UNIVERSITY OF CALIFORNIA

Los Angeles

Representation Learning based Query Answering on Knowledge Graphs

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Computer Science

by

Xuelu Chen

2021

ABSTRACT OF THE DISSERTATION

Representation Learning based Query Answering on Knowledge Graphs

by

Xuelu Chen
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2021
Professor Carlo Zaniolo, Co-Chair
Professor Yizhou Sun, Co-Chair

Knowledge graphs provide structured representations of facts about real-world entities and relations, serving as a vital knowledge source for numerous artificial intelligence applications. This dissertation seeks to extend the scope and provide theoretical guidance for representation learning based query answering on knowledge graphs. The incompleteness of knowledge graphs has recently motivated the use of representation learning models to generalize from known facts and infer new knowledge for query answering. Despite advances in answering atomic queries by representing deterministic facts within a monolingual knowledge graph, existing models must overcome the following three challenges: (i) they must address the need to incorporate uncertainty information into query answering, which is critical to many knowledge-driven applications; (ii) they must effectively leverage complementary knowledge from knowledge graphs in different languages; (iii) they must be able to embed complex first-order logical queries.

In this dissertation, we address the aforementioned challenges and extend the scope of query answering on knowledge graphs through contributions on the following three fronts: (i) To capture fact uncertainty and support reasoning under uncertainty, we propose two knowledge graph embedding models that are capable of encoding uncertain facts in the embedding space. Our proposed models thus learn entity and relation embeddings according to the confidence scores of uncertain facts. We introduce probabilistic soft logic to infer confidence scores to provide extra supervision for training. We also explore using box embeddings to embed uncertain knowledge graphs and imposing relation property constraints to enhance performance on sparse uncertain knowledge graphs. (ii) To effectively combine knowledge graphs in different languages, we introduce an ensemble learning framework that embeds all knowledge graphs in a shared embedding space, where the association of entities is captured based on self-learning. The framework performs ensemble inference to combine prediction results from embeddings of multiple language-specific knowledge graphs, for which multiple ensemble techniques are investigated. (iii) To support answering complex first-order logical queries, we present a query embedding framework based on fuzzy logic that allows us to define logical operators in a principled and learning-free manner, whereby learning is only required for entity and relation embeddings. The proposed model can further benefit when complex logical queries are available for training. As a result of this research we were able to identify some of the desirable properties that embedding models ought to possess and analyze which of the existing models have these properties. Therefore, the results presented in this dissertation advance the state-of-the-art of query answering on knowledge graphs along different axes and provide conceptual guidance for future research in this field.

The dissertation of Xuelu Chen is approved.

Junghoo Cho

Kai-Wei Chang

Muhao Chen

Carlo Zaniolo, Committee Co-Chair

Yizhou Sun, Committee Co-Chair

University of California, Los Angeles

2021

To my mother.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to express sincere gratitude and respect to my committee co-chairs Prof. Carlo Zaniolo and Prof Yizhou Sun for their insightful guidance and generous support throughout my doctoral studies. Their professionalism, rigor, diligence, and research foresight have polished up my research skills, influenced my life perspectives, and inspired my career pursuits.

I would like to thank my committee, Prof. Junghoo Cho, Prof. Kai-Wei Chang, and Prof. Muhao Chen, for their valuable time and feedback on my research. Professor Muhao Chen generously supported and closely collaborated with me on various research projects. He introduced me to research frontiers and taught me how to approach problems from a researcher's perspectives. I will always be grateful for his guidance and support.

I want to thank all my friends and fellow students at UCLA ScAi Lab for their friendship and collaboration, including Yunsheng Bai, Xiusi Chen, Ariyam Das, Changjun Fan, Jiaqi Gu, Junheng Hao, Ziniu Hu, Zijie Huang, Roshni Iyer, Song Jiang, Chelsea Ju, Mingda Li, Youfu Li, Ding Ling, Giuseppe M. Mazzeo, Arjun Sabramonian, Weijia Shi, Ankith Uppunda, Jin Wang, Yewen Wang, Patricia Xiao, and Derek Xu. I would also like to thank my collaborators from UMass Amherst, including Michael Boratko, Shib Sankar Dasgupta, Xiang Lorraine Li, and Prof. Andrew McCallum.

My parents have always shown me unconditional love and unbounded support throughout my life. Words cannot describe how much I appreciate and love them. My sincere appreciation also goes to my dear friends who have supported and cheered me on throughout my Ph.D. journey. You guys make my life colorful, and I am truly blessed to be in your company. I appreciate Shun Yao for accompanying me throughout most of my Ph.D. study, which I will always remember. Lastly, I wish to thank the Beatles for their inspirational songs that I have listened to throughout my teenage years, undergraduate years, and doctoral study.

VITA

2016	Bachelor of Engineering in Computer Science, Xi'an Jiaotong University, Shaanxi, China.
2018	Software Engineer Intern, Roche, Tucson, AZ.
2019	Software Engineer Intern, Facebook, Seattle, WA.
2020	Software Engineer Intern, Facebook, Menlo Park, CA.
2017-2021	Teaching Assistant/Associate/Fellow, Department of Computer Science, UCLA, Los Angeles, CA.

PUBLICATIONS

Xuelu Chen, Ziniu Hu, Yizhou Sun. Fuzzy Logic based Logical Query Answering on Knowledge Graphs. To appear in *Proceedings of the Thirty-sixth AAAI Conference on Artificial Intelligence* (AAAI 2022).

Xuelu Chen*, Michael Boratko*, Muhao Chen, Shib Sankar Dasgupta, Xiang Lorraine Li, and Andrew McCallum. Probabilistic Box Embeddings for Uncertain Knowledge Graph Reasoning. In *Proceedings of the 2021 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (NAACL 2021).

Xuelu Chen, Muhao Chen, Changjun Fan, Ankith Uppunda, Yizhou Sun and Carlo Zaniolo. Multilingual Knowledge Graph Completion via Ensemble Knowledge Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP): Findings* (Findings of EMNLP 2020).

Xuelu Chen, Muhao Chen, Weijia Shi, Yizhou Sun, Carlo Zaniolo. Embedding Uncertain Knowledge Graphs. In *Proceedings of the Thirty-third AAAI Conference on Artificial Intelligence* (AAAI 2019).

Muhao Chen*, Chelsea J-T Ju*, Guangyu Zhou, Xuelu Chen, Tianran Zhang, Kai-Wei Chang, Carlo Zaniolo, Wei Wang. Multifaceted protein–protein interaction prediction based on Siamese residual RCNN. In *Bioinformatics*, Oxford Academic. 2019.

Muhao Chen, Yingtao Tian, Xuelu Chen, Zijun Xue, Carlo Zaniolo. Embedding-based Relation Prediction for Ontology Population. In *Proceedings of the 17th SIAM International Conference on Data Mining* (SDM 2018).

CHAPTER 1

Introduction

1.1 Motivation

Knowledge graphs, such as Freebase [8], YAGO [68], and NELL [60], provide structured representations of facts about real-world entities and relations. They have been serving as a vital source of knowledge for numerous artificial intelligence applications [23, 41, 38], by providing answers to queries such as itendifying which songs John Lennon and Paul McCartney co-wrote. Despite their great importance, knowledge graphs are often incomplete [6], which prevents directly answering many queries by searching knowledge graphs; for example, 71% of people in Freebase [8] have no known place of birth, and 75% have no known nationality [27].

The incompleteness of knowledge graphs has motivated the use of KG embedding models to generalize and reason using known facts [9, 105, 91, 88]. These models encode entities as low-dimensional vectors and relations as algebraic operations among entity vectors in order to capture entity similarity and preserve the semantic relations between entities in the embedding space. Such models support automated knowledge graph completion in the form of atomic query answering, such as predicting the object for the subject-predicate-object triple *(John Lennon, livedIn, ?)* [9, 105, 91, 88]. In addition, these models provide crucial features for incorporating symbolic knowledge into machine learning to benefit numerous knowledge-driven tasks, including dialogue agents [38], question answering [23, 41], item recommendation [39], story comprehension [13], logic rule mining [105], and ontology population [16].

Despite extensive efforts devoted to knolwedge graph embedding [9, 98, 105, 91, 88], existing methods are limited to representing deterministic facts within a single monolingual KG and answering atomic queries while failing to address the following challenges:

1. **Uncertainty.** Existing models can neither encode uncertain facts such as *(fork, atLocation, kitchen)* with a confidence between 0 and 1 nor answer queries that involve reasoning under uncertainty, such as determining the probability that two proteins interact. Uncertainty is inherent in many forms of knowledge, however, and failure to capture this information may adversely affect prediction accuracy. For instance, while both *(Honda, competeswith, Toyota)* and *(Honda, competeswith, Chrysler)* appear somewhat correct, the former should have higher confidence than the latter, since both Honda and Toyota are Japanese car manufacturers with highly overlapping customer bases. As another example, although *(The Beatles, genre, Rock)* and *(The Beatles, genre, Pop)* are both true, the first may be given a higher level of confidence due to the Beatles' primary role as a rock band. This type of information is crucial when recommending music online or answering questions, such as *who is the main competitor of Honda?* Uncertainty information also facilitates extracting confident knowledge for drug repurposing [78], short text understanding [103], question answering [107] and named entity recognition [67].

2. **Multilingualism.** Existing embedding models cannot effectively leverage complementary knowledge from knowledge graphs in different languages for query answering. Combining several language-specific knowledge graphs for query answering offers potential benefits; for example, embedding models of well-populated knowledge graphs (e.g. English knowledge graphs) are expected to capture richer knowledge due to better data quality and denser graph structures [66], consequently providing ampler signals to facilitate query answering on sparser knowledge graphs. Furthermore, combining knowledge graphs across different languages enables leveraging knowledge that may be more prevalent in some knowledge graphs than in others, which likely leads to improved prediction accuracy. The oldest Japanese novel *The Tale of Genji* represents an example, as the English DBpedia [53] only records its genre as *Monogatari* (story), whereas the Japanese DBpedia identifies more genres, including *Love Story*, *Royal Family Related Story*, *Monogatari* and *Literature-Novel*. It is similarly reasonable to expect a Japanese knowledge graph embedding model to provide significant

advantages in answering queries about other Japanese cultural entities such as *Nintendo* and *Mount Fuji*.

3. **Complex query structure.** Answering complex logical queries on knowledge graphs remains an unresolved problem. This task involves multi-hop reasoning on knowledge graphs and serves the key step to answering complex natural language questions. For instance, the question "Who sang the songs written by John Lennon or Paul McCartney but never won a Grammy Award?" can be expressed as the first-order logical query $q : V_? : \exists V$ *Compose(John Lennon, V)* $\vee$ *Compose(Paul McCartney, V)* $\wedge$ $\neg$*AwardedTo(Grammy Award, V)* $\wedge$ *SungBy($V, V_?$)* and answered by executing the query on knowledge graphs. Recent studies [35, 69, 71] attempt to address the challenges of time complexity and knowledge incompleteness by embedding logical queries and entities into the same vector space. These methods nonetheless entail several limitations: First, the logic operators in these models are often defined ad-hoc, and many do not satisfy basic logic laws (e.g., the associative law $(\psi_1 \wedge \psi_2) \wedge \psi_3 \equiv \psi_1 \wedge (\psi \wedge \psi_3)$ for logical formulae ψ_1, ψ_2, ψ_3), which limits their inference accuracy; second, the logical operators of existing works are based on deep architectures, which require many training queries containing such logical operations in order to learn the parameters. This requirement greatly limits the models' scope of application, since it is challenging to collect a large number of reasonable complex queries with accurate answers.

1.2 Thesis Contributions

The contributions of this dissertation concern the aforementioned motivations and are summarized as follows:

1. To address fact uncertainty in knowledge graph query answering, this dissertation proposes two knowledge graph embedding methods that encode uncertain facts in the embedding space and support reasoning under uncertainty for query answering. The first model learns a non-linear regressor with a multi-relational structure encoder and incorporates Probablistic Soft Logic into the learning process to provide additional training supervision. It is the first

work that targets uncertain knowledge graph embedding and enables query answering, automated completion, fact ranking, and fact classification on uncertain knowledge graphs. The second model considers each entity as a binary random variable and models each entity as a box (i.e. axis-aligned hyperrectangle) in the vector space, with relations between two entities represented by affine transforms on the subject and object entity boxes. The geometry of the boxes endows the model with calibrated probabilistic semantics and facilitates injecting relation property constraints.

2. To transfer knowledge from knowledge graphs in different languages for query answering, this dissertation proposes a framework for ensemble learning of knowledge graph embedding models. This approach allows exchanging complementary knowledge across different language-specific knowledge graphs, thereby providing a versatile method of leveraging specific knowledge that is better captured in some knowledge graphs compared to others. We also investigate different ensemble techniques to combine prediction results from embeddings of multiple language-specific knowledge graphs, which enables assessing the credibility of prediction from different models and thus leads to a more accurate final prediction.

3. To support answering first-order logical queries, we present a fuzzy logic based logical query embedding framework for answering logical queries on knowledge graphs. We borrow the idea of fuzzy logic and use fuzzy conjunction, disjunction, and negation to implement logical operators in a more principled and learning-free manner. In addition, this dissertation identifies some of the basic properties that an embedding model ought to possess . This analysis provides theoretical guidance for future research on embedding-based logical query answering models.

1.3 Thesis Outline

The rest of this dissertation is organized as follows: We first survey the background in Section 2 before presenting two models that enable reasoning and query answering on uncertain knowledge graphs. Chapter 3 introduces the first uncertain knowledge graph embedding model called UKGE, where the embeddings of entities and relations are learned according to confidence scores, unlike previous models that characterize facts with binary classification techniques. We also introduce probabilistic soft logic to infer confidence scores to provide extra supervision during training. We propose two variants of UKGE based on various regression functions. In Chapter 4, we extend the technique to improve reasoning on sparse uncertain knowledge graphs. We provide BEUrRE, which is a novel uncertain knowledge graph embedding method endowed with probabilistic semantics. BEUrRE considers each entity as a binary random variable and models each entity as a box (i.e. axis-aligned hyperrectangle) in the vector space, with relations between two entities representing affine transforms on the subject and object entity boxes. The geometry of the boxes endows the model with calibrated probabilistic semantics and facilitates injecting relation property constraints on sparse knowledge graphs. Such representation is aligned with the human perception that entities or concepts have different levels of granularity.

In Chapter 5, we enhance question answering on deterministic knowledge graphs by transferring complementary knowledge across multiple language-specific knowledge graphs. The proposed framework KEns embeds all knowledge graphs in a shared embedding space, where the association of entities is captured based on self-learning. The KEns framework then performs ensemble inference to combine prediction results from embedding models of different language-specific knowledge graphs, for which multiple ensemble techniques are investigated.

Chapter 6 explores first-order logical query embedding. The proposed model FuzzQE follows fuzzy logic to define logical operators in a principled and learning-free manner, where only entity and realtion embeddings require learning. FuzzQE can further benefit when complex logical queries are available for training. In addition, Chapter 6 proposes basic properties that an embedding model ought to possess.

Finally, Chapter 7 concludes the dissertation and discusses avenues for future research.

CHAPTER 2

Background

In this chapter, we present the background on representation learning based query answering on knowledge graphs.

2.1 Knowledge Graphs

This section provides an introduction and categorization of knowledge graphs.

2.1.1 Monolingual and Multilingual Knowledge Graphs

A monolingual knowledge graph consists of entities and relations described in one language. Recent decades have witnessed the emergence of large-scale multilingual knowledge graphs, including Wikidata [95], DBpedia [53], ConceptNet [79], and YAGO [68]. In such multilingual knowledge graphs, vast amounts of knowledge are created in various language-specific versions that evolve independently. Multilingual knowledge graphs leverage entity and relation alignment to synchronize different language-specific versions. Aligning relations is usually feasible and has been de facto achieved in a number of major knowledge graphs, including the aforementioned DBpedia [53] and YAGO [95]. In contrast, entities in those major knowledge graphs are often so numerous that they cannot be easily aligned, and available entity alignment is only in small amounts [17].

2.1.2 Deterministic and Uncertain Knowledge Graphs

Knowledge graphs can be categorized into the following two types: (i) *Deterministic knowledge graphs*, such as YAGO [68] and FreeBase [8], consist of deterministic facts that describe semantic relations between entities; (ii) *Uncertain knowledge graphs* including ProBase [103], ConceptNet [79] and NELL [60] associate every fact with a confidence score that represents the likelihood of the fact to be true.

Recently, the development of relation extraction and crowdsourcing have enabled the construction of large-scale uncertain knowledge bases. ConceptNet [79] is a multilingual uncertain knowledge graph for commonsense knowledge that is collected via crowdsourcing. The confidence scores in ConceptNet mainly come from the co-occurrence frequency of the labels in crowdsourced task results. Probase [103] is a universal probabilistic taxonomy built by relation extraction. Every fact in Probase is associated with a marginal probability. NELL [60] collects facts by reading web pages and learns their confidence scores from semi-supervised learning with the Expectation-Maximum (EM) algorithm. Aforementioned uncertain knowledge graphs have enabled numerous knowledge-driven applications. For example, [100] utilizes Probase in short text understanding.

2.2 Representation Learning for Knowledge Graphs

Knowledge graph embedding models encode entities as low-dimensional vectors and relations as algebraic operations among entity vectors. They seek to capture the similarity of entities and preserve the structure of knowledge graphs in the embedding space. Embedding models for monolingual deterministic knowledge graphs have been extensively explored recently. A recent survey [97] categorizes these models into two groups: translational models and bilinear models.

Translational models share a common principle $\boldsymbol{h}_r + \boldsymbol{r} \approx \boldsymbol{t}_r$, where $\boldsymbol{h}_r$, $\boldsymbol{t}_r$ are the entity embeddings projected in an identical or projected embedding space. The forerunner of this family, TransE [9], lays $\boldsymbol{h}_r$ and $\boldsymbol{t}_r$ in a common space as $\boldsymbol{h}$ and $\boldsymbol{t}$ with regard to any relation r. Variants of TransE, such as TransH [98], TransR [57], TransD [43], and TransA, [44] differentiate the translations of entity embeddings in different language-specific embedded spaces based on different forms of relation-specific projections. Despite their simplicity, translational models achieve satisfying performance on knowledge graph completion and are robust against the sparsity of data [66]. Bilinear models [42] model relations as the second-order correlations between entities, using the scoring function $f(h,r,t) = \boldsymbol{h}^\top \boldsymbol{W_r t}$. This function is first adopted by RESCAL [62], a collective matrix factorization model. DistMult [105] constrains $\boldsymbol{W_r}$ as a diagonal matrix which reduces the computing cost and also enhances the performance. ComplEx extends the scoring function of DistMult into a complex embedding space, and HolE [61] substitutes the multiplication in DistMult with circular correlation. Both ComplEx and HolE lead to better characterization of asymmetric relations. There are also other models for deterministic knowledge graph embedding, such as neural models like Neural Tensor Network (NTN) [77] and ConvE [26].

Though embedding models for deterministic knowledge graphs have been extensively studied, embedding uncertain knowledge graphs has not been well explored. One recent work has proposed a matrix-factorization-based approach to embed uncertain networks [40]. However, it cannot be generalized to embed uncertain knowledge graphs because this model only considers the node proximity in the networks with no explicit relations and only generates node embeddings.

CHAPTER 3

Probabilistic Soft Logic Guided Uncertain Knowledge Graph Embedding

In this chapter, we introduce an uncertain knowledge graph embedding model to handle fact uncertainty for query answering on knowledge graphs.

3.1 Introduction

Knowledge graphs are categorized into the following two types: (i) *Deterministic knowledge graphs*, such as YAGO [68] and FreeBase [8], consist of deterministic facts that describe semantic relations between entities; (ii) *Uncertain knowledge graphs* including ProBase [103], ConceptNet [79] and NELL [60] associate every fact with a confidence score that represents the likelihood of the fact to be true.

