach by imposing a hard constraint that the relation-specific maps should be linear combinations of a small number of prototypical matrices. Our work inherits the same motivation with ITransF in terms of promoting parameter-sharing among relations.

On the other hand, the base model used in this work originates from RESCAL Nickel et al. (2011), in which relations are naturally represented as analogue to the adjacency matrices (Sec.3.2). Further developments include HolE Nickel et al. (2016b) and ConvE Dettmers et al. (2018) which improve this approach in terms of parameter-efficiency, by introducing low dimension factorizations of the matrices. We inherit the basic model of RESCAL but draw additional training techniques from Tian et al. (2016), and show that the base model already can achieve near state-of-the-art performance (Sec.3.6.1,3.6.3). This sends a message similar to Kadlec et al. (2017), saying that training tricks might be as important as model designs.

Nevertheless, we emphasize the novelty of this work in that the previous models mostly achieve dimension reduction by imposing some pre-designed hard constraints (Bordes et al., 2013a; Dettmers et al., 2018; Nickel et al., 2016b; Trouillon et al., 2016; Xie et al., 2017; Yang et al., 2015), whereas the constraints themselves are not learned from data; in contrast, our approach by jointly training an autoencoder does not impose any explicit hard constraints, so it leads to more flexible modeling.

Moreover, we additionally focus on leveraging composition in KBC. Although this idea has been frequently explored before (Guu et al., 2015a; Lin et al., 2015a; Neelakantan et al., 2015a), our discussion about the concept of compositional constraints and its connection to dimension reduction has not been addressed similarly in previous research. In experiments, we will show (Sec.3.6.2,3.6.3) that joint training with an autoencoder indeed helps finding compositional constraints and benefits from compositional training.

Autoencoders have been used solo for learning distributed representations of syntactic trees (Socher et al., 2011), words and images (Silberer and Lapata, 2014), or semantic roles (Titov and Khoddam, 2015). It is also used for pretraining other deep neural networks (Erhan et al., 2010). However, when combined with other models, the learning of autoencoders, or more generally *sparse codings* (Rubinstein et al., 2010), is usually conveyed in an alternating manner, fixing one part of the model while optimizing the other, such as in Xie et al. (2017). To our knowledge, joint training with an autoencoder is not widely used previously for reducing dimensionality.

Jointly training an autoencoder is not simple because it takes non-stationary inputs. In this work, we modified SGD so that it shares traits with some modern optimization algorithms such as Adagrad (Duchi et al., 2011), in that they both set different learning rates for different parameters. While Adagrad sets them adaptively by keeping track of gradients for all parameters, our modification of SGD is more efficient and allows us to grasp a rough intuition about which parameter gets how much update. We believe our techniques and findings in joint training with an autoencoder could be helpful to reducing dimensionality and improving interpretability in other neural network architectures as well.

3.6 Experiments

We evaluate on standard KBC datasets, including WN18 and FB15k (Bordes et al., 2013a), WN18RR (Dettmers et al., 2018) and FB15k-237 (Toutanova and Chen, 2015). The statistical information of these datasets are shown in Table 3.1.

WN18 collects word relations from WordNet (Miller, 1995), and FB15k is taken from Freebase (Bollacker et al., 2008); both have filtered out low frequency entities. However, it is reported in Toutanova and Chen (2015) that both WN18 and FB15k have information leaks because the inverses of some test triples appear in the training set. FB15k-237 and

Dataset	$\lvert\mathscr{E}\rvert$	$\lvert\mathscr{R}\rvert$	#Train	#Valid	#Test
WN18	40,943	18	141,442	5,000	5,000
FB15k	14,951	1,345	483,142	50,000	59,071
WN18RR	40,943	11	86,835	3,034	3,134
FB15k-237	14,541	237	272,115	17,535	20,466

Table 3.1 Statistical information of the KBC datasets. $\lvert\mathscr{E}\rvert$ and $\lvert\mathscr{R}\rvert$ denote the number of entities and relation types, respectively; #Train, #Valid, and #Test are the numbers of triples in the training, validation, and test sets, respectively.

WN18RR fix this problem by deleting such triples from training and test data. In this work, we do evaluate on WN18 and FB15k, but our models are mainly tuned on FB15k-237.

For all datasets, we set the dimension $d = 256$ and $c = 16$, the SGD hyper-parameters $\eta_1 = 1/64, \eta_2 = 2^{-14}$ and $\lambda_1 = \lambda_2 = 2^{-14}$. The training batch size is 32 and the triples in each batch share the same head entity. We compare the base model (BASE) to our joint training with an autoencoder model (JOINT), and the base model with compositional training (BASE+COMP) to our joint model with compositional training (JOINT+COMP). When compositional training is enabled (BASE+COMP, JOINT+COMP), we use random walk to sample paths of length $1+X$, where X is drawn from a Poisson distribution of mean $\lambda = 1.0$.

For any incomplete triple $\langle h, r, ?\rangle$ in KBC test, we calculate a score $s(h, r, e)$ from (3.1), for every entity $e \in \mathscr{E}$ such that $\langle h, r, e\rangle$ *does not appear in any of the training, validation, or test sets* (Bordes et al., 2013a). Then, the calculated scores together with $s(h, r, t)$ for the gold triple is converted to ranks, and the rank of the gold entity t is used for evaluation. Evaluation metrics include Mean Rank (MR), Mean Reciprocal Rank (MRR), and Hits at 10 (H10). Lower MR, higher MRR, and higher H10 indicate better performance.

We consult MR and MRR on validation sets to determine training epochs; we stop training when both MR and MRR have stopped improving.

3.6.1 KBC Results

The results are shown in Table 3.2. We found that joint training with an autoencoder mostly improves performance, and the improvement becomes more clear when compositional training is enabled (i.e., JOINT $\geq$ BASE and JOINT+COMP $>$ BASE+COMP). This is convincing because generally, joint training contributes with its regularizing effects, and drastic improvements are less expected[3]. When compositional training is enabled, the system usually

[3]The source code and trained models are publicly released at https://github.com/tianran/glimvec, where we also show the mean performance and deviations of multiple random initializations, to give a more complete picture.

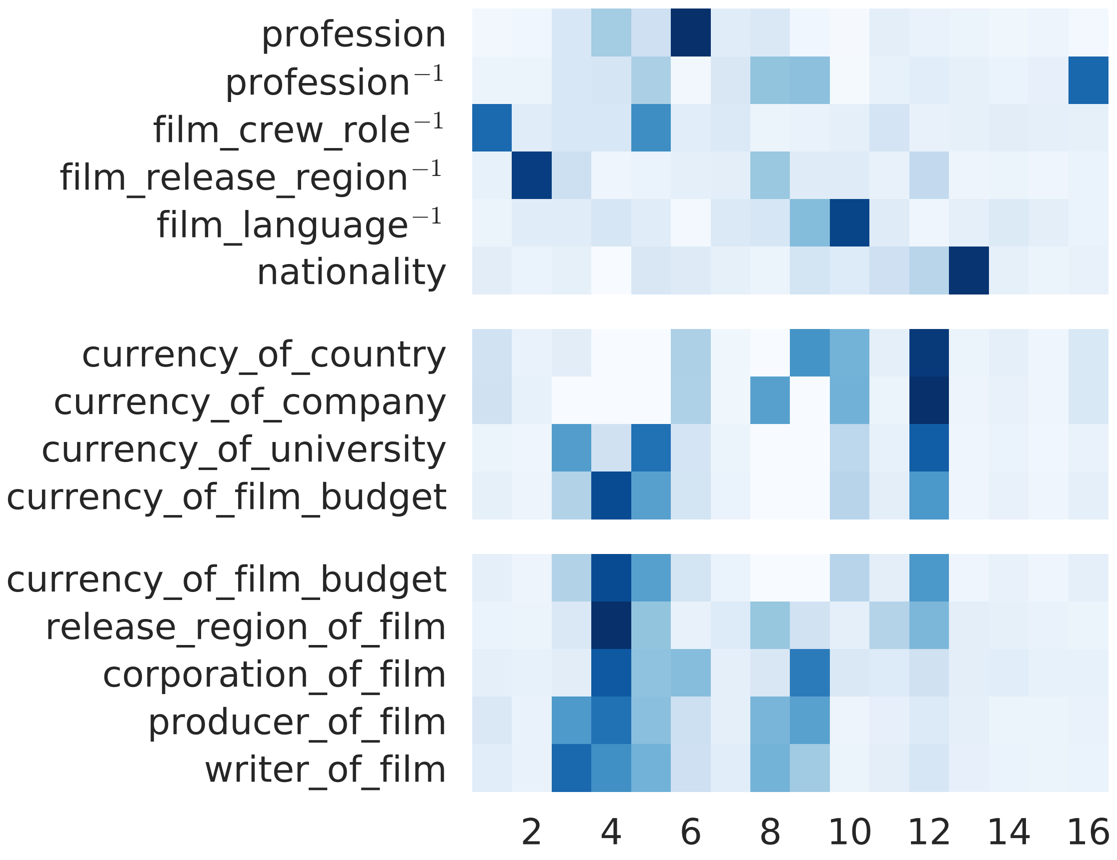


Fig. 3.2 Examples of relation codings learned from FB15k-237. Each row shows a 16 dimension vector encoding a relation. Vectors are normalized such that their entries sum to 1.

achieves better MR, though not always improves in other measures. The performance gains are more obvious on the WN18RR and FB15k-237 datasets, possibly because WN18 and FB15k contain a lot of easy instances that can be solved by a simple rule Dettmers et al. (2018).