While current embedding models focus on capturing deterministic knowledge, it is critical to incorporate uncertainty information into knowledge sources for several reasons. First, uncertainty is the nature of many forms of knowledge. An example of naturally uncertain knowledge is the interactions between proteins. Since molecular reactions are random processes, biologists label the protein interactions with their probabilities of occurrence and present them as uncertain knowledge graphs called Protein-Protein Interaction (PPI) Networks. Second, uncertainty enhances inference in knowledge-driven applications. For example, short text understanding often entails interpreting real-world concepts that are ambiguous or intrinsically vague. The probabilistic knowledge graph Probase [103] provides a prior probability distribution of concepts behind a term that has critically supported short text understanding tasks involving disambiguation [101, 100]. Furthermore, uncertain knowledge representations have largely benefited various applications, such as question answering [107] and named entity recognition [67].

Capturing the uncertainty information with knowledge graph embeddings remains an unresolved problem. This is a non-trivial task for several reasons. First, compared to deterministic knowledge graph embeddings, uncertain knowledge graph embeddings need to encode additional confidence information to preserve uncertainty. Second, current knowledge graph embedding models cannot capture the subtle uncertainty of unseen facts, as they assume that all the unseen facts are false beliefs and minimize the plausibility measures of facts. One major challenge of learning embeddings for uncertain knowledge graphs is to properly estimate the uncertainty of unseen facts.

To address the above issues, we propose a new embedding model UKGE (Uncertain Knowledge Graph Embeddings), which aims to preserve both structural and uncertainty information of facts in the embedding space. Embeddings of entities and relations on uncertain knowledge graphs are learned according to confidence scores. Unlike previous models that characterize facts with binary classification techniques, UKGE learns embeddings according to the confidence scores of uncertain facts. To further enhance the precision of UKGE, we also introduce probabilistic soft logic to infer the confidence score for unseen facts during training. We propose two variants of UKGE based on different embedding-based confidence functions. We conducted extensive experiments using three real-world uncertain knowledge graphs on three tasks: (i) *confidence prediction*, which seeks to predict confidence scores of unseen facts; (ii) *fact ranking*, which focuses on retrieving tail entities for the query $(h, r, \underline{?t})$ and ranking these retrieved tails in the right order; and (iii) *fact classification*, which decides whether or not a given fact is a *strong* fact. Our models consistently outperform the baseline models in these experiments.

3.2 Related Work

To the best of our knowledge, there has been no previous work on uncertain knowledge graph embeddings. Three lines of research are closely related to this topic: uncertain knowledge graphs, deterministic knowledge graph embedding models, and probabilistic reasoning. Uncertain knowledge graphs and deterministic knowledge graph embedding models are discussed in Sections 2.1.2 and 2.2 respectively. We hereby describe related work in probabilistic reasoning, particularly the probabilistic soft logic (PSL) framework [48] . A PSL program consists of a set of first-order logic rules. Every atom is assigned a *soft truth value* in $[0, 1]$ and PSL uses *Łukasiewicz t-norm* [59] to determine to which degree a rule is satisfied. In combination with Hinge-Loss Markov Random Field (HL-MRF), PSL is widely used in probabilistic reasoning tasks, such as social-trust prediction and preference prediction [4, 5]. In this chapter, we adopt PSL to infer confidence scores and provide extra training supervision, thereby enhancing the embedding model prediction accuracy.

3.3 Problem Definition

We define the uncertain knowledge graph embedding problem in this section by first providing the definition of uncertain knowledge graphs.

Definition 1. ***Uncertain Knowledge Graph.*** *An uncertain knowledge graph represents knowledge as a set of relations ($\mathcal{R}$) defined over a set of entities ($\mathcal{E}$). It consists of a set of weighted triples $\mathcal{G} = \{(l, s_l)\}$. For each pair (l, s_l), $l = (h, r, t)$ is a triple representing a fact where $h, t \in \mathcal{E}$ (the set of entities) and $r \in \mathcal{R}$ (the set of relations), and $s_l \in [0, 1]$ represents the confidence score for this fact to be true.*

Note that we assume the confidence score $s_l \in [0, 1]$ and interpret it as a probability to leverage probabilistic soft logic-based inference. The range of original confidence scores for some uncertain knowledge graph (e.g., ConceptNet) may not fall in $[0, 1]$, and normalization will be needed in these cases. Some examples of weighted triples are listed below.

Example 3.3.1. ***Weighted triples.***

1. *(choir,* `relatedto`*, sing): 1.00*

2. *(college,* `synonym`*, university): 0.99*

3. *(university,* `synonym`*, institute): 0.86*

4. (fork, `atlocation`, kitchen): 0.4

Definition 2. ***Uncertain Knowledge Graph Embedding Problem.*** *Given an uncertain knowledge graph $\mathcal{G}$, the embedding model aims to encode each entity and relation in a low-dimensional space in which structure information and confidence scores of facts are preserved.*

Notation wise, boldfaced $\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}$ are used to represent the embedding vectors for head h, relation r and tail t respectively. $\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}$ are assumed lie in $\mathbb{R}^k$.

3.4 Modeling

In this section, we propose our model for uncertain knowledge graph embeddings. The proposed model `UKGE` encodes the knowledge graph structure according to the confidence scores for both *observed* and *unseen* facts, such that the embeddings of facts with higher confidence scores receive higher plausibility values.

We first design fact confidence score modeling based on embeddings of entities and relations, then introduce how probabilistic soft logic can be used to infer confidence scores for unseen relations, and lastly describe the joint model `UKGE` and its two variants.

3.4.1 Embedding-based Confidence Score Modeling for facts

Unlike deterministic knowledge graph embedding models, uncertain knowledge graph embedding models need to explicitly model the confidence score for each triple and compare the prediction with the true score. We hereby first define and model the plausibility of triples, which can be considered as a unnormalized confidence score.

Definition 3. ***Plausibility.*** *Given a fact triple l, the plausibility $g(l) \in R$ measures how likely this fact holds. The higher plausibility value corresponds to the higher confidence score s.*

Given a triple $l = (h, r, t)$ and their embeddings $\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}$, we model the plausibility of (h, r, t) by the following function:

$$g(l) = \boldsymbol{r} \cdot (\boldsymbol{h} \circ \boldsymbol{t}) \tag{3.1}$$

where $\circ$ is the element-wise product, and $\cdot$ is the inner product. This function captures the relatedness between embeddings $\boldsymbol{h}$ and $\boldsymbol{t}$ under the condition of relation r and is first adopted by DistMult [105]. We employ this triple modeling technique for three reasons: (i) This technique has represented the state-of-the-art performance for modeling deterministic knowledge graphs [46], (ii) It agrees with the nature of our model to quantify the confidence of an uncertain fact by comparing the relation embeddings with the pair of head and tail embeddings, (iii) It does not introduce additional parameter complexity to the model like other techniques, such as TransH [98], TransR [57], ConvE [26] and ProjE [75]. Nevertheless, this scoring function can be further explored in future work.

3.4.1.1 From plausibility to confidence scores

In order to transform plausibility scores to confidence scores, we consider two different mapping functions and test them in the experimental section. Formally, let a triple be l and its plausibility score be $g(l)$, a transformation function $\phi(\cdot)$ maps $g(l)$ to a confidence score $f(l)$.

$$f(l) = \phi(g(l)), \phi : \mathbb{R} \rightarrow [0, 1] \tag{3.2}$$

Two choices of mapping ϕ are listed below.

Logistic function. One way to map plausibility values to confidence score is a logistic function as follows:

$$\phi(x) = \frac{1}{1 + e^{-(\mathbf{w}x+\mathbf{b})}} \tag{3.3}$$

Bounded rectifier. Another mapping is a bounded rectifier [14]:

$$\phi(x) = \min(\max(\mathbf{w}x + \mathbf{b}, 0), 1) \tag{3.4}$$

where $\mathbf{w}$ is a weight $\mathbf{b}$ is a bias.

3.4.2 PSL-based Confidence Score Reasoning for Unseen facts

In order to better estimate confidence scores, both observed and unseen facts in knowledge graphs should be utilized. Deterministic knowledge graph embedding methods assume that all unseen facts are false beliefs, and use negative sampling to add some of these false relations into training. One major challenge of learning embeddings for uncertain knowledge graphs, however, is to properly estimate the uncertainty of unseen triples, as simply treating their confidence score as 0 can no longer capture the subtle uncertainty. For example, it is common that a Protein-Protein Interaction Network knowledge graph may have no interaction records for two proteins that can be potentially binded. Ignoring such possibility will result in information loss.

We thus introduce probabilistic soft logic (PSL) [48] to infer confidence scores for these unseen facts to further enhance the embedding performance. PSL is a framework for confidence reasoning that propagates confidence of existing knowledge to unseen triples using soft logic.

3.4.2.1 Probabilistic Soft Logic

A PSL program consists of a set of first order logic rules that describe logical dependencies between facts (atoms). One example of logical rule is shown below:

Example 3.4.1. ***A Logical Rule on Transitivity of Synonym Relation.***
$(\underline{A},\mathtt{synonym},\underline{B}) \wedge (\underline{B},\mathtt{synonym},\underline{C}) \rightarrow (\underline{A},\mathtt{synonym},\underline{C})$

This logical rule describes the transitivity of the relation `synonym`. In this logical rule, $\underline{A}$, $\underline{B}$ and $\underline{C}$ are placeholders for entities, `synonym` is the predicate that corresponds to the relation in uncertain knowledge graphs, $(\underline{A},\mathtt{synonym},\underline{B}) \wedge (\underline{B},\mathtt{synonym},\underline{C})$ is the body of the rule, and $(\underline{A},\mathtt{synonym},\underline{C})$ is the head of the rule.

A logical rule serves as a template rule. By replacing the placeholders in a logical rule with concrete entities and relations, we can get rule instances, which are called *ground rules*. Considering Example 3.4.1 and uncertain facts from Example 3.3.1, we can have the following ground rule by replacing the placeholders with real facts in knowledge graph.

Example 3.4.2. ***A Ground Rule on Transitivity of Synonym.***

(college, $\texttt{synonym}$, university) $\wedge$ (university, $\texttt{synonym}$, institute) $\rightarrow$ (college, $\texttt{synonym}$, institute)

Different from Boolean logic, PSL associates every atom, i.e., a triple l, with a *soft truth value* from the interval $[0, 1]$, which corresponds to the confidence score in our context and enables fuzzy reasoning. The assignment process of soft truth values is called an *interpretation*. We denote the soft truth value of an atom l assigned by the interpretation I as $I(l)$. Naturally, for observed facts, their observed confidence scores are used for assignment; and for unseen triples, the embedding-based estimated confidence scores will be assigned to them:

$$\begin{aligned} I(l) &= s_l, l \in \mathcal{L}^+ \\ I(l) &= f(l), l \in \mathcal{L}^- \end{aligned} \tag{3.5}$$

where $\mathcal{L}^+$ denotes the observed triple set, $\mathcal{L}^-$ denotes the unseen triples, s_l denotes the confidence score for observed triple l, and $f(l)$ denotes the embedding-based confidence score function for l.

In PSL, Lukasiewicz t-norm is used to define the basic logical operations, including logical conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$), as follows:

$$I(l_1 \wedge l_2) = \max\{0, I(l_1) + I(l_2) - 1\} \tag{3.6}$$

$$I(l_1 \vee l_2) = \min\{1, I(l_1) + I(l_2)\} \tag{3.7}$$

$$I(\neg l_1) = 1 - I(l_1) \tag{3.8}$$

For example, according to Eq. (3.6) and (3.7), $0.8 \wedge 0.3 = 0.1$ and $0.8 \vee 0.3 = 1$. For a rule $\gamma \equiv \gamma_{body} \rightarrow \gamma_{head}$, as it can be written as $\neg\gamma_{body} \vee \gamma_{head}$, its value p_γ can be computed as

$$p_{\gamma_{body} \rightarrow \gamma_{head}} = \min\{1, 1 - I(\gamma_{body}) + I(\gamma_{head})\} \tag{3.9}$$

PSL regards a rule γ as *satisfied* when the truth value of its head $I(\gamma_{head})$ is the same or higher than its body $I(\gamma_{body})$, i.e., when its value is greater than or equal to 1.

$$d_\gamma = 1 - p_\gamma = \max\{0, I(\gamma_{body}) - I(\gamma_{head})\} \tag{3.10}$$

Consider Example 3.4.2. Let (college, `synonym`, university) be l_1, (university, `synonym`, college) be l_2, and (college, `synonym`, institute) be l_3. Assuming that l_1 and l_2 are observed triples in knowledge graph, and l_3 is unseen, according to Equation (3.5), (3.6), and (3.9), the distance to satisfaction of this ground rule is calculated as below:

$$\begin{aligned} d_\gamma &= \max\{0, I(l_1 \wedge l_2) - I(l_3)\} \\ &= \max\{0, s_{l_1} + s_{l_2} - 1 - f(l_3)\} \\ &= \max\{0, 0.85 - f(l_3)\} \end{aligned}$$

where s_{l_1} and s_{l_2} are the ground truth confidence scores of corresponding facts in the uncertain knowledge graph.

This equation indicates that the ground rule in Example 3.4.2 is completely satisfied when $f(l_3)$, the estimated confidence score of (college, `synonym` institute), is above 0.85. When $f(l_3)$ is under 0.85, the smaller $f(l_3)$ is, the larger loss we have. In other words, a bigger confidence score is preferable. In the above example, we can see that the embedding-based confidence score for this unseen fact, $f(l_3)$, will affect the loss function, and it is desirable to learn embeddings that minimize these losses. Note that if we simply treat the unseen relation l_3 as false and use MSE (Mean Squared Error) as the loss, the loss would be $f(l_3)^2$, which is in favor of a lower confidence score mistakenly.

Moreover, we add a rule to penalize the predicted confidence scores of all unseen facts, which can be considered as a prior knowledge, i.e., any unseen fact has a low probability to be true. Formally, for an unseen fact $l = (h, r, t) \in \mathcal{L}^-$, we have a ground rule γ_0:

$$\gamma_0 : \neg l \tag{3.11}$$

According to Eq. (3.8) and (3.10), its distance to satisfaction d_{γ_0} is derived as:

$$d_{\gamma_0} = f(l) \tag{3.12}$$

3.4.3 Embedding Uncertain knowledge graphs

In this subsection, we present the objective function of uncertain knowledge graph embeddings.

3.4.3.1 Loss on observed facts

Let $\mathcal{L}^+$ be the set of observed facts, the goal is to minimize the mean squared error (MSE) between the ground truth confidence score s_l and our prediction $f(l)$ for each relation $l \in \mathcal{L}^+$:

$$\mathcal{J}^+ = \sum_{l \in \mathcal{L}^+} |f(l) - s_l|^2 \tag{3.13}$$

3.4.3.2 Loss on unseen facts

Let $\mathcal{L}^-$ be the sampled set of unseen relations, and Γ_l be the set of ground rules with l as the rule head, the goal is to minimize the distance to rule satisfaction for each triple l. In particular, we choose to use the square of the distance as the following loss [4]:

$$\mathcal{J}^- = \sum_{l \in \mathcal{L}^-} \sum_{\gamma \in \Gamma_l} |\psi_\gamma(f(l))|^2 \tag{3.14}$$

where $\psi_\gamma(f(l))$ denotes the weighted distance to satisfaction $w_\gamma d_\gamma$ of the rule γ as a function of $f(l)$ where w_γ is a hand-crafted weight for the rule γ.

Note that when l is only covered by $\gamma_0 : \neg l$, we have $\sum_{\gamma \in \Gamma_l} |\psi_\gamma(f(l))|^2 = |f(l)|^2$, which is essentially the MSE loss by treating unseen facts as false.

3.4.3.3 The Joint Objective Function

Combining Eq. (3.13) and (3.14), we obtain the following joint objective function:

$$\mathcal{J} = \sum_{l \in \mathcal{L}^+} |f(l) - s_l|^2 + \sum_{l \in \mathcal{L}^-} \sum_{\gamma \in \Gamma_l} |\psi_\gamma(f(l))|^2 \tag{3.15}$$

Similar to deterministic knowledge graph embedding algorithms, we sample unseen relations by corrupting the head and the tail for observed facts to generate $\mathcal{L}^-$ during training.

Dataset	#Ent.	#Rel.	#Rel. Facts	Avg(s)	Std(s)
CN15k	15,000	36	241,158	0.629	0.232
NL27k	27,221	404	175,412	0.797	0.242
PPI5k	4,999	7	271,666	0.415	0.213

Table 3.1: Statistics of the uncertain knowledge graph datasets. *Ent.* denotes entities and *Rel.* stands for relations. Avg(s) and Std(s) are the average and standard deviation of the confidence scores.

Dataset	Logical Rules	Hit Ratio
CN15k	($\underline{\text{A}}$, relatedTo, $\underline{\text{B}}$)∧($\underline{\text{B}}$, relatedTo, $\underline{\text{C}}$)→($\underline{\text{A}}$, relatedTo, $\underline{\text{C}}$)	37.0%
	($\underline{\text{A}}$, causes, $\underline{\text{B}}$)∧($\underline{\text{B}}$, causes, $\underline{\text{C}}$)→($\underline{\text{A}}$, causes, $\underline{\text{C}}$)	35.6%
NL27k	($\underline{\text{A}}$, atheletePlaysForTeam,$\underline{\text{B}}$) ∧ ($\underline{\text{A}}$, athletePlaysSport, $\underline{\text{C}}$) →($\underline{\text{B}}$, teamPlaysSport, $\underline{\text{C}}$)	42.9%
PPI5k	($\underline{\text{A}}$, binding, $\underline{\text{B}}$)∧($\underline{\text{B}}$, binding, $\underline{\text{C}}$)→($\underline{\text{A}}$, binding, $\underline{\text{C}}$)	80.8%

Table 3.2: Examples of logical rules. *Hit ratio* means the proportion of facts that have already existed in the knowledge graph

We give two model variants that differ in the choice of $f(l)$. We refer to the variant that adopts Equation (3.3) as $\texttt{UKGE}_{logi}$ and name the one using Equation (3.4) as $\texttt{UKGE}_{rect}$.

3.5 Experiments

In this section, we evaluate our models on three tasks: confidence prediction, fact ranking, and fact classification.

3.5.1 Datasets

The evaluation is conducted on three datasets named as CN15k, NL27k, and PPI5k, which are extracted from ConceptNet, NELL, and the Protein-Protein Interaction Knowledge Base STRING [89] respectively. CN15k matches the number of nodes with FB15k [9] - the widely used benchmark dataset for deterministic knowledge graph embeddings [9, 105, 91], while NL27k is a larger dataset. PPI5k is a denser graph with fewer entities but more facts than the other two. Table 3.1 gives the statistics of the datasets, and more details are introduced below.

3.5.1.1 CN15k

CN15k is a subgraph of the commonsense knowledge graph ConceptNet. This subgraph contains 15,000 entities and 241,158 uncertain facts in English. The original scores in ConceptNet vary from 0.1 to 22, where 99.6% are less than or equal to 3.0. For normalization, we first bound confidence scores to $x \in [0.1, 3.0]$, and then applied the min-max normalization on $\log x$ to map them into [0.1, 1.0].

3.5.1.2 NL27

NL27k is extracted from NELL [60], an uncertain knowledge graph obtained from webpage reading. NL27k contains 27,221 entities, 404 relations, and 175,412 uncertain facts. In the process of min-max normalization, we search for the lower boundary from 0.1 to 0.9. We have found out that normalizing the confidence score to interval $[0.1, 1]$ yields best results.

3.5.1.3 PPI5k

The Protein-Protein Interaction Knowledge Base STRING labels the interactions between proteins with the probabilities of occurrence. PPI5k is a subset of STRING that contains 271,666 uncertain facts for 4,999 proteins and 7 interactions.

In an uncertain knowledge graph, a fact is considered *strong* if its confidence score s_l is above a knowledge graph-specific threshold τ. Here we set $\tau = 0.85$ for both CN15k and NL27k. We follow the instructions from [89] and set $\tau = 0.70$ for PPI5k. Under this setting, 20.4% of facts in CN15k, 20.1% of those in NL27k, and 12.4% of those in PPI5k are considered strong.

3.5.2 Experimental Setup

We split each dataset into three parts: 85% for training, 7% for validation, and 8% for testing. To test if our model can correctly interpret negative links, we add the same amount of negative links as existing facts into the test sets.

We use Adam optimizer [49] for training, for which we set the exponential decay rates $\beta_1 = 0.9$ and $\beta_2 = 0.99$. We report results for all models respectively based on their best hyperparameter settings. For each model, the setting is identified based on the validation set performance. We select among the following sets of hyper-parameter values: learning rate $lr \in \{0.001, 0.005, 0.01\}$, dimensionality $k \in \{64, 128, 256, 512\}$, batch size $b \in \{128, 256, 512, 1024\}$, The L_2 regularization coefficient λ is fixed as 0.005. Training was stopped using early stopping based on MSE on the validation set, computed every 10 epochs. The best hyper-parameter combinations on CN15k and NL27k are $\{lr = 0.001, k = 128\}$ and $b = 128$ for $\texttt{UKGE}_{rect}$, $b = 512$ for $\texttt{UKGE}_{logi}$. On PPI5k they are $\{lr = 0.001, k = 128, b = 256\}$ for both variants.

3.5.3 Logical Rule Generation

Our model requires additional input as logical rules for PSL reasoning. We heuristically create candidate logical rules by considering length-2 paths (i.e., $(E_1,R_1,E_2) \wedge (E_2,R_2,E_3) \rightarrow (E_1,R_3,E_3)$) and validate them by *hit ratio,* i.e. the proportion of facts implied by the rule to be truly existent in the knowledge graph. The higher ratio implies that the rule is more convincing. When grounding out the logical rules, to guarantee the quality of the ground rules, we only adopt observed strong facts in our rule body. We eventually create 3 logical rules for CN15k, 4 for NL27k, and 1 for PPI5k. Table 3.2 gives some examples of the logical rules and their hit ratios. How to systematically create more promising logical rules will be considered as future work.

3.5.4 Baselines

Three types of baselines are considered in our comparison, which include (i) deterministic knowledge graph embedding models TransE [9], DistMult [105] and ComplEx [91], (ii) an uncertain graph embedding model URGE [40], and (iii) $\texttt{UKGE}_{n-}$ and $\texttt{UKGE}_{p-}$ that are two simplified versions of our model.

- Deterministic knowledge graph Embedding Models. TransE, DistMult, and ComplEx have demonstrated high performance on deterministic knowledge graphs. Only the high-confidence

Dataset	CN15k		NL27k		PPI5k	
Metrics	MSE	MAE	MSE	MAE	MSE	MAE
URGE	10.32	22.72	7.48	11.35	1.44	6.00
$\texttt{UKGE}_{n-}$	23.96	30.38	24.86	36.67	7.46	19.32
$\texttt{UKGE}_{p-}$	9.02	20.05	2.67	7.03	0.96	4.09
$\texttt{UKGE}_{rect}$	**8.61**	**19.90**	**2.36**	**6.90**	**0.95**	**3.79**
$\texttt{UKGE}_{logi}$	9.86	20.74	3.43	7.93	0.96	4.07

Table 3.3: Mean squared error (MSE) and mean absolute error (MAE) of fact confidence prediction ($\times 10^{-2}$).

facts from knowledge graphs are used for training. For each knowledge graph, we have a knowledge graph-specific confidence score threshold τ to distinguish the high-confidence facts from the low-confidence ones, which will be discussed later in Section 3.5.7. These models cannot predict confidence scores. We compare our methods to them only on the ranking and the classification tasks. For the same reason, the early stopping is based on mean reciprocal rank (MRR) on the validation set. We adopt the implementation given by [91] and choose the best hyper-parameters following the same grid search procedure. This implementation uses [28] for optimization. The best hyper-parameter combinations on CN15k and NL27k are $b = 1024$, $\{lr = 0.01, k = 128\}$ for TransE and $\{lr = 0.05, k = 256\}$ for DistMult and ComplEx. On PPI5k they are $lr = 0.1$, $\{k = 128, b = 512\}$ for DistMult and $\{k = 256, b = 1024\}$ for TransE and ComplEx.

- Uncertain Graph Embedding Model. URGE is proposed very recently to embed uncertain graphs. However, it cannot deal with multiple types of relations in knowledge graphs, and it only produces node embeddings. We simply ignore relation types when applying URGE to our datasets. We adopt its first-order proximity version as our tasks focus on the edge relations between nodes.

- Two Simplified Versions of Our Model. To justify the use of negative links and PSL reasoning in our model, we propose two simplified versions of $\texttt{UKGE}_{rect}$, called $\texttt{UKGE}_{n-}$ and $\texttt{UKGE}_{p-}$. In $\texttt{UKGE}_{n-}$, we only keep the observed facts and remove negative sampling, and in $\texttt{UKGE}_{p-}$, we remove PSL reasoning and use the MSE loss for unseen facts.

metrics	CN15K		NL27k		PPI5k	
Dataset	linear	exp.	linear	exp.	linear	exp.
TransE	0.601	0.591	0.730	0.722	0.710	0.700
DistMult	0.689	0.677	0.911	0.897	0.894	0.880
ComplEx	0.723	0.712	0.921	0.913	0.896	0.881
URGE	0.572	0.570	0.593	0.593	0.726	0.723
$\texttt{UKGE}_{n-}$	0.236	0.232	0.245	0.245	0.514	0.517
$\texttt{UKGE}_{p-}$	0.769	0.768	0.933	0.929	0.940	0.944
$\texttt{UKGE}_{rect}$	0.773	0.775	0.939	0.942	0.946	0.946
$\texttt{UKGE}_{logi}$	**0.789**	**0.788**	**0.955**	**0.956**	**0.970**	**0.969**

Table 3.4: Mean normalized DCG for global ranking task. Here *linear* stands for linear gain, and *exp.* stands for exponential gain.

3.5.5 Confidence Prediction

The objective of this task is to predict confidence scores of unseen facts.

3.5.5.1 Evaluation Protocol

For each uncertain fact (l, s_l) in the test set, we predict the confidence score of l and report the mean squared error (MSE) and mean absolute error (MAE).

3.5.5.2 Results

Results are reported in Table 3.3. Both our variants $\texttt{UKGE}_{rect}$ and $\texttt{UKGE}_{logi}$ outperform the baselines URGE, $\texttt{UKGE}_{n-}$, and $\texttt{UKGE}_{p-}$, since URGE only takes node proximity information and cannot model the rich relations between entities, and $\texttt{UKGE}_{n-}$ does not adopt negative sampling and cannot recognize negative links. The better results of $\texttt{UKGE}_{rect}$ than $\texttt{UKGE}_{p-}$ demonstrate that introducing PSL into embedding learning can enhance the model performance. Between the two model variants, $\texttt{UKGE}_{rect}$ results in smaller MSE and MAE than $\texttt{UKGE}_{logi}$. We notice that all the models achieve much smaller MSE on PPI5k than CN15k and NL27k. We hypothesize that this is because the much higher density of PPI5k facilitates embedding learning [66].

Dataset	head	relation	true tail	*conf.*	*pred.* tail	*pred. conf.*	true *conf.*
CN15k	rush	relatedto	fast	0.968	fast	0.703	0.968
			motion	0.709	move	0.623	0.557
			rapid	0.709	hour	0.603	0.654
			urgency	0.709	time	0.601	0.105
	hotel	usedfor	sleeping	1.0	relaxing	0.858	N/A
			rest	0.984	sleeping	0.849	1.0
			bed away from home	0.709	rest	0.827	0.984
			stay overnight	0.709	hotel room	0.797	N/A
NL27k	Toyota	competeswith	Honda	1.0	Honda	0.942	1.0
			Ford	1.0	Hyundai	0.910	0.719
			BMW	0.964	Chrysler	0.908	N/A
			General Motors	0.930	Nissan	0.896	0.859

Table 3.5: Examples of fact ranking (global) results using UKGE$_{logi}$. Top 4 results are shown. N/A denotes facts that are not observed in knowledge graph. *conf.* stands for *confidence* and *pred.* is the abbreviation for *predicted.*

3.5.6 Fact Ranking

The next task focuses on ranking tail entities in the right order for the query $(h, r, \underline{?t})$.

3.5.6.1 Evaluation Protocol

For a query $(h, r, \underline{?t})$, we rank all the entities in the vocabulary as tail candidates and evaluate the ranking performance using the normalized Discounted Cumulative G ain (nDCG) [55]. We define the gain in retrieving a relevant tail t_0 as the ground truth confidence score $s_{(h,r,t_0)}$. We take the mean nDCG over the test query set as our ranking metric. We report the two versions of nDCG that use linear gain and exponential gain respectively. The exponential gain version puts stronger emphasis on highly relevant results.

3.5.6.2 Results

Table 3.4 shows the mean nDCG over all test queries for all compared methods. Though TransE, DistMult, and ComplEx do not encode the confidence score information, they maximize the plausibility of all observed facts and therefore rank these existing facts high. We observe that Dist-

Mult and ComplEx have considerably better performance than TransE, as TransE does not handle `1-to-N` relations well. ComplEx embeds entities and relations in the complex domain and handles asymmetric relations better than DistMult. It achieves the best results among the deterministic knowledge graph embedding models on this task. As $\texttt{UKGE}_{n-}$ removes negative sampling from the loss function, it cannot distinguish the negative links from existing facts and results in the worst performance. $\texttt{UKGE}_{p-}$ yields slightly worse performance than $\texttt{UKGE}_{rect}$. Besides ranking the existing facts highly, our models also preserve the order of the observed facts and thus achieve higher nDCG scores. Both $\texttt{UKGE}_{rect}$ and $\texttt{UKGE}_{logi}$ outperform all the baselines under all settings, while $\texttt{UKGE}_{logi}$ yields higher nDCG on all three datasets than $\texttt{UKGE}_{rect}$. Considering the confidence prediction results of $\texttt{UKGE}_{logi}$ in Section 3.5.5, we hypothesize that the easy saturation of logistic function allows $\texttt{UKGE}_{logi}$ to better distinguish negative links from true facts, while this feature compromises its ability to fit confidence scores more precisely.

3.5.6.3 Case study

Table 3.5 gives some examples of fact ranking results by $\texttt{UKGE}_{logi}$. Given a query $(h, r, \underline{?t})$, the top 4 predicted tails and true tails are given, sorted by their scores in descending order. The predictions are consistent with our common-sense. It is worth noting that some quite reasonable unseen facts such as *hotel is used for relaxing*, can be predicted correctly. In other words, our proposed approach can be potentially used to infer new knowledge from the observed ones with reasonable confidence scores, which may shed light on another line of future study.

3.5.7 Fact Classification

This last task is a binary classification task to decide whether a given fact l is a *strong* fact or not. A fact is considered *strong* if its confidence score s_l is above a knowledge graph-specific threshold τ. The embedding models need to distinguish facts in the knowledge graph from negative links and high-confidence facts from low-confidence ones.

Metrics	CN15k		NL27k		PPI5k	
Dataset	F-1	Accu.	F-1	Accu.	F-1	Accu.
TransE	23.4	67.9	65.1	53.4	83.2	98.5
DistMult	27.9	71.1	72.1	70.1	86.9	97.1
ComplEx	18.9	73.2	63.3	53.4	83.2	98.9
URGE	21.2	86.0	83.6	88.7	85.2	98.6
UKGE$_{n-}$	23.6	86.1	64.4	65.5	92.7	99.3
UKGE$_{p-}$	26.2	88.7	89.7	93.4	94.2	99.3
UKGE$_{rect}$	**28.8**	**90.4**	**92.3**	**95.2**	**95.1**	99.4
UKGE$_{logi}$	25.9	90.1	88.4	93.0	94.5	**99.5**

Table 3.6: F-1 scores (%) and accuracies (%) of fact classification

3.5.7.1 Evaluation Protocol

We follow a procedure that is similar to [98]. Our test set consists of facts from the knowledge graph and randomly sampled negative links equally. We divide the test cases into two groups, strong and weak/false, by their ground truth confidence scores. A test fact l is strong when l is in the knowledge graph and $s_l > \tau$, otherwise weak/false. We fit a logistic regression classifier as a downstream classifier on the predicted confidence scores.

3.5.7.2 Results

F-1 scores and accuracies are reported in Table 3.6. These results show that our two model variants consistently outperform all baseline models. The deterministic knowledge graph models can distinguish the existing facts from negative links, but they do not leverage the confidence information and cannot recognize the high-confidence ones. URGE does not encode the rich relations. Although UKGE$_{n-}$ fits confidence scores in the knowledge graph, it cannot correctly interpret negative links as false. Consistent with the previous two tasks, the performance of UKGE$_{p-}$ is worse than UKGE$_{rect}$.

3.6 Conclusion

This chapter introduces the first work that generalizes knowledge graph embeddings to the uncertainty scenario. Our model UKGE learns embeddings according to fact confidence score, and effectively preserves both the facts and uncertainty information in the embedding space of KG. To further enhance accuracy, we introduce probabilistic soft logic to infer confidence scores to provide extra supervision during training. We propose two variants of UKGE based on different regression functions. Experiments are conducted on three real-world uncertain knowledge graphs via three tasks, i.e. confidence prediction, fact ranking, and fact classification. UKGE shows effectiveness in capturing uncertain knowledge by achieving promising results, and it consistently outperforms baselines on these tasks.

CHAPTER 4

Box Embeddings for Uncertain Knowledge Graph Reasoning

In this chapter, we propose to embed uncertain knowledge graphs with box embeddings for query answering.

4.1 Introduction

Chapter 3 introduces the first uncertain knowledge graph embedding model UKGE, which models triple-level uncertainty and has limitations regarding enforcing logical reasoning rules. Particularly, UKGE models the triple plausibility in the form of embedding product [105] and trains the embedding model as a regressor to predict the confidence score. One interpretation of the model is that it models each triple using a binary random variable, where the latent dependency structure between different binary random variables is captured by vector similarities. Without an explicit dependency structure it is difficult to enforce logical reasoning rules to maintain global consistency.

In this chapter, in order go beyond triple-level uncertainty modeling, we consider each entity as a binary random variable. However, representing such a probability distribution in an embedding space and reasoning over it is non-trivial. It is difficult to model marginal and joint probabilities for entities using simple geometric objects like vectors. In order to encode probability distributions in the embedding space, recent works [52, 94, 56, 24] represent random variables as more complex geometric objects, such as cones and axis-aligned hyperrectangles (*boxes*), and use *volume* as the probability measure. Inspired by such advances of probability measures in embeddings, we present BEUrRE 🧈 (**B**ox **E**mbedding for **U**nce**r**tain **RE**lational Data)[1]. BEUrRE represents entities as

[1] "Beurre" is French for "butter".

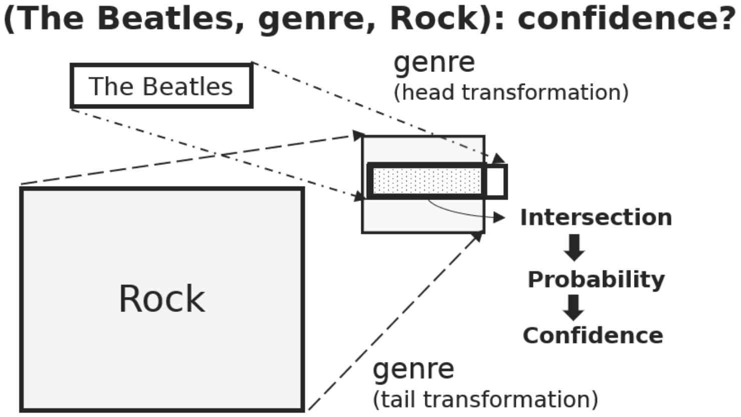

Figure 4.1: BEUrRE models entities as boxes and relations as two affine transforms.

boxes. Relations are modeled as two separate affine transforms on the head and tail entity boxes. Confidence of a triple is modeled by the intersection between the two transformed boxes. Fig. 4.1 shows how a fact about the genre of the Beatles is represented under our framework.

Such representation is not only inline with the human perception that entities or concepts have different levels of granularity, but also allows more powerful domain knowledge representation. UKGE has demonstrated that introducing domain knowledge about relation properties (e.g. transitivity) can effectively enhance reasoning on uncertain knowledge graphs. While UKGE uses Probabilistic Soft Logic (PSL) [5] to reason and add the extra training samples to training, PSL has a limited scope of application when an uncertain knowledge graph is sparse. In this chapter, we propose sufficient conditions for these relation properties to be preserved in the embedding space and directly model the relation properties by regularizing relation-specific transforms. This technique is more robust to noise and has wide coverage that is not restricted by the scarcity of existing triples. Extensive experiments on two benchmark datasets show that BEUrRE effectively captures the uncertainty information and consistently outperforms the baseline models on fact confidence prediction and fact ranking.

4.2 Related Work

Developing embedding methods to represent elements using geometric objects with more complex structures than (Euclidean) vectors is an active area of study. *Poincaré embeddings* [63] represent entities in hyperbolic space, leveraging the inductive bias of negative curvature to fit hierarchies. *Order embeddings* [93] take a region-based approach, representing nodes of a graph using infinite cones, and using containment between cones to represent edges. *Hyperbolic entailment cones* [33] combine order embeddings with hyperbolic geometry. While these methods show various degrees of promise when embedding hierarchies, they do not provide scores between entities that can be interpreted probabilistically, which is particularly useful in our setting.

[52] extend order embeddings with a probabilistic interpretation by integrating the volume of the infinite cones under the negative exponential measure, however the rigid structure imposed by the cone representation limits the representational capacity, and the resulting model cannot model negative correlation or disjointness. Introduced by [94], *probabilistic box embeddings* represent elements using axis-aligned hyperrectangles (or *boxes*). Box embeddings not only demonstrate improved performance on modeling hierarchies, such embeddings also capture probabilistic semantics based on box volumes, and are capable of compactly representing conditional probability distributions. A few training improvement methods for box embeddings have been proposed [56, 24], and we make use of the latter, which is termed *GumbelBox* after the distribution used to model endpoints of boxes.

While box embeddings have shown promise in representing hierarchies, our work is the first use of box embeddings to represent entities in multi-relational data. *Query2Box* [70] and *BoxE* [1] make use of boxes in the loss function of their models, however entities themselves are represented as vectors, and thus these models do not benefit from the probabilistic semantics of box embeddings, which we rely on heavily for modeling uncertain knowledge graphs. In [64], the authors demonstrate the capability of box embeddings to jointly model two hierarchical relations, which is improved upon using a learned transform in [25]. Similarly to [70] and [25], we also make use of a learned transform for each relation, however we differ from [70] in that entities themselves are boxes, and differ from both in the structure of the learned transform.

4.3 Preliminaries

Before we move on to the proposed method in this chapter, we use this section to introduce the background of box embeddings.

4.3.1 Probabilistic Box Embeddings

In this section we give a formal definition of probabilistic box embeddings, as introduced by [94]. A *box* is an n-dimensional hyperrectangle, i.e. a product of intervals

$$\prod_{i=1}^{d}[x_i^{\mathrm{m}}, x_i^{\mathrm{M}}], \quad \text{where} \quad x_i^{\mathrm{m}} < x_i^{\mathrm{M}}.$$

Given a space $\Omega_{\text{Box}} \subseteq \mathbb{R}^n$, we define $\mathcal{B}(\Omega_{\text{Box}})$ to be the set of all boxes in Ω_{Box}. Note that $\mathcal{B}(\Omega_{\text{Box}})$ is closed under intersection, and the volume of a box is simply the product of side-lengths. [94] note that this allows one to interpret box volumes as unnormalized probabilities. This can be formalized as follows.

Definition 4. *Let $(\Omega_{Box}, \mathcal{E}, P_{Box})$ be a probability space, where $\Omega_{Box} \subseteq \mathbb{R}^n$ and $\mathcal{B}(\Omega_{Box}) \subseteq \mathcal{E}$. Let $\mathcal{Y}$ be the set of binary random variables Y on Ω_{Box} such that $Y^{-1}(1) \in \mathcal{B}(\Omega_{Box})$. A* probabilistic box embedding *of a set S is a function : $S \to \mathcal{Y}$. We typically denote $f(s) =: Y_s$ and $Y_s^{-1}(1) =:$* $\text{Box}(s)$.