Furthermore, the numbers demonstrated by our joint and base models are among the strongest in the literature. We have conducted re-experiments of several representative algorithms, and also compare with state-of-the-art published results. For re-experiments, we use Lin et al. (2015c)'s implementation[4] of TransE (Bordes et al., 2013a) and TransR, which represent relations as vector translations; and Nickel et al. (2016b)'s implementation[5] of RESCAL (Nickel et al., 2011) and HolE, where RESCAL is most similar to the BASE model

[4]https://github.com/thunlp/KB2E
[5]https://github.com/mnick/holographic-embeddings

Model	WN18		FB15k		WN18RR			FB15k-237		
	MR	H10	MR	H10	MR	MRR	H10	MR	MRR	H10
JOINT	**277**	**95.8**	**53**	**82.5**	**4233**	**.461***	**53.4**	**212**	.336	**52.3***
BASE	286	**95.8**	**53**	**82.5**	4371	.459	52.9	215	**.337***	**52.3***
JOINT+COMP	**191***	**94.8**	**53**	**69.7**	**2268***	**.343**	**54.8***	**197***	**.331**	**51.6**
BASE+COMP	195	**94.8**	54	69.4	2447	.310	54.1	203	.328	51.5
TransE (Bordes et al., 2013a)	292	92.0	**66**	70.4	4311	.202	45.6	**278**	.236	41.6
TransR (Lin et al., 2015c)	**281**	93.6	76	**74.4**	**4222**	.210	**47.1**	320	**.282**	**45.9**
RESCAL (Nickel et al., 2011)	911	58.0	163	41.0	9689	.105	20.3	457	.178	31.9
HolE (Nickel et al., 2016b)	724	**94.3**	293	66.8	8096	**.376**	40.0	1172	.169	30.9
STransE (Nguyen et al., 2016)	206	93.4	69	79.9	-	-	-	-	-	-
ITransF (Xie et al., 2017)	**205**	94.2	65	81.0	-	-	-	-	-	-
ComplEx (Trouillon et al., 2016)	-	94.7	-	84.0	**5261**	.44	**51**	339	.247	42.8
Ensemble DistMult (Kadlec et al., 2017)	457	95.0	35.9	90.4	-	-	-	-	-	-
IRN (Shen et al., 2017)	249	95.3	38	**92.7***	-	-	-	-	-	-
ConvE (Dettmers et al., 2018)	504	95.5	64	87.3	5277	**.46**	48	**246**	**.316**	**49.1**
R-GCN+ (Schlichtkrull et al., 2017)	-	**96.4***	-	84.2	-	-	-	-	.249	41.7
ProjE (Shi and Weninger, 2017)	-	-	**34***	88.4	-	-	-	-	-	-

Table 3.2 KBC results on the WN18, FB15k, WN18RR, and FB15k-237 datasets. The first and second sectors compare our joint to the base models with and without compositional training, respectively; the third sector shows our re-experiments and the fourth shows previous published results. Bold numbers are the best in each sector, and (*) indicates the best of all.

and HolE is a more parameter-efficient variant. We experimented with the default settings, and found that our models outperform most of them.

Among the published results, STransE (Nguyen et al., 2016) and ITransF (Xie et al., 2017) are more complicated versions of TransR, achieving the previous highest MR on WN18 but are outperformed by our JOINT+COMP model. ITransF is most similar to our JOINT model in that they both learn sparse codings for relations. On WN18RR and FB15k-237, Dettmers et al. (2018)'s report of ComplEx (Trouillon et al., 2016) and ConvE were previously the best results. Our models mostly outperform them. Other results include Kadlec et al. (2017)'s simple but strong baseline and several recent models (Schlichtkrull et al., 2017; Shen et al., 2017; Shi and Weninger, 2017) which achieve best results on FB15k or WN18 in some measure. Our models have comparable results.

3.6.2 Intuition and Insight

What does the autoencoder look like? How does joint training affect relation matrices? We address these questions by analyses showing that **(i)** the autoencoder learns sparse and interpretable codings of relations, **(ii)** the joint training drives relation matrices toward a low dimension manifold, and **(iii)** it helps discovering compositional constraints.

Sparse Coding and Interpretability

Due to the ReLU function in (3.2), our autoencoder learns sparse coding, with most relations having large code values at only two or three dimensions. This sparsity makes it easy to find patterns in the model that to some extent explain the semantics of relations. Figure 3.2 shows some examples.

In the first group of Figure 3.2, we show a small number of relations that are almost always assigned a near one-hot coding, regardless of initialization. These are high frequency relations joining two large categories (e.g. film and language), which probably constitute the skeleton of a KB.

In the second group, we found the 12th dimension strongly correlates with `currency`; and in the third group, we found the 4th dimension strongly correlates with `film`. As for the relation `currency_of_film_budget`, it has large code values at both dimensions. This kind of relation clustering also seems independent of initialization. Intuitively, it shows that the autoencoder may discover similarities between relations and promote indirect parameter sharing among them. Yet, as the autoencoder only reconstructs *approximations* of relation matrices but never constrain them to be exactly equal to the original, relation matrices with very similar codings may still differ considerably. For example, `producer_of_film` and `writer_of_film` have codings of cosine similarity 0.973, but their relation matrices only have[6] a cosine similarity 0.338.

Low dimension manifold

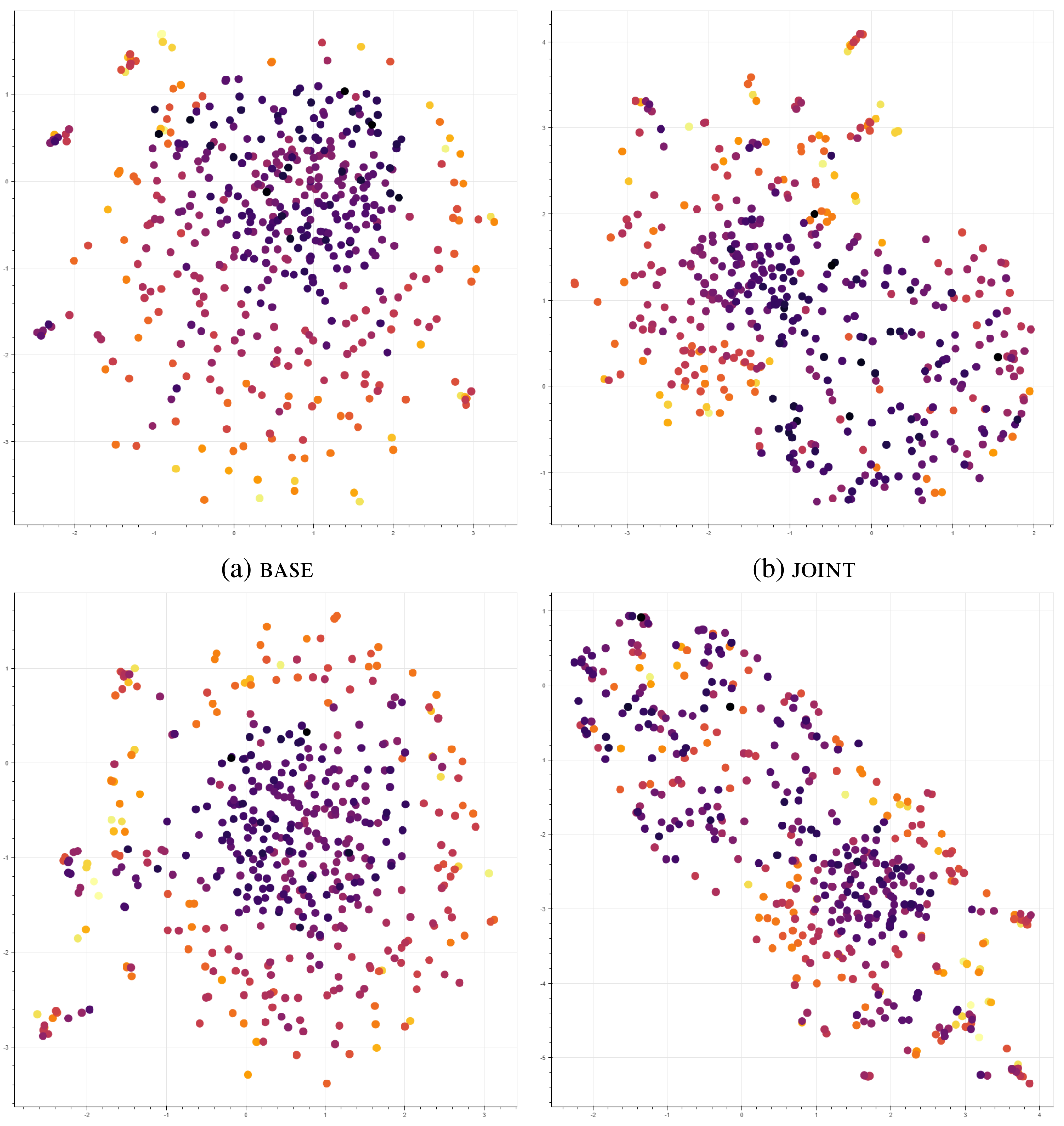
In order to visualize the relation matrices learned by our joint and base models, we use UMAP[7] (McInnes and Healy, 2018) to embed $\boldsymbol{M}_r$ into a 2D plane[8]. We use relation matrices trained on FB15k-237, and compare models trained by the same number of epochs. The results are shown in Figure 3.3.

We can see that Figure 3.3a and Figure 3.3c are mostly similar, with high frequency relations scattered randomly around a low frequency cluster, suggesting that they come from various directions of a high dimension space, with frequent relations probably being pulled further by the training updates. On the other hand, in Figure 3.3b and Figure 3.3d we found

[6]Cosine similarity 0.338 is still high for matrices, due to the high dimensionality of their parameter space.