Essentially, to each element of S we associate a box which, when taken as the support set of a binary random variable, allows us to interpret each element of S as a binary random variable. Using boxes for the support sets allows one to easily calculate marginal and conditional probabilities, for example if we embed the elements $\{\textsc{cat}, \textsc{mammal}\}$ as boxes in $\Omega_{\text{Box}} = [0,1]^d$ with P_{Box} as Lebesgue measure, then

$$\begin{aligned} P(\textsc{mammal} \mid \textsc{cat}) &= P_{\text{Box}}(X_{\textsc{mammal}} | X_{\textsc{cat}}) \\ &= \frac{\text{Vol}(\text{Box}(\textsc{mammal}) \cap \text{Box}(\textsc{cat}))}{\text{Vol}(\text{Box}(\textsc{cat}))}. \end{aligned}$$

4.3.2 Gumbel Boxes

We further give a brief description of the *GumbelBox* method, which we rely on for training our box embeddings [24].

As described thus far, probabilistic box embeddings would struggle to train via gradient descent, as there are many settings of parameters and objectives which have no gradient signal. (For example, if boxes are disjoint but should overlap.) To mitigate this, [24] propose a latent noise model, where the min and max coordinates of boxes in each dimension are modeled via Gumbel distributions, that is

$$\begin{aligned} \mathrm{Box}(X) &= \prod_{i=1}^{d} [x_i^{\mathrm{m}}, x_i^{\mathrm{M}}] \quad \text{where} \\ x_i^{\mathrm{m}} &\sim \mathrm{GumbelMax}(\mu_i^{\mathrm{m}}, \beta), \\ x_i^{\mathrm{M}} &\sim \mathrm{GumbelMin}(\mu_i^{\mathrm{M}}, \beta). \end{aligned}$$

μ_i^{m} thereof is the *location* parameter, and β is the (global) variance. The Gumbel distribution was chosen due to its min/max stability, which means that the set of all "Gumbel boxes" are closed under intersection. [24] go on to provide an approximation of the expected volume of a Gumbel box,

$$\begin{aligned} &\mathbb{E}\left[\mathrm{Vol}(\mathrm{Box}(X))\right] \approx \\ &\qquad \prod_{i=1}^{d} \beta \log\left(1 + \exp\left(\tfrac{\mu_i^{\mathrm{M}} - \mu_i^{\mathrm{m}}}{\beta} - 2\gamma\right)\right). \end{aligned}$$

A first-order Taylor series approximation yields

$$\mathbb{E}[P_{\mathrm{Box}}(X_{\mathrm{A}} \mid X_{\mathrm{B}})] \approx \frac{\mathbb{E}[\mathrm{Vol}(\mathrm{Box}(A) \cap \mathrm{Box}(B))]}{\mathbb{E}[\mathrm{Vol}(\mathrm{Box}(B))]},$$

and [24] empirically demonstrate that this approach leads to improved learning when targeting a given conditional probability distribution as the latent noise essentially ensembles over a large collection of boxes which allows the model to escape plateaus in the loss function. We therefore use this method when training box embeddings.

Remark 4.3.1. *While we use Gumbel boxes for training, intuition is often gained by interpreting these boxes as standard hyperrectangles, which is valid as the Gumbel boxes can be seen as a distribution over such rectangles, with the Gumbel variance parameter β acting as a global measure of uncertainty. We thus make statements such as $\mathrm{Box}(X) \subseteq \mathrm{Box}(Y)$, which, strictly speaking, are not well-defined for Gumbel boxes. However we can interpret this probabilistically as $P(Y \mid X) = 1$ which coincides with the conventional interpretation when $\beta = 0$.*

4.4 Modeling

In this section, we present our uncertain knowledge graph embedding model `BEUrRE`. The proposed model encodes entities as probabilistic boxes and relations as affine transforms. We also discuss how this method incorporates logical constraints into learning.

4.4.1 Modeling Uncertain Knowledge Graphs with Box Embeddings

`BEUrRE` represents entities as Gumbel boxes, and a relation r acting on these boxes by translation and scaling. Specifically, we parametrize a Gumbel box $\mathrm{Box}(X)$ using a center $\mathrm{cen}(\mathrm{Box}(X)) \in \mathbb{R}^d$ and offset $\mathrm{off}(\mathrm{Box}(X)) \in \mathbb{R}^d_+$, where the location parameters are given by

$$\begin{aligned}
\mu_i^{\mathrm{m}} &= \mathrm{cen}(\mathrm{Box}(X)) - \mathrm{off}(\mathrm{Box}(X)), \\
\mu_i^{\mathrm{M}} &= \mathrm{cen}(\mathrm{Box}(X)) + \mathrm{off}(\mathrm{Box}(X)).
\end{aligned}$$

We consider transformations on Gumbel boxes parametrized by a translation vector $\tau \in \mathbb{R}^d$ and a scaling vector $\Delta \in \mathbb{R}^d_+$ such that

$$\begin{aligned}
\mathrm{cen}(f(\mathrm{Box}(X); \tau, \Delta)) &= \mathrm{cen}(\mathrm{Box}(X)) + \tau, \\
\mathrm{off}(f(\mathrm{Box}(X); \tau, \Delta)) &= \mathrm{off}(\mathrm{Box}(X)) \circ \Delta,
\end{aligned}$$

where $\circ$ is the Hadamard product. We use separate actions for the head and tail entities of a relation, which we denote f_r and g_r, and omit the explicit dependence on the learned parameters τ and Δ.

Remark 4.4.1. *Note that these relations are not an affine transformations of the* space*, Ω_{Box}, rather they perform a transformation of a* box*. These functions form an Abelian group under composition, and furthermore define a transitive, faithful group action on the set of (Gumbel) boxes.*

Given a triple (h, r, t), BEUrRE models the confidence score using the (approximate) conditional probability given by

$$\phi(h, r, t) = \frac{\mathbb{E}[\mathrm{Vol}(f_r(\mathrm{Box}(h)) \cap g_r(\mathrm{Box}(t)))]}{\mathbb{E}[\mathrm{Vol}(g_r(\mathrm{Box}(t)))]}.$$

We can think of the box $f_r(\mathrm{Box}(h))$ as the support set of a binary random variable representing the concept h in the context of the head position of relation r, for example $\mathrm{Box}(\textsc{TheBeatles})$ is a latent representation of the concept of The Beatles, and $f_{\textsc{genre}}(\mathrm{Box}(\textsc{TheBeatles}))$ represents The Beatles in the context of genre classification as the object to be classified.

4.4.2 Logical Constraints

The sparsity of real-world uncertain knowledge graphs makes learning high quality representations difficult. To address this problem, previous work [19] introduces domain knowledge about the properties of relations (e.g., transitivity) and uses PSL over first-order logical rules to reason for unseen facts and create extra training samples. While this technique successfully enhances the performance by incorporating constraints based on relational properties, the coverage of such reasoning is still limited by the density of the graph. In UKGE, the confidence score of a triple can be inferred and benefit training only if all triples in the rule premise are already present in the knowledge graph. This leads to a limited scope of application, particularly when the graph is sparse.

In our work, we propose sufficient conditions for these relation properties to be preserved in the embedding space and directly incorporating the relational constraints by regularizing relation-specific transforms. Compared to previous work, our approach is more robust to noise since it does

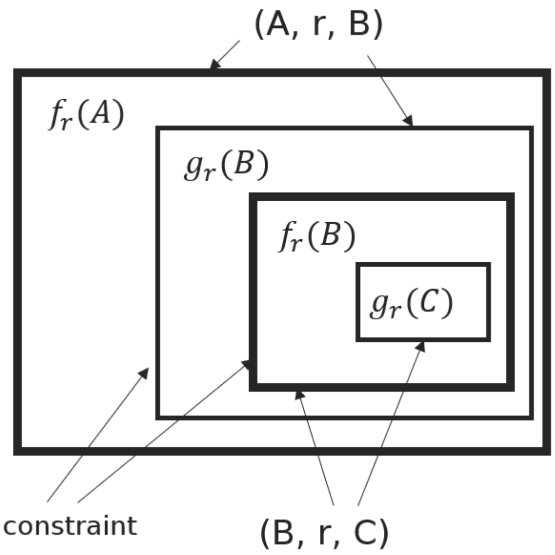

Figure 4.2: Illustration of how the constraint that $g_r(u)$ contains $f_r(u)$ preserves transitivity of relation r in the embedding space. A triple (h, r, t) is true if and only if $f_r(\text{Box}(h))$ contains $g_r(\text{Box}(t)))$. By adding this constraint, $f_r(\text{Box}(A))$ is guaranteed to contain $g_r(\text{Box}(C))$ if (A, r, B) and (B, r, C) are true.

not hardcode inferred confidence for unseen triples, and it has wide coverage that is not restricted by the scarcity of the existing triples.

In the following, we discuss the incorporation of two logical constraints — transitivity and composition — in the learning process. We use capital letters A, B, C to represent universally quantified entities from uncertain knowledge graph and use Φ to denote a set of boxes sampled from $\mathcal{B}(\Omega_{\text{Box}})$.

4.4.2.1 Transitivity Constraint

A relation r is *transitive* if $(A, r, B) \wedge (B, r, C) \implies (A, r, C)$. An example of a transitive relation is *hypernymy*.

The objective of imposing a transitivity constraint in learning is to preserve this property of the relation in the embedding space, i.e. to ensure that (A, r, C) will be predicted true if (A, r, B) and (B, r, C) are true. This objective is fulfilled if $g_r(\text{Box}(B))$ contains $f_r(\text{Box}(B))$. An illustration of the box containment relationships is given in Fig 4.2. Thus, we constrain f_r and g_r so that $g_r(u)$

contains $f_r(u)$ for any $u \in \Omega_{\text{Box}}$. We impose the constraint with the following regularization term:

$$L_{\text{tr}}(r) = \frac{1}{|\Phi|} \sum_{u \in \Phi} \| P_{\text{Box}}(g_r(u) \mid f_r(u)) - 1 \|^2 .$$

4.4.2.2 Composition Constraint

A relation r_3 is *composed of* relation r_1 and relation r_2 if $\quad (A, r_1, B) \wedge (B, r_2, C) \implies (A, r_3, C)$. For example, the relation *atheletePlaysSports* can be composed of relations *atheletePlaysForTeam* and *teamPlaysSports*.

To preserve the relation composition in the embedding space, we constrain that the relation-specific mappings f_{r_3} and g_{r_3} are the *composite mappings* of f_{r_1}, f_{r_2} and g_{r_1}, g_{r_2} respectively:

$$\begin{aligned} f_{r_3} &= f_{r_2} \cdot f_{r_1} \\ g_{r_3} &= g_{r_2} \cdot g_{r_1} \end{aligned}$$

where $\cdot$ is the mapping composition operator. Thus, for any $u \in \Omega_{\text{Box}}$, we expect that $f_{r_3}(u)$ is the same as $f_{r_2}(f_{r_1}(u))$ and $g_{r_3}(u)$ is the same as $g_{r_2}(g_{r_1}(u))$. We accordingly add the following regularization term

$$\begin{aligned} L_{\text{c}}(r_1, r_2, r_3) = \frac{1}{|\Phi|} \sum_{u \in \Phi} & f_{r_3}(u) \oplus f_{r_2}(f_{r_1}(u)) \\ & + g_{r_3}(u) \oplus g_{r_2}(g_{r_1}(u)) \end{aligned}$$

where $\oplus$ is defined as

$$\begin{aligned} \text{Box}_1 \oplus \text{Box}_2 = & \| 1 - P_{\text{Box}}(\text{Box}_1 \mid \text{Box}_2) \|^2 \\ & + \| 1 - P_{\text{Box}}(\text{Box}_2 \mid \text{Box}_1) \|^2 . \end{aligned}$$

4.4.3 Learning Objective

The learning process of BEUrRE optimizes two objectives. The main objective optimizes the loss for a regression task and, simultaneously, a constrained regularization loss enforces the aforementioned constraints.

Let $\mathcal{L}^+$ be the set of observed facts in training data. The goal is to minimize the mean squared error (MSE) between the ground truth confidence score s_l and the prediction $\phi(l)$ for each relation $l \in \mathcal{L}^+$. Following UKGE [19], we also penalize the predicted confidence scores of facts that are not observed in UKG. The main learning objective is as follows:

$$\mathcal{J}_1 = \sum_{l \in \mathcal{L}^+} |\phi(l) - s_l|^2 + \alpha \sum_{l \in \mathcal{L}^-} |\phi(l)|^2.$$

where $\mathcal{L}^-$ is a sample set of the facts not observed in UKG, and α is a hyper-parameter to weigh unobserved fact confidence penalization. Similar to previous works, we sample those facts by corrupting the head and the tail for observed facts to generate $\mathcal{L}^-$ during training.

In terms of constraints, let $\mathcal{R}_{\text{tr}}$ be the set of transitive relations, $\mathcal{R}_{\text{c}}$ be the set of composite relation groups, and w_{tr} and w_{c} be the regularization coefficients. We add the following regularization to impose our constraints on relations:

$$\mathcal{J}_2 = w_{\text{tr}} \sum_{r \in \mathcal{R}_{\text{tr}}} L_{\text{tr}}(r) + w_{\text{c}} \sum_{(r_1, r_2, r_3) \in \mathcal{R}_{\text{c}}} L_{\text{c}}(r_1, r_2, r_3).$$

Combining both learning objectives, the learning process optimizes the joint loss $J = J_1 + J_2$.

4.4.4 Inference

Once BEUrRE is trained, the model can easily infer the confidence of a new fact (h, r, t) based on the confidence score function $\phi(h, r, t)$ defined in Section 4.4.1. This inference mechanism easily supports other types of reasoning tasks, such as inferring the plausibility of a new fact, and ranking multiple related facts. The experiments presented in the next section will demonstrate the ability of BEUrRE to perform those reasoning tasks.

Dataset	#Ent.	#Rel.	#Rel. Facts	Avg(s)	Std(s)
CN15k	15,000	36	241,158	0.629	0.232
NL27k	27,221	404	175,412	0.797	0.242

Table 4.1: Statistics of the datasets. *Ent.* and *Rel.* stand for entities and relations. Avg(s) and Std(s) are the average and standard deviation of confidence.

4.5 Experiments

In this section we present evaluation of our model on two uncertain knowledge graph reasoning tasks, i.e. confidence prediction and fact ranking. More experimentation details are in Appendices.

4.5.1 Experiment Settings

4.5.1.1 Datasets

We follow [19] and evaluate our models on CN15k and NL27k benchmarks, which are subsets of ConceptNet [79] and NELL [60] respectively. Table 4.1 gives the statistics of the datasets. We use the same split provided by [19]: 85% for training, 7% for validation, and 8% for testing. We exclude the dataset PPI5k, the subgraph of the protein-protein interaction (PPI) network STRING [89], where the supporting scores of PPI information are indicators based on experimental and literary verification, instead of a probabilistic measure.

4.5.1.2 Logical Constraints

We report results of both versions of our model with and without logical constraints, denoted as `BEUrRE` (rule+) and `BEUrRE` respectively. For a fair comparison, we incorporate into `BEUrRE` (rule+) the same set of logical constraints as UKGE [19]. Table 4.2 gives a few examples of the relations on which we impose constraints.

Dataset	Transitivity	Composition
CN15k	causes	N/A
NL27k	locationAtLocation	(atheletePlaysForTeam, teamPlaysSport) → atheletePlaysSport

Table 4.2: Examples of relations with logical constraints.

4.5.1.3 Baselines

We compare our models with uncertain knowledge graph embedding models as well as deterministic knowledge graph embedding models.

UKG embedding models include UKGE [19] and URGE [40]. While UKGE has multiple versions incorporated with different regression functions, we report the results of the best performing one with the logistic function. We also include results for both settings with and without constraints, marked as UKGE (rule+) and UKGE in result tables respectively. URGE was originally designed for probabilistic homogeneous graphs and cannot handle multi-relational graphs, so accordingly we ignore relation information when embedding a UKG. UOknowledge graphE [10] cannot serve as a baseline because it requires additional ontology information for entities that is not available to these UKGs.

Deterministic knowledge graph embedding models TransE [9], DistMult [105], ComplEx [91], RotatE [88], and TuckER [7] have demonstrated high performance on reasoning tasks for deterministic knowledge graphs, and we also include them as baselines. These models cannot predict confidence scores for uncertain facts, so we compare our method with them only on the ranking task. Following [19], we only use facts with confidence above the threshold $\tau = 0.85$ to train deterministic models.

4.5.1.4 Model Configurations

We use Adam [49] as the optimizer and fine-tune the following hyper-parameters by grid search based on the performance on the validation set, i.e. MSE for confidence prediction and normalized Discounted Cumulative Gain (nDCG) for fact ranking. Training terminates with early stopping based on the same metric with a patience of 30 epochs. We repeat each experiment five times and report the average results. We search for the best hyper-parameter combination in the following space: Learning rate lr in {0.001, 0.0001, 0.00001}, Embedding dimension d in {30, 64, 128, 300}, Batch size b in {256, 512, 1024, 2048, 4096}, Gumbel box temperature in β {0.1, 0.01, 0.001, 0.0001}, L_2 in regularization λ {0.001, 0.01, 0.1, 1}. We performed grid search to choose the final setting.

The best hyper-parameter combinations for confidence prediction are $\{lr = 0.0001, b = 1024, d = 64, \beta = 0.01\}$, $b = 2048$ for CN15k and $b = 4096$ for NL27k. L_2 regularization is 1 for box sizes in logarithm scale and 0.001 for other parameters. For fact ranking they are $\{lr = 0.0001, d = 300, b = 4096, \lambda = 0.00001\}$, $\beta = 0.001$ for CN15k and $\beta = 0.0001$ for NL27k. The number of negative samples is fixed as 30. Rule weights are empirically set as $w_{tr} = w_{cp} = 0.1$.

We conduct our experiments on CPU Intel® Xeon® E5-2650 v4 12-core and a single GPU NVIDIA® GP102 TITAN Xp (12GB). RAM is 256GB. On this machine, training `BEUrRE` for the confidence prediction task takes around 1-1.5 hours. Training `BEUrRE` for the ranking task takes around 1-2 hours for CN15k and 3 hours for NL27k. For the reported model, on CN15k, `BEUrRE` has around 2M parameters for confidence prediction and 9M parameters for ranking. On NL27k, `BEUrRE` has 9M parameters for confidence prediction and 17M for ranking.

4.5.2 Confidence Prediction

This task seeks to predict the confidence of new facts that are unseen to training. For each uncertain fact (l, s_l) in the test set, we predict the confidence of l and report the mean squared error (MSE) and mean absolute error (MAE).

4.5.2.1 Results

Results are reported in Table 4.3. We compare our models with baselines under the unconstrained and logically constrained (marked with *rule+*) settings respectively. Under both settings,

Dataset	CN15k		NL27k	
Metrics	MSE	MAE	MSE	MAE
URGE	10.32	22.72	7.48	11.35
UKGE	9.02	20.05	2.67	7.03
BEUrRE	7.80	20.03	2.37	7.12
UKGE(rule+)	8.61	19.90	2.36	6.90
BEUrRE(rule+)	**7.49**	**19.88**	**2.01**	**6.89**

Table 4.3: Results of fact confidence prediction ($\times 10^{-2}$).

BEUrRE outperforms the baselines in terms of MSE on both datasets.

Under the unconstrained setting, BEUrRE improves MSE of the best baseline UKGE by 0.012 (ca. 14% relative improvement) on CN15k and 0.003 (ca. 11% relative improvement) on NL27k. The enhancement demonstrates that box embeddings can effectively improve reasoning on UKGs. It is worth noting that even without constraints in learning, BEUrRE can still achieve comparable MSE and MAE to the logically constrained UKGE (rule+) on both datasets and even outperforms UKGE (rule+) on CN15k. Considering that constraints of relations in CN15k mainly describe transitivity, the aforementioned observation is consistent with the fact that box embeddings are naturally good at capturing transitive relations, as shown in the recent study [94].

With logical constraints, BEUrRE (rule+) further enhances the performance of BEUrRE and reduces its MSE by 0.0031 (ca. 4% relative improvement) on CN15k and 0.0036 (ca. 15% relative improvement) on NL27k. This is as expected, since logical constraints capture higher-order relations of facts and lead to more globally consistent reasoning. We also observe that BEUrRE (rule+) brings larger gains over BEUrRE on NL27k, where we have both transitivity constraints and composition constraints, than on CN15k with only transitivity constraints incorporated.

In general, with box embeddings, BEUrRE effectively improves reasoning on uncertain knowledge graphs with better captured fact-wise confidence. Furthermore, the results under the logically constrained setting show the effectiveness of improving reasoning with higher-order relations of uncertain facts.

Variants	uncons.	rule+
Metrics	MSE ($\times 10^{-2}$)	
BEUrRE	7.80	7.49
—w/o Gumbel distribution	8.13	8.14
—Single relation-specific transform	7.81	7.60

Table 4.4: Ablation study results on CN15k. *uncons.* represents the unconstrained setting, and *rule+* denotes the logically constrained setting.

4.5.2.2 Ablation Study

To examine the contribution from Gumbel distribution to model box boundaries and the effectiveness of representing relations as two separate transforms for head and tail boxes, we conduct an ablation study based on CN15k. The results for comparison are given in Table 4.4. First, we resort to a new configuration of BEUrRE where we use smoothed boundaries for boxes as in [56] instead of Gumbel boxes. We refer to boxes of this kind as soft boxes. Under the unconstrained setting, using soft boxes increases MSE by 0.0033 on CN15k (ca. 4% relative degradation), with even worse performance observed when adding logical constraints. This confirms the finding by [24] that using Gumbel distribution for boundaries greatly improves box embedding training. Next, to analyze the effect of using separate transforms to represent a relation, we set the tail transform g_r as the identity function. For logical constraint incorporation, we accordingly update the constraint on transitive relation r as $P_{\text{Box}}(u \mid f_r(u)) = 1, u \in \Omega_{\text{Box}}$, which requires that u always contains $f_r(u)$, i.e. the translation vector of f_r is always zero and elements of the scaling vector are always less than 1. Although there is little difference between using one or two transforms under the unconstrained setting, under the logically constrained setting, the constraint is too stringent to be preserved with only one transform.

4.5.2.3 Case Study

To investigate whether our model can encode meaningful probabilistic semantics, we present a case study about box volumes. We examine the objects of the *atLocation* predicate on CN15k and check which entity boxes have larger volume and cover more entity boxes after the relation transformation. Ideally, geographic entities with larger areas or more frequent mentions should be at the top of the list. When using the `BEUrRE`(rule+) model, the top 10 in all entities are *place, town, bed, school, city, home, house, capital, church, camp*, which are general concepts. Among the observed objects of the *atLocation* predicate, the entities that have the least coverage are *Tunisia, Morocco, Algeria, Westminster, Veracruz, Buenos Aires, Emilia-Romagna, Tyrrhenian sea, Kuwait, Serbia*. Those entities are very specific locations. This observation confirms that the box volume effectively represents probabilistic semantics and captures specificity/granularity of concepts, which we believe to be a reason for the performance improvement.

4.5.3 Fact Ranking

Multiple facts can be associated with the same entity. However, those relevant facts may appear with very different plausibility. Consider the example about Honda Motor Co. in Section 4.1, where it was mentioned that *(Honda, competeswith, Toyota)* should have a higher belief than *(Honda, competeswith, Chrysler)*. Following this intuition, this task focuses on ranking multiple candidate tail entities for a query $(h, r, \underline{?t})$ in terms of their confidence.

4.5.3.1 Evaluation Protocol

Given a query $(h, r, \underline{?t})$, we rank all the entities in the vocabulary as tail entity candidates and evaluate the ranking performance using the normalized Discounted Cumulative Gain (nDCG) [55]. The gain in retrieving a relevant tail t_0 is defined as the ground truth confidence $s_{(h,r,t_0)}$. Same as [19], we report two versions of nDCG that use linear gain and exponential gain respectively. The exponential gain puts stronger emphasis on the most relevant results.

4.5.3.2 Results

We report the mean nDCG over the test query set in Table 4.5. Although the deterministic models do not explicitly capture the confidence of facts, those models are trained with high-confidence facts and have a certain ability to differentiate high confidence facts from lesser ones. URGE

Dataset	CN15K		NL27k	
Metrics	linear	exp.	linear	exp.
TransE	0.601	0.591	0.730	0.722
DistMult	0.689	0.677	0.911	0.897
ComplEx	0.723	0.712	0.921	0.913
RotatE	0.715	0.703	0.901	0.887
TuckER	0.736	0.724	0.877	0.870
URGE	0.572	0.570	0.593	0.593
UKGE	0.769	0.768	0.933	0.929
BEUrRE	0.796	0.795	0.942	0.942
UKGE(rule+)	0.789	0.788	0.955	0.956
BEUrRE(rule+)	**0.801**	**0.803**	**0.966**	**0.970**

Table 4.5: Mean nDCG for fact ranking. *linear* stands for linear gain, and *exp.* stands for exponential gain.

ignores relation information and yields worse predictions than other models. UKGE explicitly models uncertainty of facts and is the best performing baseline.

The proposed BEUrRE leads to more improvements under both the unconstrained and logically constrained settings. Under the unconstrained setting, BEUrRE offers consistently better performance over UKGE. Specifically, on CN15k, BEUrRE leads to 0.027 improvement in both linear nDCG and exponential nDCG. On NL27k, it offers 0.009 higher linear nDCG and 0.013 higher exponential nDCG. Similar to the results on the confidence prediction task, even unconstrained BEUrRE is able to outperform the logically constrained UKGE (rule+) on CN15k without incorporating any constraints of relations. This further confirms the superior expressive power of box embeddings.

In summary, box embeddings improve accuracy and consistency of reasoning and BEUrRE delivers better fact ranking performance than baselines.

4.6 Conclusion

This chapter presents a novel uncertain knowledge graph embedding method with calibrated probabilistic semantics. Our model BEUrRE encodes each entity as a Gumble box representation whose volume represents marginal probability. A relation is modeled as two affine transforms on the head and tail entity boxes. We also incorporate logic constraints that capture the high-order dependency of facts and enhance global reasoning consistency. Extensive experiments demonstration the promising capability of BEUrRE on confidence prediction and fact ranking for uncertain knowledge graphs.

CHAPTER 5

Knowledge Graph Completion via Ensemble Knowledge Transfer

In this chapter, we present our ensemble learning framework that combines multiple knowledge graphs in different languages for query answering.

5.1 Introduction

Existing representation learning methods mainly investigate knowledge graph query answering within a single monolingual knowledge graph. As different language-specific knowledge graphs have their own strengths and limitations on data quality and coverage, we investigate a more natural solution, which seeks to combine embedding models of multiple knowledge graphs in an ensemble-like manner. This approach offers several potential benefits. First, embedding models of well-populated knowledge graphs (e.g. English knowledge graphs) are expected to capture richer knowledge because of better data quality and denser graph structures. Therefore, they would provide ampler signals to facilitate inferring missing facts on sparser knowledge graphs. Second, combining the embeddings allows exchanging complementary knowledge across different language-specific knowledge graphs. This provides a versatile way of leveraging specific knowledge that is better known in some knowledge graphs than the others. For example, consider the facts about the oldest Japanese novel *The Tale of Genji*. English DBpedia [53] only records its genre as *Monogatari* (story), whereas Japanese DBpedia identifies more genres, including *Love Story*, *Royal Family Related Story*, *Monogatari* and *Literature-Novel*. Similarly, it is reasonable to expect a Japanese knowledge graph embedding model to offer significant advantages in inferring

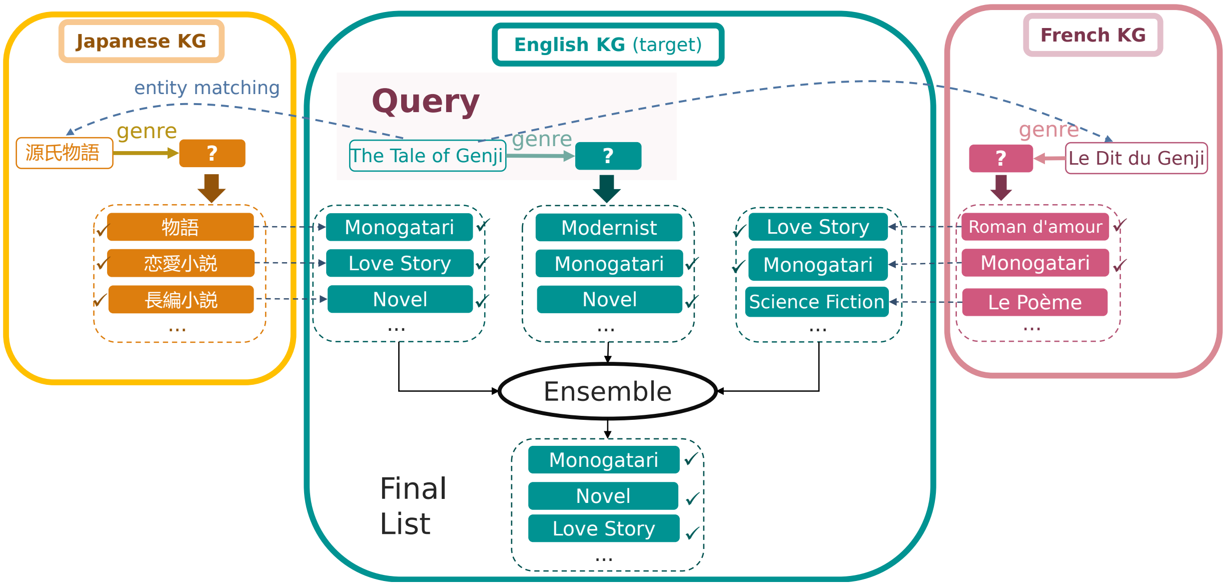

Figure 5.1: A depiction of the ensemble inference process answering the query *(The Tale of Genji, genre, ?t)* with multiple language-specific knowledge graph embeddings. Ground truth answers are marked*Monogatari* is a traditional Japanese literary form.

knowledge about other Japanese cultural entities such as *Nintendo* and *Mount Fuji*. Moreover, ensemble inference provides a mechanism to assess the credibility of different knowledge sources and thus leads to a more accurate final prediction.

Despite the potential benefits, combining predictions from multiple knowledge graph embeddings represents a non-trivial technical challenge. On the one hand, knowledge transfer across different embeddings is hindered by the lack of reliable alignment information that bridges different knowledge graphs. Recent works on multilingual knowledge graph embeddings provide support for automated entity matching [18, 15, 84, 85]. However, the performance of the state-of-the-art entity matching methods is still far from perfect [85], which may cause erroneous knowledge transfer between two knowledge graphs. On the other hand, independently extracted and maintained language-specific knowledge graphs may inconsistently describe some facts, therefore causing different knowledge graph embeddings to give inconsistent predictions and raising a challenge to identifying the trustable sources. For instance, while the English DBpedia strictly distinguishes the network of a TV series (e.g. BBC) from its channel (e.g. BBC One) with two separate relations, i.e., `network` and `channel`, the Greek DBpedia only uses `channel` to represent all of those. Another example of inconsistent information is that Chinese DBpedia labels the birth place of the ancient Chinese poet *Li Bai* as *Sichuan, China*, which is mistakenly recorded as *Chuy, Kyrgyz* in English DBpedia. Due to the rather independent extraction process of each knowledge graph, such inconsistencies are inevitable, calling upon a reliable approach to identify credible knowledge among various sources.

In this chapter, we propose `KEns` (Knowledge Ensemble), which, to the best of our knowledge, is the first ensemble framework of knowledge graph embedding models. Fig. 5.1 depicts the ensemble inference process of `KEns`. `KEns` seeks to improve knowledge graph completion in a multilingual setting, by combining predictions from embedding models of multiple language-specific knowledge graphs and identifying the most probable answers from those prediction results that are not necessarily consistent. Experiments on five real-world language-specific knowledge graphs show that `KEns` significantly improves state-of-the-art fact prediction methods that solely rely on a single knowledge graph embedding. We also provide detailed case studies to interpret how a sparse, low-resource knowledge graph can benefit from embeddings of other knowledge graphs, and how exclusive knowledge in one knowledge graph can be broadcasted to others.

5.2 Related Work

We hereby discuss two lines of work that are closely related to this topic, in addition to monolingual knowledge graph embedding models, which have been discussed in Section 2.2

5.2.1 Multilingual Knowledge Graph Embeddings

Recent studies have extended embedding models to bridge multiple knowledge graphs, typically for knowledge graphs of multiple languages. MTransE [18] jointly learns a transformation across two separate translational embedding spaces along with the knowledge graph structures. BootEA [84] introduces a bootstrapping approach to iteratively propose new alignment labels to enhance the performance. MuGNN [12] encodes knowledge graphs via multi-channel Graph Neural Network to reconcile the structural differences. Some others also leverage side information to enhance the alignment performance, including entity descriptions [15, 108], attributes [92, 83, 106], neighborhood information [99, 54, 86, 85] and degree centrality measures [65]. A systematic summary of relevant approaches is given in a recent survey by [87]. Although these approaches focus on the knowledge graph alignment that is different from the problem we tackle here, such techniques can be leveraged to support entity matching between knowledge graphs, which is a key component of our framework.

5.2.2 Ensemble Techniques

Ensemble learning has been widely used to improve machine learning results by combining multiple models on the same task. Representative approaches include voting, bagging [11], stacking [102] and boosting [31]. Boosting methods seek to combine multiple weak models into a single strong model, particularly by learning model weights from the sample distribution. Representative methods include AdaBoost [31] and RankBoost [30], which target at classification and ranking respectively. AdaBoost starts with a pool of weak classifiers and iteratively selects the best one based on the sample weights in that iteration. The final classifier is a linear combination of the selected weak classifiers, where each classifier is weighted by its performance. In each iteration, sample weights are updated according to the selected classifier so that the subsequent classifiers will focus more on the *hard* samples. RankBoost seeks to extend AdaBoost to ranking model combination. The model weights are learned from the ranking performance in a boosting manner. In this chapter, we extend RankBoost to combine ranking results from multiple knowledge graph embedding models. This technique addresses knowledge graph completion by combining knowledge from multiple sources and effectively compensates for the inherent errors in any entity matching processes.

5.3 Method

In this section, we introduce `KEns`, an embedding-based ensemble inference framework for multilingual knowledge graph completion.

`KEns` conducts two processes: *embedding learning* and *ensemble inference*. The embedding learning process trains the *knowledge model* that encodes entities and relations of every knowledge graph in a shared embedding space, as well as the *alignment model* that seizes the correspondence in different knowledge graphs and enables the projection of queries and answers across different knowledge graph embeddings. The ensemble inference process combines the predictions from multiple knowledge graph embeddings to improve fact prediction. Particularly, to assess the confidence of predictions from each source, we introduce a boosting method to learn entity-specific weights for knowledge models.

5.3.1 Preliminaries

A knowledge graph G consists of a set of (relational) facts $\{(h, r, t)\}$, where h and t are the head and tail entities of the fact (h, r, t), and r is a relation. Specifically, $h, t \in E$ (the set of entities in G), and $r \in R$ (the set of relations). To cope with knowledge graph completion, the fact prediction task seeks to fill in the right entity for the missing head or tail of an unseen triple. Without loss of generality, we hereafter discuss the case of predicting missing tails. We refer to a triple with a missing tail as a *query* $q = (h, r, \underline{?t})$. The answer set $\Omega(q)$ consists of all the right entities that fulfill q. For example, we may have a query *(The Tale of Genji, genre, $\underline{?t}$)*, and its answer set will include *Monogatari*, *Love Story*, etc.

Given knowledge graphs in M languages $G_1, G_2, \ldots, G_M$ ($|E_i| \leq |E_j|, i < j$), we seek to perform fact prediction on each of those by transferring knowledge from the others. We consider fact prediction as a ranking task in the knowledge graph embedding space, which is to transfer the query to external knowledge graphs and to combine predictions from multiple embedding models into a final ranking list. Particularly, given the existing situation of the major knowledge graphs, we use the following settings: (i) entity alignment information is available between any two knowledge graphs, though limited; and (ii) relations in different language-specific knowledge graphs are represented with a unified schema. The reason for the assumption is that unifying relations is usually feasible, since the number of relations is often much smaller compared to the enormous number of entities in knowledge graphs. This has been de facto achieved in a number of influential knowledge bases, including DBpedia [53], Wikidata [95] and YAGO [68]. In contrast, knowledge graphs often consist of numerous entities that cannot be easily aligned, and entity alignment is available only in small amounts.

5.3.2 Embedding Learning

The embedding learning process jointly trains the *knowledge model* and the *alignment model* following [18], while self-learning is added to improve the alignment learning. The details are described below.

5.3.2.1 Knowledge Model

A knowledge model seeks to encode the facts of a knowledge graph in the embedding space. For each language-specific knowledge graph, it characterizes the plausibility of its facts. Notation-wise, we use boldfaced $\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}$ as embedding vectors for head h, relation r and tail t respectively. The learning objective is to minimize the following margin ranking loss:

$$\mathcal{J}_K^G = \sum_{\substack{(h,r,t)\in G,\\ (h',r,t')\notin G}} [f(\boldsymbol{h'}, \boldsymbol{r}, \boldsymbol{t'}) - f(\boldsymbol{h}, \boldsymbol{r}, \boldsymbol{t}) + \gamma]_+ \tag{5.1}$$

where $[\cdot]_+ = \max(\cdot, 0)$, and f is a model-specific triple scoring function. The higher score indicates the higher likelihood that the fact is true. γ is a hyperparameter, and (h', r, t') is a negative sampled triple obtained by randomly corrupting either head or tail of a true triple (h, r, t).

We here consider two representative triple scoring techniques: TransE [9] and RotatE [88]. TransE models relations as translations between head entities and tail entities in a Euclidean space, while RotatE models relations as rotations in a complex space. The triple scoring functions are defined as follows.

$$f_{\text{TransE}}(h, r, t) = -\|\boldsymbol{h} + \boldsymbol{r} - \boldsymbol{t}\|_2 \tag{5.2}$$

$$f_{\text{RotatE}}(h, r, t) = -\|\boldsymbol{h} \circ \boldsymbol{r} - \boldsymbol{t}\|_2 \tag{5.3}$$

where $\circ : \mathbb{C}^d \times \mathbb{C}^d \rightarrow \mathbb{C}^d$ denotes Hadamard product for complex vectors, and $\| \cdot \|_2$ denotes L_2 norm.

5.3.2.2 Alignment Model

An alignment model is trained to match entity counterparts between two knowledge graphs on the basis of a small amount of seed entity alignment. We embed all knowledge graphs in one vector space and make each pair of aligned entities embedded closely. Given two knowledge graphs G_i and G_j with $|E_i| \leq |E_j|$, the alignment model loss is defined as:

$$\mathcal{J}_A^{G_i \leftrightarrow G_j} = \sum_{(e_i, e_j) \in \Gamma_{G_i \leftrightarrow G_j}} \|\boldsymbol{e}_i - \boldsymbol{e}_j\|_2 \tag{5.4}$$

where $e_i \in E_i, e_j \in E_j$ and $\Gamma_{G_i \leftrightarrow G_j}$ is the set of seed entity alignment between G_j and G_i. Assuming the potential inaccuracy of alignment, we do not directly assign the same vector to aligned entities of different language-specific knowledge graphs.

Particularly, as the seed entity alignment is provided in small amounts, the alignment process conducts self-learning, where training iterations incrementally propose more training data on unaligned entities to guide subsequent iterations. At each iteration, if a pair of unaligned entities in two knowledge graphs are mutual nearest neighbors according to the CSLS measure [21], `KEns` adds this highly confident alignment to the training data.

5.3.2.3 Learning Objective

We conduct joint training of knowledge models for multiple knowledge graphs and alignment models between each pair of them via minimizing the following loss function:

$$\mathcal{J} = \sum_{m=1}^{M} \mathcal{J}_K^{G_m} + \lambda \sum_{i=1}^{M} \sum_{j=i+1}^{M} \mathcal{J}_A^{G_i \leftrightarrow G_j} \tag{5.5}$$

where $\mathcal{J}_K^{G_m}$ is the loss of the knowledge model on G_m as defined in Eq (5.1), $\mathcal{J}_A^{G_i \leftrightarrow G_j}$ is the alignment loss between G_i and G_j. λ is a positive hyperparameter that weights the two model components. Following [18], instead of directly optimizing $\mathcal{J}$ in Eq. (5.5), our implementation optimizes each $\mathcal{J}_K^G$ and each $\lambda\mathcal{J}_A^{G_i\leftrightarrow G_j}$ alternately in separate batches. In addition, we enforce L_2-regularization to prevent overfitting.