[7]https://github.com/lmcinnes/umap

[8]UMAP is a recently proposed manifold learning algorithm based on the fuzzy topological structure. We also tried t-SNE (van der Maaten and Hinton, 2008) but found UMAP more insightful.



(a) BASE (b) JOINT

(c) BASE+COMP (d) JOINT+COMP

Fig. 3.3 By UMAP, relation matrices are embedded into a 2D plane. Colors show frequencies of relations; and lighter color means more frequent.

less frequent relations being clustered with frequent ones, and multiple traces of low dimension structures. It suggests that joint training with an autoencoder indeed drives relations toward a low dimension manifold. In addition, Figure 3.3d shows different structures against Figure 3.3b, which we conjecture could be related to compositional constraints discovered by compositional training.

Model	MR	MRR
JOINT+COMP	**130±27**	**.0481±.0090**
BASE+COMP	150±3	.0280±.0010
RANDOMM2	181±19	.0356±.0100

Table 3.3 Performance at discovering compositional constraints extracted from FB15k-237

Compositional constraints

In order to directly evaluate a model's ability to find compositional constraints, we extracted from FB15k-237 a list of $(r_1/r_2, r_3)$ pairs such that r_1/r_2 matches r_3. Formally, the list is constructed as below. For any relation r, we define a *content set* $C(r)$ as the set of (h, t) pairs such that $\langle h, r, t\rangle$ is a fact in the KB. Similarly, we define $C(r_1/r_2)$ as the set of (h, t) pairs such that $\langle h, r_1/r_2, t\rangle$ is a path. We regard $(r_1/r_2, r_3)$ as a compositional constraint if their content sets are similar; that is, if $|C(r_1/r_2) \cap C(r_3)| \geq 50$ and the Jaccard similarity between $C(r_1/r_2)$ and $C(r_3)$ is ≥ 0.4. Then, after filtering out degenerated cases such as $r_1 = r_3$ or $r_2 = r_1^{-1}$, we obtained a list of 154 compositional constraints, e.g. (`currency_of_country/country_of_film`, `currency_of_film_budget`).

For each compositional constraint $(r_1/r_2, r_3)$ in the list, we take the matrices $\boldsymbol{M}_1$, $\boldsymbol{M}_2$ and $\boldsymbol{M}_3$ corresponding to r_1, r_2 and r_3 respectively, and rank $\boldsymbol{M}_3$ according to its cosine similarity with $\boldsymbol{M}_1\boldsymbol{M}_2$, among all relation matrices. Then, we calculate MR and MRR for evaluation. We compare the JOINT+COMP model to BASE+COMP, as well as a randomized baseline where M_2 is selected randomly from the relation matrices in JOINT+COMP instead (RANDOMM2). The results are shown in Table 3.3. We have evaluated 5 different random initializations for each model, trained by the same number of epochs, and we report the mean and standard deviation. We verify that JOINT+COMP performs better than BASE+COMP, indicating that joint training with an autoencoder indeed helps discovering compositional constraints. Furthermore, the random baseline RANDOMM2 tests a hypothesis that joint training might be just clustering M_3 and M_1 here, to the extent that M_3 and M_1 are so close that even a random M_2 can give the correct answer; but as it turns out, JOINT+COMP largely outperforms RANDOMM2, excluding this possibility. Thus, joint training performs better not simply because it clusters relation matrices; it learns compositions indeed.

3.6.3 Losses and Gains

In the KBC task, where are the losses and what are the gains of different settings? With additional evaluations, we show **(i)** some crucial settings for the base model, and **(ii)** joint training with an autoencoder benefits more from compositional training.

Settings	MR	MRR	H10
BASE	**214**	**.338**	**52.5**
no normalization	309	.326	49.9
no regularizer	400	.328	51.3
pure Gaussian	221	.336	52.1
unigram distribution	215	.324	50.6

Table 3.4 Ablation of the four settings of the base model as described in Sec.3.4.1

Crucial settings for the base model

It is noteworthy that our base model already achieves strong results. This is due to several detailed but crucial settings as we discussed in Sec.3.4.1; Table 3.4 shows their gains on the FB15k-237 validation data. The most dramatic improvement comes from the regularizer that drives matrices to orthogonal.

Gains with compositional training

One can force a model to focus more on (longer) compositions of relations, by sampling longer paths in compositional training. Since joint training with an autoencoder helps discovering compositional constraints, we expect it to be more helpful when the sampled paths are longer. In this work, path lengths are sampled from a Poisson distribution, we thus vary the mean λ of the Poisson to control the strength of compositional training. The results on FB15k-237 are shown in Table 3.5.

We can see that, as λ gets larger, MR improves much but MRR slightly drops. It suggests that in FB15k-237, composition of relations might mainly help finding more appropriate candidates for a missing entity, rather than pinpointing a correct one. Yet, joint training improves base models even more as the paths get longer, especially in MR. It further supports our conjecture that joint training with an autoencoder may strongly interact with compositional training.

3.7 Conclusion

We have investigated a dimension reduction technique which trains a KB embedding model jointly with an autoencoder. We have developed new training techniques and achieved state-of-the-art results on several KBC tasks with strong improvements in Mean Rank. Furthermore, we have shown that the autoencoder learns low dimension sparse codings that can be

Model	λ	Valid			Test		
		MR	MRR	H10	MR	MRR	H10
BASE	0	209	.341	52.9	215	.337	52.3
JOINT	0	+1	-.001	-.2	**-3**	-.001	0
BASE	0.5	204	.337	52.2	211	.332	51.7
JOINT	0.5	**-3**	**+.002**	**+.1**	+1	**+.002**	**+.2**
BASE	1.0	191	.334	52.0	203	.328	51.5
JOINT	1.0	**-5**	**+.002**	-.1	**-6**	**+.003**	**+.1**

Table 3.5 Evaluation of BASE and gains by JOINT, on FB15k-237 with different strengths of compositional training. Bold numbers are improvements.

easily explained; the joint training technique drives high-dimensional data toward low dimension manifolds; and the reduction of dimensionality may interact strongly with composition, help discovering compositional constraints and benefit from compositional training. We believe these findings provide insightful understandings of KB embedding models and might be applied to other neural networks beyond the KBC task.

Acknowledgments

This work was supported by JST CREST Grant Number JPMJCR1301, Japan. We thank Pontus Stenetorp, Makoto Miwa, and the anonymous reviewers for many helpful advices and comments.

Chapter 4

Universal Graph Embedding: An Empirical Analysis

In this chapter, we consider an approach, called Universal Graph Embedding (UGE), that embeds the whole graph (Universal Graph) into a vector space by integrating the relational knowledge in the knowledge base and the relational knowledge extracted from the text. UGE can effectively learn knowledge base embedding from a large amount of raw text, and has the potential to speed up compositional reasoning in knowledge bases. However, Toutanova et al. (2015) is the only previous work that has addressed UGE, and the developmental potential of UGE has yet to be discussed. In this chapter, we address three main issues of UGE with the aim of gaining insight into UGE: (1) learning strategies, (2) extension to relational paths, and (3) integration with pre-trained language models.

4.1 Introduction

Knowledge graph completion (or knowledge base completion) is a task that aims to predict unknown facts from known facts in a knowledge graph, and has been applied to the question answering (Qiu et al., 2019; Yang et al., 2019) and knowledge acquisition systems in the biomedical domain (Dai et al., 2019a,b). In typical knowledge bases such as Wikidata (Vrandečiundefined and Krötzsch, 2014), Freebase (Bollacker et al., 2008), and UMLS (Lindberg et al., 1993), real-world facts are stored in the form of triples of head entities, relations, and tail entities. In this context, knowledge graph completion is represented as a problem of predicting missing entities (tail prediction problem).

Most existing research models inference in a knowledge graph as an operation on a low-dimensional vector space, where embeddings of entities and relations are learned from a set of triples (Wang et al., 2017). Since the tail prediction problem requires the selection of tail candidates from the entire entity set, the time efficiency during inference is also an important metric. The approach of embedding the entire knowledge graph in a vector space is advantageous in terms of efficiency, as it does not require the inference of the original knowledge graph during inference, and the candidates can be narrowed down by using only operations in the vector space. However, previous studies have shown that the performance on artificial benchmark datasets, where the locally dense part of the knowledge graph is extracted, is high, while the performance on sparse situations, such as those found in real-world knowledge graphs, is limited (Pujara et al., 2017).

Considering the background, there are ongoing researches to incorporate the complementary information during the learning of knowledge graph embeddings for the more realistic situations. The one is to consider multiple sequences of relations (relation paths) to enable compositional reasoning (Guu et al., 2015b; Takahashi et al., 2018). For example, to infer ⟨*Tensorflow*, *used_for*, *Machine Learning*⟩ from ⟨*Tensorflow*, *developer*, *Google Brain*⟩ and ⟨*Google Brain*, *field_of_work*, *Machine Learning*⟩, the system learns to make the embedding of the relational path *developer/field_of_work* closer to the embedding of *used_for*. The other is integration with textual information Toutanova et al. (2015). For example, if we can capture the fact that the relation *developer* is likely to appear as "*A is developed by B.*" in the text, we can infer ⟨*Chainer*, *used_for*, *Machine Learning*⟩ by combining it with the above triplet using the text "*Chainer is developed by PFN.*" as a clue, even if ⟨*Chainer*, *developer*, *PFN*⟩ is not stored in the knowledge graph. Although these two directions have been studied independently in previous research, they are complementary, and by combining the relational knowledge in the knowledge graph with the relational knowledge extracted from the text, there is a possibility of further improving the inference.