5.3.3 Ensemble Inference

We hereby introduce how KEns performs fact prediction on multiple knowledge graphs via ensemble inference.

5.3.3.1 Cross-Lingual Query and Knowledge Transfer

To facilitate the process of completing knowledge graph G_i with the knowledge from another knowledge graph G_j, KEns first predicts the alignment for entities between G_i and G_j. Then, it uses the alignment to transfer queries from G_i to G_j, and transfer the results back. Specifically, alignment prediction is done by performing an kNN search in the embedding space for each entity in the smaller knowledge graph (i.e. the one with fewer entities) and find the closest counterpart from the larger knowledge graph. Inevitably, some entities in the larger knowledge graph will not be matched with a counterpart due to the 1-to-1 constraint. In this case, we do not transfer queries and answers for that entity.

5.3.3.2 Weighted Ensemble Inference

We denote the embedding models of $G_1, \ldots, G_M$ as $f_1, \ldots, f_M$. On the target knowledge graph where we seek to make predictions, given each query, the entity candidates are ranked by the weighted voting score of the models:

$$s(e) = \sum_{i=1}^{M} w_i(e) N_i(e) \tag{5.6}$$

where e is an entity on the target knowledge graph, and $w_i(e)$ is an entity-specific model weight, $N_i(e)$ is 1 if e is ranked among top K by f_i, otherwise 0.

We propose three variants of $\texttt{KEns}$ that differ in the computing of $w_i(e)$, namely $\texttt{KEns}_b$, $\texttt{KEns}_v$ and $\texttt{KEns}_m$. Specifically, $\texttt{KEns}_b$ learns an entity-specific weight $w_i(e)$ for each entity in a *boosting* manner, $\texttt{KEns}_v$ fixes $w_i(e) = 1$ for all f_i and e (i.e. *majority voting)*, and $\texttt{KEns}_m$ adopts *mean reciprocal rank* (MRR) of f_i on the validation set of the target knowledge graph as $w_i(e)$. We first present the technical details of the boosting-based $\texttt{KEns}_b$.

5.3.3.3 Boosting based Weight Learning

$\texttt{KEns}_b$ seeks to learn model weights for ranking combination, which aims at reinforcing correct beliefs and compensating for alignment error. An embedding model that makes more accurate predictions should receive a higher weight. Inspired by RankBoost [30], we reduce the ranking combination problem to a classifier ensemble problem. $\texttt{KEns}_b$ therefore learns model weights in a similar manner as AdaBoost.

To compute entity-specific weights $w_i(e)$, $\texttt{KEnS}_b$ evaluates the performance of f_i on a set of *validation queries* related to e. These queries are converted from all the triples in the validation set that mention e. An example of validation queries for the entity *The Tale of Genji* is given as below.

Example 5.3.1. *Examples of triples and validation queries for the entity* The Tale of Genji.

Triples:

{*(The Tale of Genji, country, Japan)*
(The Tale of Genji, genre, Monogatari)
(The Tale of Genji, genre, Love Story)}

Queries:

$Q = \{q_1 =$ *(The Tale of Genji, country, ?t)*
$q_2 =$ *(The Tale of Genji, genre, ?t)*$\}$

Similar to RankBoost [30], given a query q, $\texttt{KEns}_b$ evaluates the ranking performance of a model by checking if each of the *critical entity pairs* $\{(e, e')\}$ is ranked in correct order, where e is a ground truth tail and e' is an incorrect one. An example of critical entity pairs is given as below:

Example 5.3.2. *Critical entity pairs for the query* (The Tale of Genji, genre, ?t). *Ground truth tails are boldfaced. Pairs with x-marks indicate wrong prediction orders.*

Correct ranking :

***Monogatari, Love Story**, Modernist, Science Fiction*

Predicted ranking:

*Modernist, **Monogatari, Love Story**, Science Fiction*

Critical pair ranking results:

*(**Monogatari**, Modernist)* ✗, *(**Love Story**, Modernist)* ✗

*(**Monogatari**, Science Fiction)* ✓,

*(**Love Story**, Science Fiction)* ✓

Uncritical pairs:

(Monogatari, Love Story), (Modernist, Science Fiction)

The overall objective of $\texttt{KEnS}_b$ is to minimize the sum of ranks of all correct answers in the combined ranking list $\sum_q \sum_{e \in \Omega(q)} r(e)$, where $\Omega(q)$ is the answer set of query q and $r(e)$ is the rank of entity e in the combined ranking list of the ensemble inference. Essentially, the above objective is minimizing the number of mis-ordered critical entity pairs in the combined ranking list. Let the set of all the critical entity pairs from all the validation queries of an entity as P. [30] have proved that, when using RankBoost, this ranking loss is bounded as follows:

$$|\{p : p \in P, p \text{ is mis-ordered}\}| \leq |P| \prod_{m=1}^{M} Z^m$$

where M is the number of knowledge graphs and therefore the maximum number of rounds in boosting. Z^m is the weighted ranking loss of the m-th round:

$$Z^m = \sum_{p \in P} D^m(p) e^{-w^m [\![p]\!]^m} \tag{5.7}$$

where $[\![p]\!]^m = 1$ if the critical entity pair p is ranked in correct order by the selected embedding model in the m-th round, otherwise $[\![p]\!]^m = -1$, $D^m(p)$ is the weight of the critical entity pair p in the m-th round, and w^m is the weight of the chosen model in that round. Now the ranking combination problem is reduced to a common classifier ensemble problem.

The boosting process alternately repeats two steps: (i) Evaluate the ranking performance of the embedding models and choose the best one f^m according to the entity pair weight distribution in that round; (ii) Update entity pair weights to put more emphasis on the pairs which f_m ranks incorrectly.

Entity pair weights are initialized uniformly over P as $D^1(p) = \frac{1}{|P|}, p \in P$. In the m-th round ($m = 1, 2, ..., M$), $\texttt{KEnS}_b$ chooses an embedding model f^m and sets its weight w^m, seeking to minimize the weighted ranking loss Z^m defined in Eq.(5.7). By simple calculus, when choosing the embedding model f_i as the model of the m-th round, w_i^m should be set as follows to minimize Z^m:

$$w_i^m = \frac{1}{2} \ln\left(\frac{\sum_{p \in P, [\![p]\!]=1} D^m(p)}{\sum_{p \in P, [\![p]\!]=-1} D^m(p)}\right) \tag{5.8}$$

As we can see from Eq. (5.8), the higher w_i^m indicates the better performance of f_i under the current entity pair weight distribution D^m. We select the best embedding model in the m-th round f^m based on the maximum weight $w^m = \max\{w_1^m, ..., w_M^m\}$.

After choosing the best model f^m at this iteration, we update the entity pair weight distribution to put more emphasis on what f^m ranked wrong. The new weight distribution D^{m+1} is updated as:

$$D^{m+1}(p) = \frac{1}{Z^m} D^m(p) e^{-w^m [\![p]\!]^m} \tag{5.9}$$

where Z^m works as a normalization factor. $\texttt{KEnS}_b$ decreases the weight of $D(p)$ if the selected model ranks the entity pair in correct order and increases the weight otherwise. Thus, $D(p)$ will

Lang.	EN	FR	ES	JA	EL
#Ent.	13,996	13,176	12,382	11,805	5,231
#Rel.	831	178	144	128	111
#Triples	80,167	49,015	54,066	28,774	13,839

Table 5.1: Statistics of DBP-5Ldataset. *Ent.* and *Rel.* stand for entities and relations respectively.

tend to concentrate on the pairs whose relative ranking is hardest to determine.

For queries related to a specific entity, this process is able to recognize the embedding models that perform well on answering those queries and rectify the mistakes made in the previous iteration.

5.3.3.4 Other Ensemble Techniques

We also investigate two other model variants with simpler ensemble techniques.

1. Majority Vote ($\texttt{KEns}_v$): A straightforward ensemble method is to re-rank entities by their nomination counts in the prediction of all knowledge models, which substitutes the voting score (Eq. 5.6) with $s(e) = \sum_{i=1}^{M} N_i(e)$, where $N_i(e)$ is 1 if e is ranked among the top K by the knowledge model f_i, otherwise 0. When there is a tie, we order by the MRR given by the models on the validation set.

2. MRR Weighting ($\texttt{KEns}_m$): MRR is a widely-used metric for evaluating the ranking performance of a model [9, 105, 91], which may also serve as a weight metric for estimating the prediction confidence of each language-specific embedding in ensemble inference [74]. Let the MRR of f_i be u_i on the validation set, the entities are ranked according to the weighted voting score $s(e) = \sum_{i=1}^{M} u_i N_i(e)$.

5.4 Experiments

In this section, we conduct the experiment of fact prediction by comparing KEns variants with various knowledge graph embeddings. We also provide a detailed case study to help understand the principle of ensemble knowledge transfer.

5.4.1 Experiment Settings

To the best of our knowledge, existing datasets for fact prediction contain only one monolingual knowledge graph or bilingual knowledge graphs. Hence, we prepared a new dataset DBP-5L, which contains five language-specific knowledge graphs extracted from English (EN), French (FR), Spanish (ES) and Japanese (JA) and Greek (EL) DBpedia [53]. Table 5.1 lists the statistics of the contributed dataset DBP-5L. The relations of the five knowledge graphs are represented in a unified schema, which is consistent with the problem definition in Section 5.3.1. The English knowledge graph is the most populated one among the five. To produce knowledge graphs with a relatively consistent set of entities, we induce the subgraphs by starting from a set of seed entities where we have alignment among all language-specific knowledge graphs and then incrementally collecting triples that involve other entities. Eventually between any two knowledge graphs, the alignment information covers around 40% of entities. Based on the same set of seed entities, the Greek knowledge graph ends up with a notably smaller vocabulary and fewer triples than the other four. We split the facts in each knowledge graph into three parts: 60% for training, 30% for validation and weight learning, and 10% for testing.

5.4.1.1 Experimental Setup

We use the Adam [49] as the optimizer and fine-tune the hyper-parameters by grid search based on $Hits@1$ on the validation set. We select among the following sets of hyper-parameter values: learning rate $lr \in \{0.01, 0.001, 0.0001\}$, dimension $d \in \{64, 128, 200, 300\}$, batch size $b \in \{256, 512, 1024\}$, and TransE margin $\gamma \in \{0.3, 0.5, 0.8\}$. The best setting is $\{lr = 0.001$, $d = 300$, $b = 256\}$ for `KEns`(TransE) and $\{lr = 0.01$, $d = 200$, $b = 512\}$ for `KEns`(RotatE). The margin for TransE is 0.3. The L_2 regularization coefficient is fixed as 0.0001.

5.4.1.2 Evaluation Protocol

For each test case (h, r, t), we consider it as a query $(h, r, \underline{?t})$ and retrieve top K prediction results for $\underline{?t}$. We compare the proportion of queries with correct answers ranked within top K retrieved entities. We report three metrics with K as $1, 3, 10$. $Hits@1$ is equivalent to accuracy. All three metrics are preferred to be higher. Although another common metric, Mean Reciprocal Rank (MRR), has been used in previous works [9], it is not applicable to the evaluation of our framework because our ensemble framework combines the top entity candidates from multiple knowledge models and yields top K final results without making any claims for entities out of this scope. Following previous works, we use the "filtered" setting with the premise that the candidate space has excluded the triples that have been seen in the training set [98].

5.4.1.3 Competitive Methods

We compare six variants of KEns, which are generated by combining two knowledge models and three ensemble inference techniques introduced in in Section 5.3. For baseline methods, besides the single-embedding TransE [9] and RotatE [88], we also include DistMult [105], TransD [43], and HolE [61]. After extensive hyperparameter tuning, the baselines are set to their best configurations. We also include a baseline named *RotatE+PARIS*, which trains RotatE on 5 knowledge graphs and uses the representative non-embedding symbolic entity alignment tool PARIS [80] for entity matching. PARIS delivered entity matching predictions for 58%-62% entities in the English, French, and Spanish knowledge graph, but almost no matches are delivered for entities in the Greek and Japanese knowledge graph, since PARIS mainly relies on entity label similarity. The results on the Greek and Japanese knowledge graph are thus omitted for RotatE+PARIS.

5.4.2 Main Results

The results are reported in Table 5.2. As shown, the ensemble methods by KEns lead to consistent improvement in fact prediction. Overall, the ensemble inference leads to 1.1%-13.0% of improvement in $Hits@1$ over the best baseline methods. The improved accuracy shows that it is effective

KG	Greek			Japanese			Spanish			French			English		
Hits@k (%)	1	3	10	1	3	10	1	3	10	1	3	10	1	3	10
TransD	2.8	16.9	29.8	4.2	16.3	28.8	2.12	20.4	11.5	3.3	14.4	25.7	2.9	15.4	27.4
DistMult	8.9	13.0	11.3	9.3	18.4	27.5	7.4	15.0	22.4	6.1	14.3	23.8	8.8	19.4	30.0
HolE	4.2	9.5	18.3	25.5	29.5	32.8	20.1	26.8	29.4	22.4	24.4	28.9	12.3	20.4	25.4
TransE	13.1	23.4	43.7	21.1	34.4	48.5	13.5	29.4	45.0	17.5	33.1	48.8	7.3	16.4	29.3
KEns$_v$(TransE)	23.1	36.7	64.7	22.6	35.2	52.5	15.0	28.3	**49.0**	18.7	29.4	52.0	10.8	20.4	**39.4**
KEns$_m$(TransE)	26.3	42.1	65.8	26.1	37.7	55.3	16.8	**32.9**	48.6	20.5	35.6	52.8	11.4	21.2	31.3
KEns$_b$(TransE)	**26.4**	$\underline{\mathbf{42.4}}$	$\underline{\mathbf{66.1}}$	**26.7**	**39.8**	**56.4**	**17.4**	**32.6**	48.3	**20.8**	**35.9**	**53.1**	**11.7**	**21.8**	32.0
RotatE	14.5	18.8	36.2	26.4	36.2	60.2	21.2	31.6	53.9	23.2	29.4	55.5	12.3	25.4	30.4
RotatE+PARIS	-	-	-	-	-	-	20.8	39.4	59.1	22.8	32.4	60.8	12.4	22.7	31.5
KEns$_v$(RotatE)	20.5	34.3	50.1	31.9	50.0	65.0	20.8	41.0	59.9	23.7	42.7	61.9	13.4	23.6	34.2
KEns$_m$(RotatE)	22.0	35.0	51.4	32.0	49.9	65.0	21.2	41.6	60.0	24.5	44.8	62.5	12.1	24.5	34.3
KEns$_b$(RotatE)	$\underline{\mathbf{27.5}}$	**40.6**	**56.5**	$\underline{\mathbf{32.9}}$	$\underline{\mathbf{49.9}}$	$\underline{\mathbf{64.8}}$	$\underline{\mathbf{22.3}}$	$\underline{\mathbf{42.4}}$	$\underline{\mathbf{60.6}}$	$\underline{\mathbf{25.2}}$	$\underline{\mathbf{44.5}}$	$\underline{\mathbf{62.6}}$	$\underline{\mathbf{14.4}}$	$\underline{\mathbf{27.0}}$	$\underline{\mathbf{39.6}}$

Table 5.2: Fact prediction results on DBP-5L. The overall best results are under-scored.

to leverage complementary knowledge from external knowledge graphs for knowledge graph completion. We also observe that KEns brings larger gains on sparser knowledge graphs than on the well-populated ones. Particularly, on the low-resource Greek knowledge graph, KEns$_b$(RotatE) improves $Hits@1$ by as much as 13.0% over its single-knowledge graph counterpart. This finding corroborates our intuition that the knowledge graph with lower knowledge coverage and sparser graph structure benefits more from complementary knowledge.

Among the variants of ensemble methods, KEns$_m$ offers better performance than KEns$_v$, and KEns$_b$ outperforms the other two in general. For example, on the Japanese knowledge graph, KEns$_v$(TransE) improves $Hits@1$ by 3.5% from the single-knowledge graph TransE, while KEns$_m$ leads to a 5.0% increase, and KEns$_b$ further provides a 5.6% of improvement. The results suggest that MRR is an effective measure of the trustworthiness of knowledge models during ensemble inference. Besides, KEns$_b$ is able to assess trustworthiness at a finer level of granularity by learning entity-specific model weights and can thus further improve the performance.

In summary, the promising results by KEns variants show the effectiveness of transferring and leveraging cross-lingual knowledge for knowledge graph completion. Among the ensemble techniques, the boosting technique represents the most suitable one for combining the prediction results from different models.

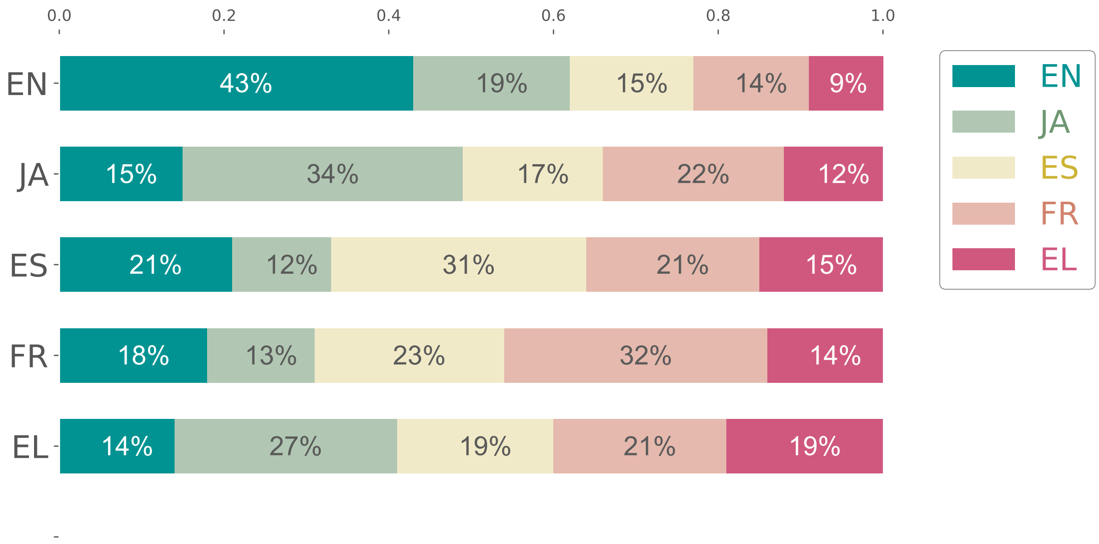

Figure 5.2: Average model weights learned by $\texttt{KEns}_b$(TransE).

5.4.3 Case Studies

In this section, we provide case studies to show how $\texttt{KEns}$ is able to transfer cross-lingual knowledge to populate different knowledge graphs.

5.4.3.1 Model Weights

The key to the significantly enhanced performance of $\texttt{KEns}_b$ is the effective combination of multilingual knowledge from multiple sources. Fig 5.2 shows the average model weight learnt by $\texttt{KEns}_b$(TransE), which depicts how external knowledge from cross-lingual knowledge graphs contributes to target knowledge graph completion in general. The model weights imply that sparser knowledge graphs benefit more from the knowledge transferred from others. Particularly, when predicting for the Greek knowledge graph, the weights of other languages sums up to 81%. This observation indicates that the significant boost received on the Greek knowledge graph comes with the fact that it has accepted the most complementary knowledge from others. In contrast, when predicting on the most populated English knowledge graph, the other language-specific models give a lesser total weight of 57%.

Among the three KEns variants, the superiority of $\texttt{KEns}_b$ is attributed to identification of more

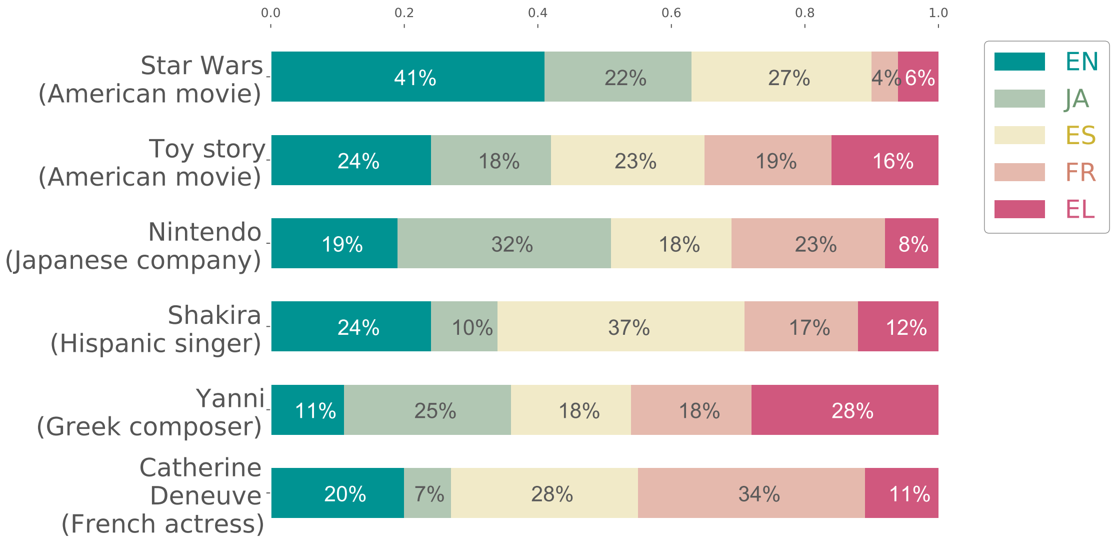

Figure 5.3: Examples of language-specific model weights learned by KEns$_b$(TransE). Percentages have been rounded.

credible knowledge sources, thus making more accurate predictions. For language-specific knowledge graphs, the higher level of credibility often stems from the cultural advantage the knowledge graph has over the entity. Fig 5.3 presents the model weights for 6 culture-related entities learned by KEns$_b$(TransE). It shows that KEns can locate the language-specific knowledge model that has a cultural advantage and assign it with a higher weight, which is the basis of an accurate ensemble prediction.

5.4.3.2 Ensemble Inference

To help understand how the combination of multiple knowledge graphs improves knowledge graph completion and show the effectiveness of leveraging complementary culture-specific knowledge , we present a case study about predicting the fact (Nintendo, industry, ?t) for English knowledge graph. Table 5.3 lists the top 3 predicted tails yielded by the KEns(TransE) variants, along with those by the English knowledge model and supporter knowledge models before ensemble. The predictions made by the Japanese knowledge graph are the closest to the ground truths. The reason may be that Japanese knowledge graph has documented much richer knowledge about this Japanese video game company, including many of the video games that this company has re-

Table 5.3: An example of fact prediction on the English knowledge graph by the English knowledge model, four supporter knowledge models, and KEns(TransE) variants. Top 3 predicted tails for the query (Nintendo, industry, ?t) are listed. Ground truths are boldfaced.

Model	Top 3 Predicted Tails
English	Television, Publishing, Information technology
Japanese	**Video game**, Anime, **Consumer electronics**
Spanish	Music, Telecommunication, Retail
French	Retail, Television, **Video game**,
Greek	Nintendo, Music, Wii
KEns_v	[**Video game**, Television](tie), Music
KEns_m	Television, **Video game**, Music
KEns_b	**Video game**, Television, **Consumer electronics**

leased. Among the three KEns variants, KEns_b correctly identifies Japanese as the most credible source and yields the best ranking.

5.5 Conclusion

In this chapter, we have proposed a new ensemble prediction framework aiming at collaboratively predicting unseen facts using embeddings of different language-specific knowledge graphs. In the embedding space, our approach jointly captures both the structured knowledge of each knowledge graph and the entity alignment that bridges the knowledge graphs. The significant performance improvements delivered by our model on the task of knowledge graph completion were demonstrated by extensive experiments. This work also suggests promising directions of future research. One is to exploit the potential of KEns on completing low-resource knowledge graphs, and the other is to extend the ensemble transfer mechanism to population sparse domain knowledge in biological [36] and medical knowledge bases [109]. Pariticularly, we also seek to ensure the global logical consistency of predicted facts in the ensemble process by incorporating probabilistic constraints [19].

CHAPTER 6

Answering Complex First-Order Logical Queries

6.1 Introduction

One of the fundamental tasks over knowledge graphs is to answer complex queries involving logical reasoning, e.g., answering First-Order Logical (FOL) queries with existential quantification ($\exists$), conjunction ($\wedge$), disjunction ($\vee$), and negation ($\neg$). For instance, the question "Who sang the songs that were written by John Lennon or Paul McCartney but never won a Grammy Award?" can be expressed as the FOL query shown in Fig 6.1.

This task is challenging due to time complexity and incompleteness of knowledge graphs. FOL query answering has been studied as a graph query optimization problem in the database community [37, 111, 73]. These methods traverse the knowledge graph to retrieve answers for each sub-query and then merge the results. Though being extensively studied, these methods cannot well resolve the above-mentioned challenges. The time complexity of traversing on knowledge graph exponentially grows with the query complexity and is affected by the size of the intermediate results. This makes it difficult to scale to modern knowledge graphs, whose entities are often numbered in millions [8, 95]. For example, Wikidata is one of the most influential knowledge graphs and reports that their query engine fails when the number of entities in a sub-query (e.g. people born in Germany) exceeds a certain threshold[1]. In addition, real-world knowledge graphs are often incomplete, which prevents directly answering many queries by searching knowledge graphs. A recent study shows that only 0.5% of football players in Wikidata have a highly complete profile, while over 40% contain only basic information [6].

[1] https://www.wikidata.org/wiki/Wikidata:SPARQL_query_service/query_optimization

$$q = V_?: \exists V \quad (Compose(\text{John Lennon}, V) \lor Compose(\text{Paul McCartney}, V)) \\ \land \neg AwardedTo(\text{Grammy Award}, V) \land SungBy(V, V_?)$$

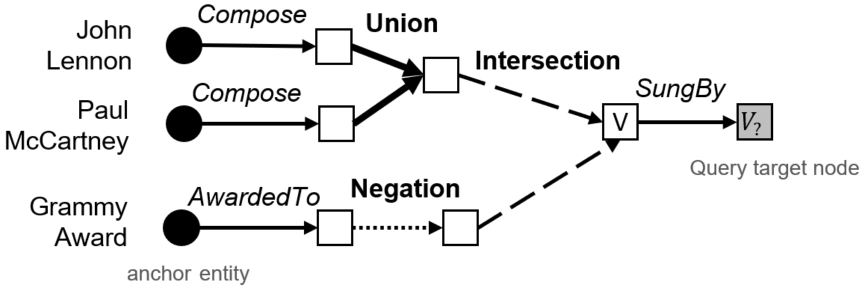

Figure 6.1: FOL query and its dependency graph for the question “Who sang the songs that were written by John Lennon or Paul McCartney but never won a Grammy Award?”.

To address the challenges of time complexity and knowledge graph incompleteness, a line of recent studies [35, 69, 71] embed logical queries and entities into the same vector space. The idea is to represent a query using a tree-shaped *dependency graph* (Figure 6.1) and embed a complex logical query by iteratively computing embeddings from *anchor entities* to the *target node* in a bottom-up manner. The continuous and meaningful entity embeddings empower these approaches to handle missing edges. In addition, these models significantly reduce time and space complexity for inference, as they reduce query answering to dense similarity matching of query and entity embeddings and can speed it up using methods like maximum inner product search (MIPS) [76].

These methods nonetheless entail several limitations: First, the logic operators in these models are often defined ad-hoc, and many do not satisfy basic logic laws (e.g. the associative law $(\psi_1 \wedge \psi_2) \wedge \psi_3 \equiv \psi_1 \wedge (\psi \wedge \psi_3)$ for logical formulae ψ_1, ψ_2, ψ_3), which limits their inference accuracy. Second, the logical operators of existing works are based on deep architectures, which require many training queries containing such logic operations to learn the parameters. This greatly limits the models’ scope of application, since it is challenging to collect a large number of reasonable complex queries with accurate answers.

Our goal is to create a logical query embedding framework that satisfies logical laws and provides learning-free logical operators. We hereby present `FuzzQE` (Fuzzy Query Embedding), a fuzzy logic based embedding framework for answering logical queries on knowledge graphs. We borrow the idea of fuzzy logic and use the fuzzy conjunction, disjunction, and negation to implement logical operators in a more principled and learning-free manner. Our approach provides the following advantages over existing approaches: (i) FuzzQE employs differentiable logical operators that fully satisfy the axioms of logical operations and can preserve logical operation properties in vector space. This superiority is corroborated by extensive experiments on two benchmark datasets, which demonstrate that FuzzQE delivers a significantly better performance compared to state-of-the-art methods in answering FOL queries. (ii) Our logical operations do not require learning any operator-specific parameters. We conduct experiments to show that even when our model is only trained with link prediction, it achieves better results than state-of-the-art logical query embedding models trained with extra complex query data. This represents a huge advantage in real-world applications since complex FOL training queries are often arduous to collect. In addition, when complex training queries are available, the performance of FuzzQE can be further enhanced.

In addition to proposing this novel and effective framework, we propose some basic properties that an embedding model ought to possess as well as analyze whether existing models can fulfill these conditions. The analysis provides theoretical guidance for future research on embedding-based logical query answering models.

6.2 Related Work

Embedding entities in Knowledge Graphs (knowledge graphs) into continuous embeddings have been extensively studied [9, 105, 91, 88], which can answer one-hop relational queries via link prediction. These models, however, cannot handle queries with multi-hop [34] or complex logical reasoning. [35] thus propose a graph-query embedding (GQE) framework that encodes a conjunctive query via a dependency graph with relation projection and conjunction ($\wedge$) as operators. [69] extend GQE by using box embedding to represent entity sets, where they define the disjunction ($\vee$) operator to support Existential Positive First-Order (EPFO) queries. [81] concurrently propose to represent sets as count-min sketch [22] that can support conjunction and disjunction operators. More recently, [71] further include the negation operator ($\neg$) by modeling the query and entity set as beta distributions. [32] extend FOL query answering to probabilistic databases. These query embedding models have shown promising results to conduct multi-hop logical reasoning over incomplete knowledge graphs efficiently regarding time and space; however, we found that these models do not satisfy the axioms of either Boolean logic [20] or fuzzy logic [50], which limits their inference accuracy. To address this issue, our approach draws from fuzzy logic and uses the fuzzy conjunction, disjunction, and negation operations to define the logical operators in vector space.

In addition to the above logical query embedding models, a recent work CQD [2] proposes training an embedding-based knowledge graph completion model (e.g. ComplEx [91]) to impute missing edges during inference and merge entity rankings with *t-norms* and *t-conorms* [50]. Using beam search for inference, CQD has demonstrated strong capability of generalizing from knowledge graph edges to arbitrary EPFO queries. However, CQD has severe scalability issues since it involves scoring every entity for every atomic query. This is undesirable in real-world applications, since the number of entities in real-world knowledge graphs are often in millions [8, 95]. Furthermore, its inference accuracy is thus bound by knowledge graph link prediction performance. In contrast, our model is highly scalable, and its performance can be further enhanced when additional complex queries are available for training.

6.3 Preliminaries

A knowledge graph consists of a set of triples $\langle e_s, r, e_o \rangle$, with $e_s, e_o \in \mathcal{E}$ (the set of entities) denoting the subject and object entities respectively and $r \in \mathcal{R}$ (the set of relations) denoting the relation between e_s and e_o. Without loss of generality, a knowledge graph can be represented as a First-Order Logic (FOL) Knowledge Base, where each triple $\langle e_s, r, e_o \rangle$ denotes an atomic formula $r(e_s, e_o)$, with $r \in \mathcal{R}$ denoting a binary predicate and $e_s, e_o \in \mathcal{E}$ as its arguments.

We aim to answer FOL queries expressed with existential quantification ($\exists$), conjunction ($\wedge$), disjunction($\vee$), and negation ($\neg$). [2] The disjunctive normal form (DNF) of an FOL logical query q

[2] As in previous works [69, 2, 71], we do not consider FOL queries with universal quantification ($\forall$). Queries with universal quantification do not apply in real-world knowledge graphs since no entity connects with all the other entities.

is defined as follows:

$$q[V_?] \triangleq V_? : \exists V_1, ..., V_k (v_{11} \wedge ... \wedge v_{1N_1}) \vee ... \vee (v_{M1} \wedge ... \wedge v_{MN_M})$$

where $V_?$ is the *target* variable of the query, and $V_1, ..., V_K$ denote the bound variable nodes. Each v_{mn} ($m = 1, ..., M, n = 1, ..., N_m$) represents a literal, i.e. a logical atom or the negation of a logical atom:

$$v_{mn} = \begin{cases} r(e, V) & r \in \mathcal{R}, e \in \mathcal{E}, V \in \{V_?, V_1, ..., V_k\} \\ \neg r(e, V) & r \in \mathcal{R}, e \in \mathcal{E}, V \in \{V_?, V_1, ..., V_k\} \\ r(V, V') & r \in \mathcal{R}, V, V' \in \{V_?, V_1, ..., V_k\}, V \neq V' \\ \neg r(V, V') & r \in \mathcal{R}, V, V' \in \{V_?, V_1, ..., V_k\}, V \neq V' \end{cases}$$

The goal of answering the logical query q is to find a set of entities $S_q = \{a | a \in \mathcal{E}, q[a] \text{ holds true}\}$, where $q[a]$ is a logical formula that substitutes the query target variable $V_?$ with the entity a.

A query can be considered as a combination of multiple sub-queries. For example, the query $q[V_?] = V_?$:Compose(John Lennon, $V_?$) $\wedge$ Compose(Paul McCartney, $V_?$) can be considered as $q_1 \wedge q_2$, where

$$q_1[V_?] = V_? : \text{Compose(John Lennon}, V_?)$$
$$q_2[V_?] = V_? : \textit{Compose}\text{(Paul McCartney}, V_?)$$

Formally, we have:

$$S_{q_1 \wedge q_2} = S_{q_1} \cap S_{q_1};$$
$$S_{q_1 \vee q_2} = S_{q_1} \cup S_{q_1};$$
$$S_{\neg q} = S_q^{\complement}$$

where $(\cdot)^{\complement}$ denote set complement respectively.

Notation wise, we use boldfaced notations $\mathbf{p}_e$ and $\mathbf{S}_q$ to represent the embedding for entity e

Table 6.1: Here we list eight logic laws (I - VIII) from classical logic [110] and give the corresponding properties that a query embedding model should possess. ψ_1, ψ_2, ψ_3 represent logical formulae. ϕ denotes the scoring function that estimates the probability that the entity e can answer the query q. $\phi(q, e) \uparrow \Rightarrow \phi(\neg q, e) \downarrow$ means $\phi(\neg q, e)$ is monotonically decreasing with regard to $\phi(q, e)$.

		Logic Law	Model Property
$\wedge$	I	Conjunction Elimination $\psi_1 \wedge \psi_2 \rightarrow \psi_1$ $\psi_1 \wedge \psi_2 \rightarrow \psi_2$	$\phi(q_1 \wedge q_2, e) \leq \phi(q_1, e)$ $\phi(q_1 \wedge q_2, e) \leq \phi(q_2, e)$
	II	Commutativity $\psi_1 \wedge \psi_2 \leftrightarrow \psi_2 \wedge \psi_1$	$\phi((q_1 \wedge q_2), e) = \phi((q_2 \wedge q_1), e)$
	III	Associativity $(\psi_1 \wedge \psi_2) \wedge \psi_3 \leftrightarrow$ $\psi_1 \wedge (\psi_2 \wedge \psi_3)$	$\phi((q_1 \wedge q_2) \wedge q_3, e)$ $= \phi(q_1 \wedge (q_2 \wedge q_3), e)$
$\vee$	IV	Disjunction Amplification $\psi_1 \rightarrow \psi_1 \vee \psi_2$ $\psi_1 \rightarrow \psi_1 \vee \psi_1$	$\phi(q_1, e) \leq \phi(q_1 \vee q_2, e)$ $\phi(q_2, e) \leq \phi(q_1 \vee q_2, e)$
	V	Commutativity $\psi_1 \vee \psi_2 \leftrightarrow \psi_2 \vee \psi_1$	$\phi((q_1 \vee q_2), e) = \phi((q_2 \vee q_1), e)$
	VI	Associativity $(\psi_1 \vee \psi_2) \vee \psi_3 \leftrightarrow$ $\psi_1 \vee (\psi_2 \vee \psi_3)$	$\phi((q_1 \vee q_2) \vee q_3, e)$ $= \phi(q_1 \vee (q_2 \vee q_3), e)$
$\neg$	VII	Involution $\neg\neg\psi_1 \rightarrow \psi_1$	$\phi(q, e) = \phi(\neg\neg q, e)$
	VIII	Non-Contradiction $\psi_1 \wedge \neg\psi_1 \rightarrow \overline{0}$	$\phi(q, e) \uparrow \quad \Rightarrow \quad \phi(\neg q, e) \downarrow$

and the embedding for S_q, i.e. the answer entity set for query q, respectively. We use ψ_1, ψ_2, ψ_3 denote logical formulae.

6.3.1 Logic Laws and Model Properties

The general idea of logical query embedding models is to recursively define the embedding of a query (e.g., $q_1 \wedge q_2$) based on logical operations on its sub-queries' embeddings (e.g., q_1 and q_2). These logical operations have to satisfy logic laws, which serve as additional constraints to learning-based query embedding models. Unfortunately, most existing query embedding models have (partially) neglected these laws, which result in inferior performance.

Table 6.2: Comparisons of different models regarding the properties of logical operations. *Expr.* stands for *expressivity* and indicates whether the model can handle such logical operations, and *closed* indicates whether the embedding is in a closed form. *Commu., Asso., Elim., Ampli., Inv.* and *Non-contra.* stand for commutativity, associativity, conjunction elimination, disjunction amplification, involution, and non-contradiction respectively.

	$\wedge$				$\vee$				$\neg$		
	Expr. (Closed)	*Com.*	*Asso.*	*Elim.*	*Expr.* (Closed)	*Com.*	*Asso.*	*Ampli.*	*Expr.* (Closed)	*Inv.*	*Non-Contra.*
GQE	✓(✓)	✓	✗	✗	✓(✗)	✓	✓	✓	✗	N/A	N/A
Query2Box	✓(✓)	✓	✓	✓	✓(✗)	✓	✓	✓	✗	N/A	N/A
BetaE	✓(✓)	✓	✗	✗	(i) DNF ✓(✗) (ii) DM ✓(✓)	✓ ✓	✓ ✓	✓ ✗	✓(✓)	✓	✗
FuzzQE	✓(✓)	✓	✓	✓	✓(✓)	✓	✓	✓	✓(✓)	✓	✓

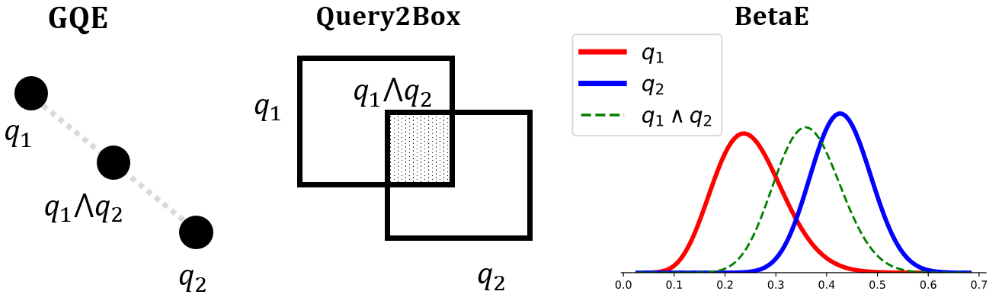

Figure 6.2: Illustration of query embeddings and embeddings of conjunctive queries in GQE, Query2Box, and BetaE. The conjunction operators takes embeddings of queries q_1, q_2 as input and produce an embedding for $q_1 \wedge q_2$.

In this section, we study these logic laws shared by both classical logic and basic fuzzy logic [110] and deduce several basic properties that the logical operators should possess. The logic laws and corresponding model properties are summarized in Table 6.1.