In this study, we attempt to incorporate both the relation paths and textual information in the learning of knowledge graph embeddings. Specifically, we construct a graph that integrates the relational knowledge in the knowledge graph and the relational knowledge extracted from the text (hereafter referred to as the Universal Graph (UG); the exact definition of UG is given later in Section 4.3) as shown in Figure 1, and consider an approach called Universal Graph Embedding (UGE), which embeds UG into a vector space. To the best of our knowledge, Toutanova et al. (2015) is the only previous work on UGE. Toutanova et al. (2015) showed that the performance of knowledge graph completion can be improved by simultaneously training the model of knowledge graph completion and the encoder of the text on a UGE constructed from a large corpus of text. However, the potential of UGE, such as integration with pre-trained language models and consideration of multi-hop paths, has not yet been explored.

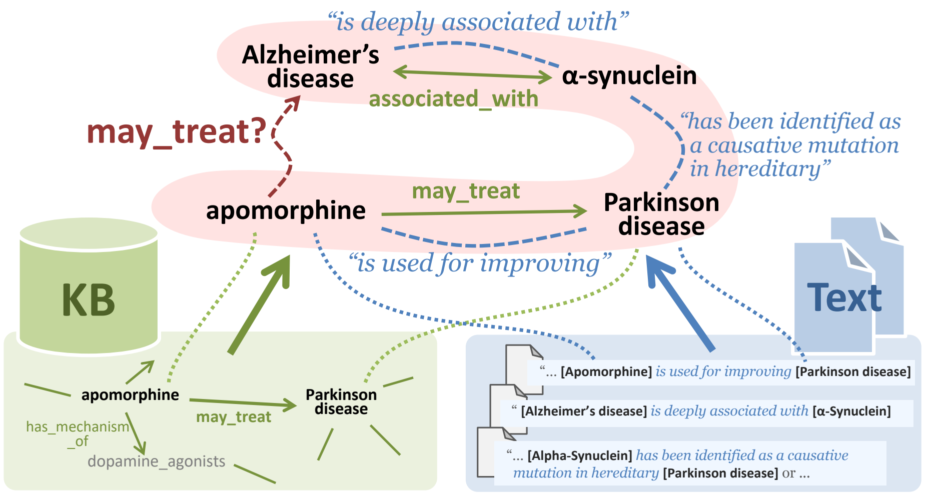

"is deeply associated with"
Alzheimer's disease
α-synuclein
associated_with
may_treat?
"has been identified as a causative mutation in hereditary"
apomorphine
may_treat
Parkinson disease
"is used for improving"
KB
Text
apomorphine
has_mechanism_of
dopamine_agonists
may_treat
Parkinson disease
"... [Apomorphine] is used for improving [Parkinson disease]
" [Alzheimer's disease] is deeply associated with [α-Synuclein]
"... [Alpha-Synuclein] has been identified as a causative mutation in hereditary [Parkinson disease] or ...

Fig. 4.1 Example of relational reasoning on Universal Graph.

In this chapter, we start with a simple model and provide quantitative evaluation results from different angles, in order to gain insights into the UGE. We conducted an exhaustive quantitative evaluation on three main topics: (1) learning the order of the relations in the knowledge graph and the relations extracted from the text, (2) whether multi-hop paths are sampled, and (3) whether the language model is pre-trained. The experimental results reconfirm the usefulness of integrating textual information in knowledge graph completion, but suggest that the noise in the textual information and relation paths must be addressed to further improve the performance by integrating with pre-trained language models and considering multi-hop paths.

4.2 Knowledge Graph Completion

In this study, we consider a set of triples $\mathscr{T}$ (e.g., $\langle$*Chainer*, *developer*, *PFN*$\rangle$) consisting of entities h, r and relations r as a knowledge graph. A knowledge graph can be regarded as a directed multigraph with entities as nodes and relations as edges. knowledge graph completion is a task that completes the triples that do not appear in the training data when the knowledge graph lacking some triples is given as training data.

Most of the prior work addressing this task trains a score function that measures the facticity of the triples. For example, one of the most widely known models, TransE (Bordes et al., 2013b), represents two entities h, t (hereafter referred to as the head and tail entities, respectively) and the relation r by a d-dimensional vector $\boldsymbol{h}, \boldsymbol{t}, \boldsymbol{r} \in \mathbb{R}^d$, respectively, and learns the embedding to maximize the following score function:

$$f(h, r, t; \Theta) := -\|\boldsymbol{h} + \boldsymbol{r} - \boldsymbol{t}\|\,. \tag{4.1}$$

4.2.1 Relation Paths in Knowledge Graph Embedding

More generally, learning for knowledge base embedding can consider the path $r_1/\dots/r_l$ [1] of the relation $r_1 \dots r_l$ (Guu et al., 2015b; Takahashi et al., 2018). Starting from an entity (node) h, let t be the entity that is reached after l transitions in the knowledge graph. The score function of TransE can be extended as in

$$f(h, r_1/\dots/r_l, t; \Theta) := -\left\|\boldsymbol{h} + \boldsymbol{r_1} + \cdots + \boldsymbol{r_l} - \boldsymbol{t}\right\|\,. \tag{4.2}$$

It is known that the constraints between relations inherent in the knowledge graph can be automatically obtained from the data by optimizing the score function for paths sampled from the knowledge graph (Takahashi et al., 2018). For example, the relation *used_for* in the knowledge graph is statistically likely to share a head entity and a tail entity with the relation that is a composite of *developer* and *field_of_work*. In this case, the relation embedding $\boldsymbol{r}_{used\_for}$ of *used_for* acts as a force that brings it closer to the vector $\boldsymbol{r}_{developer} + \boldsymbol{r}_{field\_of\_work}$, which is a composite of *developer* and *field_of_work* ($\boldsymbol{r}_{developer} + \boldsymbol{r}_{field\_of\_work} \approx \boldsymbol{r}_{used\_for}$). At the same time, this corresponds to applying a regularization on the spatial arrangement of the entity embeddings (Guu et al., 2015b).

4.2.2 Joint Representation Learning of Text and Knowledge Graph

Methods for utilizing information from text during learning for knowledge base embedding have been actively studied (Riedel et al., 2013; Toutanova et al., 2015; Wang et al., 2014b). The common idea in these studies is to compensate for the missing information in a clean but sparse knowledge base with a large but noisy set of text.

The most important issue in integrating textual information during training for knowledge base embedding is to solve what kind of relationship does a mention pair appear in a text. The textual relation is a textual representation that describes the relationship between mentions. For example, given the sentence *"Chainer is developed by PFN,"* the relationship between *Chainer* and *PFN* is considered to be specified by the textual relation *"is developed by"*. Mention pairs appearing in the text do not necessarily represent a specific relation. This

[1] In this chapter, a relation path of length one consisting of a single relation is called a single-hop path, and a relation path of length two or more is called a multi-hop path.

is known as a noise problem in dinstantly supervised relation extraction. There has been a lot of discussion on the specific representation of textual relations, but mainly in the field of relation extraction, simply word sequences between the mentions or engaged paths between the mentions have been widely used (Pawar et al., 2017).

To the best of our knowledge, Toutanova et al. (2015) is the only work that corresponds to the UGE approach, although there are many previous studies that utilize textual information when training knowledge graph embedding. The model of Toutanova et al. (2015) is based on the addition of a textual triple to the training data of knowledge base embedding, which is composed of a textual relation and a mention pair extracted from a text corpus. They also found that there is a common substructure in textual relations such that they share mention pairs[2]. They showed that by sharing the embedding of mentions with the embedding of entities and learning the substructure of the textual relation through CNNs, the predictive performance of knowledge graph completion can be improved.

In order to gain insight into UGE, this study starts with a simple modeling and examines design choices not considered in the work of Toutanova et al. (2015). Therefore, comparison with them as a model is outside the focus of this study.

4.3 Universal Graph

A UG consists of a union set of knowledge graph triplets $\mathscr{T}_{\text{KG}} = \{\langle h, r_{\text{KG}}, t\rangle | h, t \in \mathscr{E}, r_{\text{KG}} \in \mathscr{R}_{\text{KG}}\}$ and textual triples $\mathscr{T}_{\text{text}} = \{\langle h, r_{\text{text}}, t\rangle | h, t \in \mathscr{E}, r_{\text{text}} \in \mathscr{R}_{\text{text}}\}$, where $r_{\text{KG}} \in \mathscr{R}_{\text{KG}}$ is the relation on the knowledge graph (KG relation) and $r_{\text{text}} \in \mathscr{R}_{\text{text}}$ is the relation extracted from the text (textual relation). The representation format of r_{text} is arbitrary, but in this study, we assume that it is an anonymized word sequence containing several words between and before a mention pair, in consideration of integration with language models. For example, from the text "*<u>Chainer</u> has been used at <u>PFN</u> since 2015 ...*," we obtain the triplet ⟨*Chainer*, *"[ENT] has been used at [ENT] since 2015"*, *PFN*⟩. The path p to be sampled from UG can contain both relations on the knowledge base and relations extracted from the text:

$$p = r_1 / \dots / r_l, \quad r_1, \dots, r_l \in \mathscr{R}_{\text{KG}} \cup \mathscr{R}_{\text{text}} \tag{4.3}$$

UGE implies embedding each element $\boldsymbol{h}, \boldsymbol{t}, \boldsymbol{r}_{\text{KG}}, \boldsymbol{r}_{\text{text}}$ of UG in a common vector space $\mathbb{R}^d$.

[2]They adopted the inter-mention engagement path as the textual relation, but the same argument can be made for the inter-mention word sequence.

4.4 Model

Our model is based on TransE (Bordes et al., 2013b), which is the most basic model of knowledge base embedding. To take into account the learning of multi-hop paths, we adopt Equation 4.2 as the score function. Since the paths we sample from the UG contain different kinds of relations, we vary the way we compute the fixed-length vector $\boldsymbol{r} \in \mathbb{R}^d$ depending on whether the input r is knowledge-base or text-derived according to the following equation:

$$\boldsymbol{r} = \begin{cases} \boldsymbol{R}(r), & r \in \mathscr{R}_{\mathrm{KG}} \\ g(\mathrm{Encoder}\,(r)), & r \in \mathscr{R}_{\mathrm{text}}, \end{cases} \tag{4.4}$$

where $\boldsymbol{R}$ is a lookup function for embedding relations, $\mathrm{Encoder}$ is an arbitrary encoder that takes a word sequence as input and returns a fixed-length h-dimensional vector, and $g \in \mathbb{R}^{h \times d}$ is a function that projects the vector output by the encoder into the vector space of the knowledge base.

The property that must be satisfied by $\mathrm{Encoder}$ in UGE is that textual relations that represent semantically similar relations become similar vectors. The simplest way for $\mathrm{Encoder}$ to satisfy this property is to pre-train a language model with a corpus of textual relations and then construct $\mathrm{Encoder}$ from the language model. In this study, we used the k-layer Bi-LSTM as $\mathrm{Encoder}$, but it is also possible to use a high-performance sentence encoder based on a pre-trained language model such as Sentence-BERT (Reimers and Gurevych, 2019), which is a future work.

Under the properties of $\mathrm{Encoder}$ described above, the property that g must satisfy is that the vector of the encoding result of the textual relation is projected near the vector of the KG relation that represents a similar meaning. If the property of $\mathrm{Encoder}$ is satisfied satisfactorily, the property of g is also expected to be satisfied satisfactorily by the usual learning of knowledge base completion, but a deeper analysis of this point is left for future work.

4.4.1 Training

The training data is generated by a random walk on KG or UG. In other words, starting from the head entity h sampled from $\mathscr{E}$, the training data $\{(h_i, p_i, t_i)\}_{i=1}^{N}$ is generated by repeating N times to reach the tail entity t via the appropriate path $p = r_1 / \ldots / r_{|p|}$. In this study, we minimize the following loss function:

$$\mathscr{L}(\Theta) = \sum_{i=1}^{N} \sum_{t^* \in \mathscr{N}(h_i, p_i)} \max(0, [\gamma + f^- - f^+]) \tag{4.5}$$

$$f^- = f(h_i, p_i, t^*; \Theta), \quad f^+ = f(h_i, p_i, t_i; \Theta), \tag{4.6}$$

where t^* is a negative example for (h_i, p_i, t_i) and γ is the margin. Various loss functions have been proposed in the research on knowledge-based embedding (Wang et al., 2017), and a comparative study with other loss functions is a future work. The path sampling method follows Takahashi et al. (2018).

4.5 Experiments

4.5.1 Settings

In order to understand the behavior of the UGE model, we created and experimented with a controlled evaluation dataset. To create the dataset, we constructed a seed UG from a real knowledge base and a text corpus, and took a subset of the graph as the evaluation dataset.

Seed UG

The construction of a Universal Graph requires a knowledge base and an entity-linked text corpus. In this study, we constructed the UG based on UMLS[3], a knowledge base in the biomedical domain, and MEDLINE[4], a literature database in the same domain.

As described in Section 4.3, the UG consists of a knowledge graph $\mathscr{T}_{\text{KG}}$ and textual triples $\mathscr{T}_{\text{text}}$. First, we collect a subset of UMLS as the knowledge graph $\mathscr{T}_{\text{KG}}$. In order to make sense of the multi-hop paths sampled from the knowledge base, we filtered the UMLS entities using the information of the types associated with the relationships. Specifically, the relation type is RO (RO is defined as "has Relationship Other than synonymous, narrower, or broader." It includes *may_treat*, *disease_has_associated_gene*, and etc.) The entity types include Protein, Gene, Disease or Syndrome, Enzyme, Chemical, Sign or Symptom, and Pharmacologic Substance. Next, we collected textual triples $\mathscr{T}_{\text{text}}$ from MEDLINE. ScispaCy Neumann et al. (2019) was used to identify UMLS entities in the MEDLINE text. For each mention pair in a sentence that refers to an entity in $\mathscr{T}_{\text{KG}}$, the The word sequence be-

[3]https://www.nlm.nih.gov/research/umls/
[4]https://www.nlm.nih.gov/databases/download/pubmed_medline.html

Table 4.1 Statistics of evaluation datasets.

		$\lvert\mathscr{E}\rvert$	$\lvert\mathscr{R}\rvert$	#Train	#Valid	#Test
UMLS.SYN	$\mathscr{T}_{\text{KG}}$	2.137	31	2,280	281	253
	$\mathscr{T}_{\text{text}}$	2,137	17,716	18,498	0	0
UMLS.NAT	$\mathscr{T}_{\text{KG}}$	988	25	2,382	294	264
	$\mathscr{T}_{\text{text}}$	715	20,663	21,595	0	0

tween them and at most two words before and after the mensuration were used as the textual relation.

Datasets

We constructed two types of evaluation datasets based on the seed UGs described in the previous section.

- UMLS.SYN: We sample triples from the seed UGs in such a way that the condition that for a given KG relation, there exists a textual relation that shares the entity pair is always satisfied.
- UMLS.NAT: Sample randomly to more closely resemble the actual knowledge graph while loosely satisfying the condition UMLS.SYN.

Note that for each entity pair, there should be at most one KG relation connecting them.

Evaluation Protocol

The knowledge base $\mathscr{T}_{\text{KG}}$ was divided into training, validation, and evaluation sets at a ratio of 8:1:1 for evaluation. The data with textual triples $\mathscr{T}_{\text{text}}$ added to the training set was used to train the model. The use of textual triples only for training is a similar setup to that of Toutanova et al. (2015). This means that by adding textual information to the existing knowledge base, we are evaluating how well we can supplement the missing triples in the knowledge base. In addition, we made sure that no unknown vocabulary in the training set appears in the validation/evaluation set at the time of segmentation. The statistics of the evaluation dataset are shown in Table 4.1.

The evaluation protocol follows the *filtered* setting of Bordes et al. (2013b); we compute a score $f(h, r, e)$ for all e except for entities $e \in \mathscr{E}$ such that the triples are included in the training, validation and evaluation set. The computed scores are converted to ranks along with the scores $f(h, r, t)$ for the gold triplets, and the evaluation index is obtained based on the

Table 4.2 Predictive performance of knowledge graph completion on the UMLS.SYN dataset.

		Valid			Test		
Model	LM-pret	MR	MRR	H10	MR	MRR	H10
Ks		287.3	0.167	0.311	305.1	0.155	0.300
Km		256.3	0.178	0.342	265.0	0.160	0.330
Us		51.6	0.383	0.699	54.3	0.346	0.617
Us	✓	43.0	0.377	0.698	45.9	0.342	0.617
Um		65.7	0.368	0.683	61.2	0.338	0.638
Um	✓	70.5	0.357	0.694	66.3	0.305	0.642

predicted rank for the gold entities t. The mean of the reciprocal of the predicted rank (Mean Reciprocal Rank; MRR) and The proportion of gold entities ranked in the top k (Hits@k) is reported.

We compare following four models:

- KG-single (Ks): samples only single-hop paths (paths of length one) from KG $\mathscr{T}_{\mathrm{KG}}$. This setting is equivalent to Bordes et al. (2013b).
- KG-multi (Km): samples single-hop and multi-hop paths (paths of length at least two) from KG. This setting is equivalent to Guu et al. (2015b).
- UG-single (Us): samples only single-hop paths (paths of length one) from UG $\mathscr{T}_{\mathrm{KG}} \cup \mathscr{T}_{\mathrm{text}}$. This setting is equivalent to Toutanova et al. (2015).
- UG-multi (Um): samples single-hop and multi-hop paths (paths of length at least two) from UG.

Pre-training of Language Models

As mentioned in Section 4.4, we believe that pre-training the language model is a promising approach for better embedding of UGs. In this experiment, we pre-trained the Bi-LSTM language model from about 200,000 textual relations, including the textual relations of UMLS.SYN and UMLS.NAT. For the Us and Um models, we trained and evaluated them separately by (1) randomly initializing the parameters in the $\mathrm{Encoder}$ part and (2) replacing them with the pre-trained Bi-LSTM parameters.

Table 4.3 Predictive performance of knowledge graph completion on the UMLS.NAT dataset.

		Valid			Test		
Model	LM-pret	MR	MRR	H10	MR	MRR	H10
Ks		115.9	0.123	0.255	106.8	0.115	0.267
Km		76.7	0.141	0.274	70.6	0.125	0.269
Us		73.1	0.138	0.282	73.5	0.118	0.261
Us	✓	75.5	0.155	0.330	77.8	0.122	0.273
Um		102.6	0.089	0.162	96.1	0.061	0.148
Um	✓	91.8	0.089	0.184	84.1	0.075	0.184

Table 4.4 The types of paths that each model can sample.

	Ks	Km	Us	Um
Single-hop path consisting of KG relations	✓	✓	✓	✓
Multi-hop path consisting of KG relations		✓		✓
Single-hop path consisting of textual relations			✓	✓
Multi-hop path with textual relations				✓

4.5.2 Results

Table 4.2 shows the performance of knowledge base completion for each model on the UMLS.SYN dataset. First, the UG-based models (Us, Um) show a significant gain from the KG-based models (Ks, Km). Since each KG relation in UMLS.SYN always has a textual relation that shares the head and tail entities, the textual relation is likely to be a powerful cue in predicting the unknown KG triple. Therefore, the high performance of UMLS.SYN means that it is able to learn so that the textual relation and the KG relation it implies are close in vector space. On the other hand, we did not observe a consistent change in performance with compositional training or pre-training of the language model.

Table 4.3 shows the performance of knowledge base completion for each model on the UMLS.NAT dataset. Compared to the results for UMLS.SYN, the performance improvement by UG is very small. In addition, Um performs worse than the most basic model Km in terms of MRR and H10. This may reflect the inherent difficulty of learning relational paths in UG, which is discussed in detail in 4.5.3.

Learning Strategy

In compositional training of knowledge graph embedding, single-hop paths and multi-hop paths capture different aspects of the knowledge graph, so learning strategies such as which

Table 4.5 Performance variation of knowledge graph completion by learning strategies on the UMLS.NAT dataset.

		Valid			Test		
Model	LM-pret	MR	MRR	H10	MR	MRR	H10
Noisy-to-clean							
Um		102.6	0.089	0.162	96.1	0.061	0.148
Um-Us		73.8	0.150	0.313	82.4	0.104	0.258
Um-Us-Km		55.2	0.164	0.389	58.8	0.127	0.328
Um-Us-Km-Ks		55.8	0.166	0.383	59.8	0.129	0.337
Um	✓	91.8	0.089	0.184	84.1	0.075	0.184
Um-Us	✓	71.9	0.150	0.313	65.0	0.119	0.267
Um-Us-Km	✓	57.4	0.168	0.379	53.8	0.131	0.331
Um-Us-Km-Ks	✓	58.7	0.169	0.405	55.1	0.139	0.345
Us	✓	75.5	0.155	0.330	77.8	0.122	0.273
Us-Km	✓	61.2	0.173	0.371	62.7	0.137	0.318
Us-Km-Ks	✓	63.1	0.178	0.374	63.2	0.139	0.331
Clean-to-noisy							
Ks		115.9	0.123	0.255	106.8	0.115	0.267
Ks-Km		76.6	0.140	0.274	70.6	0.126	0.267
Ks-Km-Us		77.3	0.145	0.301	75.3	0.115	0.265
Ks-Km-Us-Um		76.9	0.144	0.298	74.9	0.115	0.263

path to sample more during the training phase is one important design choice that has not been explored in detail in existing research. For example, Guu et al. (2015b) reported that they pre-trained their model on data consisting of single-hop paths and then trained on multi-hop paths, but they did not report the results of ablation experiments. Since relational paths on a UG contain different types of relations, it is increasingly important to consider effective learning strategies.

In this study, we consider a strategy based on the noisy nature of relational paths on KG and UG. A noisy relational path means that the head entity and the tail entity connected by the relational path do not have a meaningful relationship. A noisy relational path means that the head and tail entities connected by the relational path do not have a meaningful relationship. For example, the multi-hop path of the KG relation is likely to be noisy due to the small-world nature of the knowledge graph. For example, the multi-hop path of the KG relation is likely to be noisy due to the small-world nature of the knowledge graph, and the single-hop path of the textual relation is conceptually the same as the noise in relation extraction by far-field supervised learning. In this study, we order the relation paths from

Table 4.6 MRR for each shortest path length.

		Shortest Path Length		
Model	LM-pret	1	2	3 or more
Ks		0.175	0.184	0.041
Km		0.205	0.220	0.038
Us		0.226	0.149	0.041
Us	✓	0.256	0.179	0.037
Um		0.130	0.096	0.043
Um	✓	0.123	0.102	0.048
Um-Us-Km-Ks	✓	0.292	0.162	0.041
Us-Km-Ks	✓	0.317	0.157	0.040

noisy to noisy. Multi-hop paths containing the textual relation, single-hop paths consisting of the textual relation a single-hop path consisting of textual relations, a multi-hop path consisting of KG relation, a multi-hop path consisting of KG relation, and single-hop paths consisting of and The *Noisy-to-clean* strategy, which proceeds from noisy to clean models, and We compare the *Noisy-to-clean* strategy, which moves from a noisy model to a clean model, and the *Clean-to-noisy* strategy, which does the opposite. Table 4.4 summarizes the types of paths that can be sampled by each model. The models are considered to be more susceptible to noisy paths in the order Ks, Km, Us, and Um.

Each time a model is trained, hyperparameters are searched for, and the parameters of the model with the largest MRR in the validation set are inherited and re-trained.

The experimental results for UMLS.NAT are shown in Table 4.5. The "Um-Us" indicates that Us was learned after Um. From the results, we can see that the *Noisy-to-clean* strategy consistently improves the performance. We can also see that the performance of the final state for each learning strategy outperforms that of the stand-alone learning strategy as shown in Table 4.3. On the other hand, no performance improvement was observed for the *Clean-to-noisy* strategy due to prior learning. In other words, Ks-Km and Ks-Km-Us did not exceed the performance of the final-state model alone, and Ks-Km-Us-Um actually recorded the highest MRR in the first epoch and then continued to deteriorate its MRR. These results suggest that the paradigm of fine-tuning to each task after prior learning in recent research on language models may be effectively transferred to learning UG embeddings.

4.5.3 Analysis

Evaluation by Shortest Path Length

In order to identify the source of the gain in the UG-based model, we In order to identify the source of the gain of the UG-based model, we partitioned the validation data of the UMLS.NAT dataset into subsets based on the properties of the graph. The MRR was calculated for each partition. As a partitioning method, we used the shortest path length in the entity pair training data.

Table 4.6 shows the evaluation results. The UG-based model, except for Um, shows significantly higher performance in the subset with the shortest path length of 1 than the KG-based model. In the case where the shortest path length is 1, the construction method described in Section 4.5.1 indicates that the path will always consist of a single-hop path with a textual relation. Thus, the high performance of the subset with the shortest path length of 1 is a consequence of the Therefore, the high performance of the subset with the shortest path length of 1 means that it is able to learn the textual relation and its implied KG relation to be close in vector space.

In the subset where the shortest path length is 3 or more, the performance of all models is low and there is no difference between models. The difficulty of prediction in this subset can be explained as follows. In other words, in order for a knowledge-based embedding model to make correct predictions, it must The entity pairs in the vector space must be geometrically correctly aligned. When the shortest path of an entity pair is long, correct placement of the entity pair requires If the shortest path of an entity pair is long, in order to place them correctly, it must be learned so that the composition of the relational paths it traverses along the way represents the relationship between the entity pairs. A possible solution is to bias the learning to sample longer paths. However, this would mean that the learning would be more sensitive to noise in the relational paths. This suggests that we need a mechanism to reduce the noise of the paths involved in order to improve the performance in the future.

Weighting of Learning Rates

Since the encoding method is different between the KG relation and the textual relation, the amount of movement in the vector space for each parameter update received from the loss function is also different. Since the learning cycle of Us and Um includes both the update of Θ_{KG}, a parameter related to the vectors in the KG relation, and the update of Θ_{text}, a parameter related to the vectors in the textual relation, preliminary experiments have shown that the difference in the amount of movement leads to instability in learning. Therefore, in this study, we set separate initial learning rates for Θ_{KG} and Θ_{text}.

Table 4.7 Performance impact of weighting the learning rate on the UMLS.SYN dataset.

w	MRR
10^{1}	0.184 ± 0.161
10^{0}	0.256 ± 0.104
10^{-1}	0.319 ± 0.010
10^{-2}	$\mathbf{0.329 \pm 0.039}$
10^{-3}	0.291 ± 0.023

Table 4.8 Examples of nearest neighbor for KG relations.

Relation	Pseudo target	Dist
may_treat	-	0.000
“, cefotaxime or [ENT] should be used for the empiric treatment suspected severe [ENT] childhood .”	-	0.669
“were treated with [ENT] a parenteral cephalosporin antibiotic for 45 episodes of [ENT] by a variety”	*may_be_treated_by*$^{-1}$	0.677
“, statins , [ENT] , and LDL-apheresis , the patient developed symptomatic [ENT] the age of”	*may_treat*	0.689
“, were given [ENT] (AMK) for suspected [ENT]”	*may_prevent*	0.693
“potassium clavulanate and [ENT] were compared in a clinical trial 78 hospitalized patients with [ENT]”	*may_treat*	0.693

Ablation experiments were conducted to confirm the stabilizing effect of setting different initial learning rates. When the initial learning rate for Θ_{KG} is set as α_{KG}, the initial learning rate for Θ_{text} is α_{text} for α_{text} using the weight w as $\alpha_{\text{text}} := w\alpha_{\text{KG}}$. $\alpha_{\text{KG}} = \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}\}$, $w = \{10^{1}, 10^{0}, 10^{-1}, 10^{-2}, 10^{-3}\}$, grid search in the range The mean and standard deviation of MRR were calculated for each w. The data is UMLS.SYN and the model is Us.

The results are shown in Table 4.7. Compared to the case where the initial learning rate is not weighted ($w = 10^{0}$) Reducing the initial learning rate for Θ_{text} ($w = 10^{-1}, 10^{-2}$) It can be seen that reducing the initial learning rate for Θ_{text} ($w = 10^{-1}, 10^{-2}$) stabilizes learning and improves performance. In practice, since w is a value that depends on the architecture of the model In practice, since w is a value that depends on the architecture of the model, there is no need to perform the hyperparameter search again with different data.

4.6 Related Works

4.6.1 Using Multi-hop Paths in Knowledge Graph Completion

In the field of knowledge base completion, various attempts to adopt multi-hop paths on the knowledge base for learning have been reported. For example, Yang et al. (2014) and Guu et al. (2015b) have proposed compositional embedding methods, and Sun et al. (2019) have used complex spaces to achieve compositionality. An approach that estimates the reliability of paths and collects only useful paths (Lin et al., 2015b), and an approach that adopts reinforcement learning for multi-hop path exploration (Das et al., 2018; Lin et al., 2018; Xiong et al., 2017). However, all of these studies only focus on the embedding of KGs alone, and do not cover the embedding of UGs integrated with textual information. As reported in this chapter, our UG embedding aims to embed KG edges and text edges, which are very different in nature, in the same space, and we need to solve new problems that have not been seen in KG embedding.

4.6.2 Using Textual Information in Knowledge Graph Completion

The problem of extracting relational knowledge from a set of texts and extending the KG has been actively studied as tasks such as relation extraction and Knowledge Base Population (An et al., 2018,?; Cao et al., 2017; Das et al., 2017b; Fu et al., 2019; Han et al., 2016; Mousselly-Sergieh et al., 2018; Neelakantan et al., 2015b; Toutanova et al., 2016b; Wang et al., 2019; Xie et al., 2016). However, all of these studies target the problem of predicting the relationship between two given entities (relational prediction problem), and take the approach of collecting relevant textual and KG information about the input entity pairs and using them as features for prediction. In other words, our approach is essentially different from UG embedding in that we need to refer to UG or equivalent resources to collect relevant information during prediction. This difference is important when considering large-scale KG extensions. The approach that attributes the extension of KG to a relational prediction problem requires solving a relational prediction problem of the order of the square of the number of entities in the KG, which limits the scalability. On the other hand, an approach that embeds the entire UG in the vector space has the potential to solve this problem, since it does not need to refer to the UG during prediction. As mentioned in Section 4.1, to the best of our knowledge, there is no other report on UG embedding except for the work of Toutanova et al. (2015). In contrast, in this chapter, as the first study to investigate the properties of UGE, we empirically analyze the behavior of the basic model of UGE from three new perspectives:

(1) learning strategies, (2) extension to relational paths, and (3) integration with pre-trained language models.

4.7 Conclusion and Future Work

In this chapter, we empirically analyze the behavior of KGE models in terms of (1) learning strategies, (2) extension to relational paths, and (3) integration with pre-trained language models in order to gain insight into UGE. We empirically analyzed the behavior of KGE models from the following three perspectives. (1) In the learning strategy, the *Noisy-to-clean* strategy provides additional gain, (2) the extension to relational paths suggests that noise in the relational paths needs to be reduced, and (3) the integration with pre-trained language models does not have a significant effect on a small dataset and model like this one. (3) Integration with pre-trained language models did not have a significant effect on small datasets and models such as this one. In the future, we plan to expand the evaluation dataset and models, train UGE and language models simultaneously, and deal with the noise in the relation paths.

Acknowledgments

This work was supported by JST CREST Grant Number JPMJCR1301, Japan, and JSPS KAKENHI Grant Number JP18J20936.

Chapter 5

Conclusions

We discussed machine learning approaches in two problem settings for multi-hop reasoning over relational knowledge: predicate logic and propositional logic. Even in the simpler problem setting of relational reasoning, knowledge graph completion, where the tail prediction of the given triple, there are still many issues that need to be addressed. The integration of structured and unstructured data is useful not only in knowledge graph completion but also in any problem setting. In order to make the experimental setup closer to reality, the whole community needs to make efforts.

References

(1999). *Kiken-yosoku-master*. Chubu Nippon Driver School.

Althoff, M., Stursberg, O., and Buss, M. (2009). Model-based probabilistic collision detection in autonomous driving. *IEEE Trans. Intelligent Transportation Systems*, 10(2):299–310.

Ambardekar, A., Nicolescu, M., Bebis, G., and Nicolescu, M. N. (2014). Vehicle classification framework: a comparative study. *EURASIP J. Image and Video Processing*, 2014:29.

An, B., Chen, B., Han, X., and Sun, L. (2018). Accurate text-enhanced knowledge graph representation learning. 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 745–755.

Armand, A., Filliat, D., and Guzman, J. I. (2014). Ontology-based context awareness for driving assistance systems. In *2014 IEEE Intelligent Vehicles Symposium Proceedings, Dearborn, MI, USA, June 8-11, 2014*, pages 227–233.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. G. (2007). Dbpedia: A nucleus for a web of open data. In *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007.*, pages 722–735.

Bengler, K., Dietmayer, K., Färber, B., Maurer, M., Stiller, C., and Winner, H. (2014). Three decades of driver assistance systems: Review and future perspectives. *IEEE Intell. Transport. Syst. Mag.*, 6(4):6–22.

Berant, J., Chou, A., Frostig, R., and Liang, P. (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. Association for Computational Linguistics.

Bollacker, K. D., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250.

Bordes, A., Usunier, N., García-Durán, A., Weston, J., and Yakhnenko, O. (2013a). Translating embeddings for modeling multi-relational data. In *Advances in Neural Information*

Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States., pages 2787–2795.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. (2013b). Translating embeddings for modeling multi-relational data. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 2787–2795. Curran Associates, Inc.

Bottou, L. (2012). Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, pages 421–436. Springer.

Broadhurst, A., Baker, S., and Kanade, T. (2005). Monte carlo road safety reasoning. In *Intelligent Vehicles Symposium, 2005. Proceedings. IEEE*, pages 319–324.

Cao, Y., Huang, L., Ji, H., Chen, X., and Li, J. (2017). Bridge text and knowledge by learning multi-prototype entity mention embedding. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1623–1633.

Dai, Q., Inoue, N., Reisert, P., Ryo, T., and Inui, K. (2019a). Incorporating chains of reasoning over knowledge graph for distantly supervised biomedical knowledge acquisition. In *Proceedings of the 33nd Pacific Asia Conference on Language, Information and Computation (PACLIC33)*, pages 19–28, Hakodate, Japan. Waseda Institute for the Study of Language and Information.

Dai, Q., Inoue, N., Reisert, P., Takahashi, R., and Inui, K. (2019b). Distantly supervised biomedical knowledge acquisition via knowledge graph based attention. In *Proceedings of the Workshop on Extracting Structured Knowledge from Scientific Publications*, pages 1–10, Minneapolis, Minnesota. Association for Computational Linguistics.

Das, R., Dhuliawala, S., Zaheer, M., Vilnis, L., Durugkar, I., Krishnamurthy, A., Smola, A., and McCallum, A. (2018). Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *International Conference on Learning Representations*.

Das, R., Neelakantan, A., Belanger, D., and McCallum, A. (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: Volume 1, Long Papers*, pages 132–141, Valencia, Spain. Association for Computational Linguistics.

Das, R., Neelakantan, A., Belanger, D., and McCallum, A. (2017b). Chains of reasoning over entities, relations, and text using recurrent neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 132–141, Valencia, Spain. Association for Computational Linguistics.

Dettmers, T., Pasquale, M., Pontus, S., and Riedel, S. (2018). Convolutional 2d knowledge graph embeddings. In *Proceedings of the 32th AAAI Conference on Artificial Intelligence*.

Dollár, P., Wojek, C., Schiele, B., and Perona, P. (2009). Pedestrian detection: A benchmark. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 304–311.

Dollár, P., Wojek, C., Schiele, B., and Perona, P. (2012). Pedestrian detection: An evaluation of the state of the art. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(4):743–761.

Duchi, J. C., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159.

Enzweiler, M. and Gavrila, D. M. (2009). Monocular pedestrian detection: Survey and experiments. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(12):2179–2195.

Erhan, D., Bengio, Y., Courville, A. C., Manzagol, P., Vincent, P., and Bengio, S. (2010). Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660.

Ess, A., Leibe, B., Schindler, K., and Gool, L. J. V. (2008). A mobile vision system for robust multi-person tracking. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), 24-26 June 2008, Anchorage, Alaska, USA*.

Fu, C., Chen, T., Qu, M., Jin, W., and Ren, X. (2019). Collaborative policy learning for open knowledge graph reasoning. *arXiv preprint arXiv:1909.00230*.

Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the KITTI vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pages 3354–3361.

Gutmann, M. and Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13:307–361.

Guu, K., Miller, J., and Liang, P. (2015a). Traversing knowledge graphs in vector space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 318–327, Lisbon, Portugal. Association for Computational Linguistics.

Guu, K., Miller, J., and Liang, P. (2015b). Traversing knowledge graphs in vector space. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 318–327, Lisbon, Portugal. Association for Computational Linguistics.

Han, X., Liu, Z., and Sun, M. (2016). Joint representation learning of text and knowledge for knowledge graph completion. *arXiv preprint arXiv:1611.04125*.

Hayashi, K. and Shimbo, M. (2017). On the equivalence of holographic and complex embeddings for link prediction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 554–559, Vancouver, Canada. Association for Computational Linguistics.

Hixon, B., Clark, P., and Hajishirzi, H. (2015). Learning knowledge graphs for question answering through conversational dialog. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 851–861, Denver, Colorado. Association for Computational Linguistics.

Hobbs, J. R., Stickel, M. E., Appelt, D. E., and Martin, P. A. (1993). Interpretation as abduction. *Artif. Intell.*, 63(1-2):69–142.

Inoue, N., Kuriya, Y., Kobayashi, S., and Inui, K. (2015). Recognizing Potential Traffic Risks through Logic-based Deep Scene Understanding. In *Proceedings of the 22nd ITS World Congress*.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada*, pages 133–142.

Kadlec, R., Bajgar, O., and Kleindienst, J. (2017). Knowledge base completion: Baselines strike back. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 69–74, Vancouver, Canada. Association for Computational Linguistics.

Lefèvre, S., Vasquez, D., and Laugier, C. (2014). A survey on motion prediction and risk assessment for intelligent vehicles. *Robomech Journal*, 1(1):1.

Lin, X. V., Socher, R., and Xiong, C. (2018). Multi-hop knowledge graph reasoning with reward shaping. *arXiv preprint arXiv:1808.10568*.

Lin, Y., Liu, Z., Luan, H., Sun, M., Rao, S., and Liu, S. (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. Association for Computational Linguistics.

Lin, Y., Liu, Z., Luan, H., Sun, M., Rao, S., and Liu, S. (2015b). 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.

Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X. (2015c). Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 2181–2187.

Lindberg, D. A. B., Humphreys, B. L., and McCray, A. T. (1993). The unified medical language system. *Methods of information in medicine*, 32 4:281–91.

McInnes, L. and Healy, J. (2018). UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction. *ArXiv e-prints*.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013). Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia. Association for Computational Linguistics.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41.

Mohammad, M. A., Kaloskampis, I., Hicks, Y., and Setchi, R. (2015). Ontology-based framework for risk assessment in road scenes using videos. In *19th International Conference in Knowledge Based and Intelligent Information and Engineering Systems, KES 2015, Singapore, 7-9 September 2015*, pages 1532–1541.

Mousselly-Sergieh, H., Botschen, T., Gurevych, I., and Roth, S. (2018). A multimodal translation-based approach for knowledge graph representation learning. In *Proceedings of the Seventh Joint Conference on Lexical and Computational Semantics*, pages 225–234.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel*, pages 807–814.

Neelakantan, A., Roth, B., and McCallum, A. (2015a). Compositional vector space models for knowledge base completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 156–166, Beijing, China. Association for Computational Linguistics.

Neelakantan, A., Roth, B., and McCallum, A. (2015b). Compositional vector space models for knowledge base completion. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 156–166.

Neumann, M., King, D., Beltagy, I., and Ammar, W. (2019). ScispaCy: Fast and Robust Models for Biomedical Natural Language Processing. In *Proceedings of the 18th BioNLP Workshop and Shared Task*, pages 319–327, Florence, Italy. Association for Computational Linguistics.

Nguyen, D. Q., Sirts, K., Qu, L., and Johnson, M. (2016). Stranse: a novel embedding model of entities and relationships in knowledge bases. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 460–466, San Diego, California. Association for Computational Linguistics.

Nickel, M., Murphy, K., Tresp, V., and Gabrilovich, E. (2016a). A review of relational machine learning for knowledge graphs. *Proceedings of the IEEE*, 104(1):11–33.

Nickel, M., Rosasco, L., and Poggio, T. A. (2016b). Holographic embeddings of knowledge graphs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 1955–1961.

Nickel, M., Tresp, V., and Kriegel, H.-P. (2011). A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11, pages 809–816, USA. Omnipress.

Pawar, S., Palshikar, G. K., and Bhattacharyya, P. (2017). Relation extraction : A survey. *ArXiv*, abs/1712.05191.

Pujara, J., Augustine, E., and Getoor, L. (2017). Sparsity and noise: Where knowledge graph embeddings fall short. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1751–1756, Copenhagen, Denmark. Association for Computational Linguistics.

Qiu, D., Zhang, Y., Feng, X., Liao, X., Jiang, W., Lyu, Y., Liu, K., and Zhao, J. (2019). Machine reading comprehension using structural knowledge graph-aware network. 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 5895–5900, Hong Kong, China. Association for Computational Linguistics.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Rendon-Velez, E., Horváth, I., and Opiyo, E. (2009). Progress with situation assessment and risk prediction in advanced driver assistance systems: A survey. In *Proceedings of the 16th ITS World Congress*.

Riedel, S., Yao, L., McCallum, A., and Marlin, B. M. (2013). Relation extraction with matrix factorization and universal schemas. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 74–84, Atlanta, Georgia. Association for Computational Linguistics.

Rubinstein, R., Bruckstein, A. M., and Elad, M. (2010). Dictionaries for sparse representation modeling. *Proceedings of the IEEE*, 98(6):1045–1057.

Schlichtkrull, M. S., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., and Welling, M. (2017). Modeling relational data with graph convolutional networks. *CoRR*, abs/1703.06103.

Shen, Y., Huang, P.-S., Chang, M.-W., and Gao, J. (2017). Modeling large-scale structured relationships with shared memory for knowledge base completion. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 57–68, Vancouver, Canada. Association for Computational Linguistics.

Shi, B. and Weninger, T. (2017). Proje: Embedding projection for knowledge graph completion. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 1236–1242.

Silberer, C. and Lapata, M. (2014). Learning grounded meaning representations with autoencoders. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 721–732, Baltimore, Maryland. Association for Computational Linguistics.

Socher, R., Chen, D., Manning, C. D., and Ng, A. Y. (2013). Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013.*

Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States., pages 926–934.

Socher, R., Pennington, J., Huang, E. H., Ng, A. Y., and Manning, C. D. (2011). Semi-supervised recursive autoencoders for predicting sentiment distributions. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 151–161, Edinburgh, Scotland, UK. Association for Computational Linguistics.

Souza, C. R. C. and Santos, P. E. (2011). Probabilistic logic reasoning about traffic scenes. In *Towards Autonomous Robotic Systems - 12th Annual Conference, TAROS 2011, Sheffield, UK, August 31 - September 2, 2011. Proceedings*, pages 219–230.

Sun, X., Matsuzaki, T., Okanohara, D., and Tsujii, J. (2009). Latent variable perceptron algorithm for structured classification. In *IJCAI*, volume 9, pages 1236–1242.

Sun, Z., Deng, Z.-H., Nie, J.-Y., and Tang, J. (2019). Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*.

Takahashi, R., Tian, R., and Inui, K. (2018). Interpretable and compositional relation learning by joint training with an autoencoder. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2148–2159, Melbourne, Australia. Association for Computational Linguistics.

Tian, R., Okazaki, N., and Inui, K. (2016). Learning semantically and additively compositional distributional representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1277–1287, Berlin, Germany. Association for Computational Linguistics.

Titov, I. and Khoddam, E. (2015). Unsupervised induction of semantic roles within a reconstruction-error minimization framework. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1–10, Denver, Colorado. Association for Computational Linguistics.

Toutanova, K. and Chen, D. (2015). Observed versus latent features for knowledge base and text inference. In *Proceedings of the 3rd Workshop on Continuous Vector Space Models and their Compositionality*, pages 57–66, Beijing, China. Association for Computational Linguistics.

Toutanova, K., Chen, D., Pantel, P., Poon, H., Choudhury, P., and Gamon, M. (2015). Representing text for joint embedding of text and knowledge bases. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1499–1509, Lisbon, Portugal. Association for Computational Linguistics.

Toutanova, K., Lin, V., Yih, W.-t., Poon, H., and Quirk, C. (2016a). Compositional learning of embeddings for relation paths in knowledge base and text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1434–1444, Berlin, Germany. Association for Computational Linguistics.

Toutanova, K., Lin, V., Yih, W.-t., Poon, H., and Quirk, C. (2016b). Compositional learning of embeddings for relation paths in knowledge base and text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1434–1444, Berlin, Germany. Association for Computational Linguistics.

Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., and Bouchard, G. (2016). Complex embeddings for simple link prediction. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 2071–2080.

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

Vrandečiundefined, D. and Krötzsch, M. (2014). Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85.

Wang, Q., Mao, Z., Wang, B., and Guo, L. (2017). Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.*, 29(12):2724–2743.

Wang, X., Gao, T., Zhu, Z., Liu, Z., Li, J., and Tang, J. (2019). Kepler: A unified model for knowledge embedding and pre-trained language representation. *arXiv preprint arXiv:1911.06136*.

Wang, Z., Zhang, J., Feng, J., and Chen, Z. (2014a). Knowledge graph and text jointly embedding. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1591–1601, Doha, Qatar. Association for Computational Linguistics.

Wang, Z., Zhang, J., Feng, J., and Chen, Z. (2014b). Knowledge graph and text jointly embedding. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1591–1601, Doha, Qatar. Association for Computational Linguistics.

Wang, Z., Zhang, J., Feng, J., and Chen, Z. (2014c). Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, July 27 -31, 2014, Québec City, Québec, Canada.*, pages 1112–1119.

Xiao, H., Huang, M., and Zhu, X. (2016). From one point to a manifold: Knowledge graph embedding for precise link prediction. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pages 1315–1321.

Xie, Q., Ma, X., Dai, Z., and Hovy, E. (2017). An interpretable knowledge transfer model for knowledge base completion. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 950–962, Vancouver, Canada. Association for Computational Linguistics.

Xie, R., Liu, Z., Jia, J., Luan, H., and Sun, M. (2016). Representation learning of knowledge graphs with entity descriptions. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Xiong, W., Hoang, T., and Wang, W. Y. (2017). Deeppath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 564–573.

Yamamoto, K., Inoue, N., Inui, K., Arase, Y., and Tsujii, J. (2015). Boosting the efficiency of first-order abductive reasoning using pre-estimated relatedness between predicates. *International Journal of Machine Learning and Computing*, 5(2):114.

Yang, A., Wang, Q., Liu, J., Liu, K., Lyu, Y., Wu, H., She, Q., and Li, S. (2019). Enhancing pre-trained language representations with rich knowledge for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2346–2357, Florence, Italy. Association for Computational Linguistics.

Yang, B., Yih, W.-t., He, X., Gao, J., and Deng, L. (2014). Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*.

Yang, B., Yih, W.-t., He, X., Gao, J., and Deng, L. (2015). Embedding Entities and Relations for Learning and Inference in Knowledge Bases. In *Proceedings of the 3rd International Conference on Learning Representations*, pages 1–12.

Zhang, S., Benenson, R., Omran, M., Hosang, J. H., and Schiele, B. (2016). How far are we from solving pedestrian detection? *CoRR*, abs/1602.01237.

Zhao, L., Ichise, R., Yoshikawa, T., Naito, T., Kakinami, T., and Sasaki, Y. (2015). Ontology-based decision making on uncontrolled intersections and narrow roads. In *2015 IEEE Intelligent Vehicles Symposium, IV 2015, Seoul, South Korea, June 28 - July 1, 2015*, pages 83–88.