6.3.1.1 Axiomatic Systems of Logic

Let $\mathcal{L}$ be the set of all the valid logic formulae under a logic system, and $\psi_1, \psi_2, \psi_3 \in \mathcal{L}$ represent logical formulae. $I(\cdot)$ denotes the truth value of a logical formula. The semantics of Boolean Logic are defined by (i) the interpretation $I : \mathcal{L} \to \{0, 1\}$, (ii) the Modus Ponen inference rule "from ψ_1 and $\psi_1 \to \psi_2$ infer ψ_2", which characterizes logic implication ($\to$) as follows:

$$\psi_1 \to \psi_2 \quad \text{holds if and only if} \quad I(\psi_2) \geq I(\psi_1)$$

and (iii) a set of axioms written in Hilbert-style deductive systems [51]. Those axioms define other logic connectives via logic implication (→); for example, the following three axioms characterize the conjunction (∧) of Boolean logic [20]:

$$\begin{aligned}
&\psi_1 \wedge \psi_2 \rightarrow \psi_1 \\
&\psi_1 \wedge \psi_2 \rightarrow \psi_2 \\
&(\psi_3 \rightarrow \psi_1) \rightarrow ((\psi_3 \rightarrow \psi_2) \rightarrow (\psi_3 \rightarrow \psi_1 \wedge \psi_2))
\end{aligned}$$

The first two axioms guarantee that the truth value of $\psi_1 \wedge \psi_2$ never exceeds the truth values of ψ_1 and ψ_2, and the last one enforces that $I(\psi_1 \wedge \psi_2) = 1$ if $I(\psi_1) = I(\psi_2) = 1$. The three axioms also imply commutativity and associativity of $\wedge$.

6.3.1.2 Model Properties

Let $\phi(q, e)$ be the embedding model scoring function estimating the probability that the entity e can answer the query q. This means that $\phi(q, e)$ estimates the truth value $I(q[e])$, where $q[e]$ is a logical formula that uses e to fill q. For example, given the query $q = V_? : \textit{Compose}(\textit{John Lennon}, V_?)$ and the entity $e =$ *"Let it Be"*, $\phi(q, e)$ estimates the truth value of the logical formula *Compose*(*John Lennon*, *Let it Be*). We can thus use logic laws to deduce reasonable properties that a query embedding model should possess. For instance, $\psi_1 \wedge \psi_2 \rightarrow \psi_1$ is an axioms that characterizes logic conjunction (∧), which enforces that $I(\psi_1 \wedge \psi_2) \leq I(\psi_1)$, and we accordingly expect the embedding model to satisfy $\phi(q_1 \wedge q_2, e) \leq \phi(q_1, e)$, i.e., an entity e is less likely to satisfy $q_1 \wedge q_2$ than q_1.

Based on the the axioms and deduced logic laws of classical logic [29], we summarize a series of model properties that an embedding model should possess in Table 6.1. The list is not exhaustive but indicative.

6.3.2 Analysis of Prior Models on Model Properties

This section examines three representative logical query embedding models, namely GQE [35], Query2Box [69], and BetaE [71], regarding their capability of satisfying the properties in Table 6.1. We summarize our findings in Table 6.2. GQE, Query2Box, BetaE represent queries as vectors, boxes (axis-aligned hyper-rectangles), and Beta distributions, respectively. The embedding-based logical operators transform embeddings of sub-queries into embeddings of the outcome query.

6.3.2.1 Conjunction ($\wedge$)

Fig. 6.2 illustrates embedding-based conjunction operators of the three models, which takes embeddings of queries q_1, q_2 as input and produce an embedding for $q_1 \wedge q_2$. GQE, Query2Box, and BetaE are purposely constructed to be permutation invariant [35, 69, 71], and their conjunction operators all satisfy *commutativity* (Law II). The conjunction operators of GQE and BetaE do not satisfy *associativity* (III) since they rely on the operation of averaging, which is not associative. GQE does not satisfy *conjunction elimination* (I); for example, supposing that $\mathbf{p}_e = \frac{1}{2}(\mathbf{S}_{q_1} + \mathbf{S}_{q_2})$ and $\mathbf{S}_{q_1} \neq \mathbf{S}_{q_2}$, we have $\phi(q_1 \wedge q_2, e) > \phi(q_1, e)$. BetaE does not satisfy *conjunction elimination* (I) for similar reasons.

6.3.2.2 Disjunction ($\vee$)

Previous works handle disjunction in two ways: the *Disjunctive Normal Form (DNF) rewriting* approach proposed by Query2Box [69], and the *De Morgan's law (DM)* approach proposed by BetaE [71]. The DNF rewriting method involves rewriting each query as a DNF to ensure that the disjunction only appears in the last step, which enables the model to simply retain all input embeddings. The model correspondingly cannot represent the disjunction result as a closed form; for example, the disjunction of two boxes remains two separate boxes instead of one [69]. The DM approach uses De Morgan's law $\psi_2 \vee \psi_1 \equiv \neg(\neg\psi_2 \wedge \neg\psi_1)$ to compute the disjunctive query embedding, which requires the model to have a conjunction operator and a negation operator. This approach advantageously produces representation in a closed form, allowing disjunction to

Table 6.3: Prominent examples of t-norms and the corresponding t-norms derived by De Morgan's law and the negator $c(a) = 1 - a$. $a, b \in [0, 1]$. We list the special properties of the formulas in addition to the basic properties (i.e. commutativity, associativity, monotonicity, and boundary condition) of t-norm and t-conorm.

	t-norm ($\wedge$)	t-conorm ($\vee$)	Special Properties
minimum (Gödel)	$t(a, b) = \min(a, b)$	$s(a, b) = \max(a, b)$	idempotent
product	$t(a, b) = ab$	$s(a, b) = a + b - ab$	strict monotonicity
Łukasiewicz	$t(a, b) = \max(a + b - 1, 0)$	$s(a, b) = \min(a + b, 1)$	nilpotent

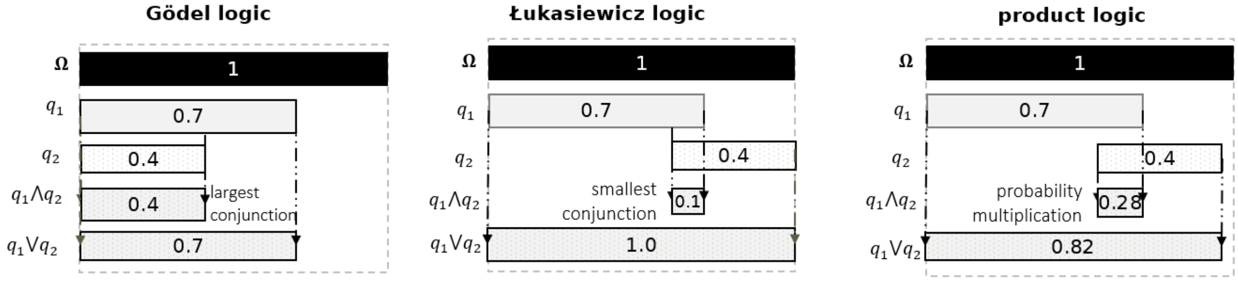

Figure 6.3: Illustration of fuzzy conjunction and disjunction, which is equivalent to fuzzy set intersection and union respectively.

be performed at any step of the computation. The disadvantage is that if the negation operator does not work well, the error will be amplified and affect disjunction. The BetaE$_{\text{DM}}$ does not satisfy *disjunction amplification* (IV) since its negation operator violates *non-contradiction* (VIII).

6.3.2.3 Negation

To the best of our knowledge, BetaE is the only previous model that can handle negation. BetaE has proved that its negation operator is *involutory* (VII). However, this operator lacks the *non-contradiction* property (VIII), as for BetaE $\phi(\neg q, e)$ is not monotonically decreasing with regard to $\phi(q, e)$.

6.3.3 Fuzzy Logic

Fuzzy logic differs from Boolean logic by associating every logical formula with a truth value in $[0, 1]$. Fuzzy logic systems usually retain the axioms of Boolean logic, which ensures that all logical operation behaviors are consistent with Boolean logic when the truth values are 0 or 1. Different

Table 6.4: Comparison between classical logic and product logic. F_{mL} denote all valid logic formulae under the logic system, and $\varphi, \psi \in F_{mL}$ are logical formulae. $I(\cdot)$ denotes the truth value of a logical formula.

	Classical Logic	Product Logic
Interpretation I	$I : F_{mL} \to \{0, 1\}$	$I : F_{mL} \to [0, 1]$
$I(\varphi \wedge \psi)$	$I(\varphi)I(\psi)$	$I(\varphi)I(\psi)$
$I(\varphi \vee \psi)$	$I(\varphi) + I(\psi) - I(\varphi)I(\psi)$	$I(\varphi) + I(\psi) - I(\varphi)I(\psi)$
$I(\varphi \to \psi)$	$\begin{cases} 1, & \text{if } I(\varphi) \leq I(\psi) \\ I(\psi), & \text{otherwise} \end{cases}$	$\begin{cases} 1, & \text{if}I(\varphi) \leq I(\psi) \\ I(\psi), & \text{otherwise} \end{cases}$

fuzzy logic systems add different axioms to define the logical operation behavior for the case when the truth value is in $(0, 1)$ [51]. A t-norm $\top : [0, 1] \times [0, 1] \mapsto [0, 1]$ represents generalized conjunction in fuzzy logic. Prominent examples of t-norms include Gödel t-norm $\top_{\text{min}}\{x, y\} = \min(x, y)$, product t-norm $\top_{\text{prod}}\{x, y\} = xy$, and Łukasiewicz t-norm$\top_{\text{Ł}ukasiewicz}(x, y) = \max\{0, x+y-1\}$, for $x, y \in [0, 1]$. Any other continuous t-norm can be described as an ordinal sum of these three basic ones [50]. Analogously, t-conorm are dual to t-norms for disjunction in fuzzy logic – given a t-norm $\top$, the t-conorm is defined as $\bot(x, y) = 1 - \top(1 - x, 1 - y)$ based on De Morgan's law and the negator $n(x) = 1 - x$ for $x, y \in [0, 1]$ [50]. The formulas of t-conorms that correspond to the minimum (Gödel), product, and Łukasiewicz t-norms are given in Table 6.3. An illustration of t-norm and t-conorm based conjunction and disjunction in fuzzy logic is given in Fig. 6.3. In Table 6.4, we compare the semantics of classical logic and product logic and show that product logic operations are fully compatible with classical logic. This technique inspired numerous subsequent works. For example, CQD [2] uses t-norms and t-conorms to rank entities for query answering on knowledge graphs.

6.4 Methodology

In this section, we propose our model `FuzzQE`, a framework for answering FOL queries in the presence of missing edges. `FuzzQE` embeds entities as *stochastic vectors* and queries as *fuzzy vectors* [47]. Logical operators are implemented via fuzzy conjunction, fuzzy disjunction and fuzzy negation in the embedding space.

6.4.1 Queries and Entities in Fuzzy Space

Predicting whether an entity can answer a query means predicting the probability that the entity belongs to the answer set of this query. In our work, we embed queries and entities to the *fuzzy space* $[0,1]^d$, a subspace of $\mathbb{R}^d$ [47].

6.4.1.1 Query Embedding

Consider a query q and its fuzzy answer set S_q, its embedding $\mathbf{S}_q$ is defined as a fuzzy vector $\mathbf{S}_q \in [0,1]^d$ [47]. Intuitively, let Ω denote the universe of all the elements, and let $\{U_i\}_{i=1}^d$ denote a partition over Ω as follows:

$$\begin{aligned} &\Omega = \cup_{i=1}^d U_i \\ &U_i \cap U_j = \emptyset \quad \text{for} \quad i \neq j \end{aligned}$$

Each dimension i of $\mathbf{S}_q$ denotes the probability whether the corresponding subset U_i is part of the answer set S_q:

$$\mathbf{S}_q(i) = \Pr(U_i \subseteq S_q).$$

6.4.1.2 Entity Embedding

For an entity e, we consider its embedding $\mathbf{p}_e$ from the same fuzzy space. To model its uncertainty, we model it as a categorical distribution to fall into each subset U_i as follows:

$$\begin{aligned} &\mathbf{p}_e \in [0,1]^d \\ &\mathbf{p}_e(i) = \Pr(e \in U_i) \\ &\sum_{i=1}^d \mathbf{p}_e(i) = 1. \end{aligned}$$

6.4.1.3 Score Function

Accordingly, the score function $\phi(q, e)$ is defined as the expected probability that e belongs to the fuzzy set S_q:

$$\begin{aligned}\phi(q, e) &= \mathbb{E}_{e \sim \mathbf{p}_e}[e \in S_q] \\ &= \sum_{i=1}^{d} \Pr(e \in U_i) \Pr(U_i \subseteq S_q) \\ &= \mathbf{S}_q{}^{\mathsf{T}} \mathbf{p}_e\end{aligned}$$

Note for query embedding in `FuzzQE`, the all-one vector $\mathbf{1}$ represents the universe set (i.e. Ω), and the all-zero vector $\mathbf{0}$ represents an empty set $\emptyset$.

The above representation and scoring provides the following benefits: (i) The representation is endowed with probabilistic interpretation, and (ii) each dimension of the embedding vector is between $[0, 1]$, which satisfies the domain and range requirements of fuzzy logic and allows the model to execute element-wise fuzzy conjunction/disjunction/negation.

6.4.2 Relation Projection for Atomic Queries

Atomic queries like $q = \textit{Compose}(\textit{John Lennon}, V_?)$ serve as the building blocks to compute the complex queries. To embed atomic queries, we associate each relation $r \in \mathcal{R}$ with a projection operator $\mathcal{P}_r$, which is modeled by a neural network with a weight matrix $\mathbf{W}_r \in \mathbb{R}^{d \times d}$ and a bias vector $\mathbf{b}_r \in \mathbb{R}^d$, and transforms an anchor entity embedding $\mathbf{p}_e$ into a query embedding:

$$\mathbf{S}_q = \mathcal{P}_r(\mathbf{p}_e) = \mathbf{g}(\mathrm{LN}(\mathbf{W}_r \mathbf{p}_e + \mathbf{b}_r))$$

where LN is Layer Normalization [3], and $\mathbf{g} : \mathbb{R}^d \mapsto [0, 1]^d$ is a mapping function that constrains $\mathbf{S}_q \in [0, 1]^d$. Particularly, we consider two different choices for $\mathbf{g}$:

$$\text{Logistic function} : \mathbf{g}(\mathbf{x}) = \frac{1}{1 + e^{-(\mathbf{x})}}$$

$$\text{Bounded rectifier} : \mathbf{g}(\mathbf{x}) = \min(\max(\mathbf{x}, 0), 1)$$

We follow [72] and adopt *basis-decomposition* to define $\mathbf{W}_r$ and $\mathbf{b}_r$:

$$\mathbf{W}_r = \sum_{j=1}^{K} \alpha_{rj} \mathbf{M}_j$$

$$\mathbf{b}_r = \sum_{j=1}^{K} \alpha_{rj} \mathbf{v}_j$$

Namely, $\mathbf{W}_r$ as a linear combination of K basis transformations $\mathbf{M}_j \in \mathbb{R}^{d \times d}$ with coefficients α_{rj} that depend on r. Similarly, $\mathbf{b}_r$ is a linear combination of K basis vectors $\mathbf{v}_j \in \mathbb{R}^d$ with coefficients α_{rj}. This form prevents the rapid growth in the number of parameters with the number of relations and alleviates overfitting on rare relations. It can be seen as a form of effective weight sharing among different relation types [72]. Atomic queries that project from one set to another can be embedded similarly.

In principle, any sufficiently expressive neural network or translation-based knowledge graph embedding model [9, 43] could also be employed as the relation projection operator in our framework.

6.4.3 Fuzzy Logic based Logical Operators

Fuzzy logic is mathematically equivalent to fuzzy set theory [51], with fuzzy conjunction equivalent to fuzzy set intersection, fuzzy disjunction equivalent to fuzzy set union, and fuzzy negation to fuzzy set complement. Fuzzy logic is thus used to define operations over fuzzy vectors. Particularly, we present `FuzzQE` with reference to product logic, one of the most prominent fuzzy logic systems [50]. The embeddings of $q_1 \wedge q_2$, $q_1 \vee q_2$, and $\neg q$ are computed as follows:

$$\begin{aligned} q_1 \wedge q_2: &\quad \mathcal{C}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2}) &&= \mathbf{S}_{q_1} \circ \mathbf{S}_{q_2} \\ q_1 \vee q_2: &\quad \mathcal{D}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2}) &&= \mathbf{S}_{q_1} + \mathbf{S}_{q_2} - \mathbf{S}_{q_1} \circ \mathbf{S}_{q_2} \\ \neg q: &\quad \mathcal{N}(\mathbf{S}_q) &&= \mathbf{1} - \mathbf{S}_q \end{aligned}$$

where $\circ$ denotes element-wise multiplication (fuzzy conjunction), $\mathbf{1}$ is the all-ones vector, and $\mathcal{C}, \mathcal{D}, \mathcal{N}$ denote the embedding based logical operators respectively.

6.4.4 Model Learning and Inference

Given a query q, we optimize the following objective:

$$L = -\log \sigma(\phi(q, e) - \gamma) - \frac{1}{k} \sum_{i=1}^{k} \log \sigma(\gamma - \phi(q, e'))$$

where $e \in S_q$ is an answer to the query, $e' \notin S_q$ represents a random negative sample, and γ denotes the margin. In the loss function, we use k random negative samples and optimize the average. We seek to maximize $\phi(q, e)$ for $e \in S_q$ and minimize $\phi(q, e')$ for $e' \in S_q$.

For the model inference, given a query q, FuzzQE embeds it as $\mathbf{S}_q$ and rank all the entities by $\phi(q, \cdot)$.

6.5 Theoretical Analysis

For FuzzQE, we present the following propositions with proof.

Proposition 1. *Our conjunction operator $\mathcal{C}$ is commutative, associative, and satisfies conjunction elimination.*

Proposition 2. *Our disjunction operator $\mathcal{D}$ is commutative, associative, and satisfies disjunction amplification.*

Proposition 3. *Our negation operator $\mathcal{N}$ is involutory and satisfies non-contradiction.*

6.5.1 Proof of Proposition 1

Commutativity: $\phi(q_1 \wedge q_2, e) = \phi(q_2 \wedge q_1, e)$

Proof. We have $\mathcal{C}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2}) = q_1 \circ q_2 = q_2 \circ q_1 = \mathcal{C}(\mathbf{S}_{q_2}, \mathbf{S}_{q_1})$ where $\circ$ denotes element-wise multiplication.

Therefore, $\phi(q_1 \wedge q_2, e) = \mathbf{p}_e{}^\intercal \mathcal{C}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2}) = \mathbf{p}_e{}^\intercal \mathcal{C}(\mathbf{S}_{q_2}, \mathbf{S}_{q_1}) = \phi(q_2 \wedge q_1, e)$. □

Associativity: $\phi((q_1 \wedge q_2) \wedge q_3, e) = \phi(q_1 \wedge (q_2 \wedge q_3), e)$

Proof. Since $\mathcal{C}(\mathcal{C}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2})), \mathbf{S}_{q_3}) = q_1 \circ q_2 \circ q_3 = \mathcal{C}(\mathbf{S}_{q_1}, \mathcal{C}(\mathbf{S}_{q_2}, \mathbf{S}_{q_3}))$, we have

$$\begin{aligned} &\phi((q_1 \vee q_2) \vee q_3, e) \\ =&\mathbf{p}_e{}^\intercal \mathcal{C}(\mathcal{C}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2})), \mathbf{S}_{q_3}) \\ =&\mathbf{p}_e{}^\intercal \mathcal{C}(\mathbf{S}_{q_1}, \mathcal{C}(\mathbf{S}_{q_2}, \mathbf{S}_{q_3})) \\ =&\phi(q_1 \vee (q_2 \vee q_3), e) \end{aligned}$$

□

Conjunction elimination: $\phi(q_1 \wedge q_2, e) \leq \phi(q_1, e)$**,** $\quad \phi(q_1 \wedge q_2, e) \leq \phi(q_2, e)$

Proof. $\phi(q_1 \wedge q_2, e) \leq \phi(q_1, e)$ can be proved by

$$\begin{aligned}
&\phi(q_1 \wedge q_2, e) \\
=&\mathbf{p}_e{}^{\intercal}\mathcal{C}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2}) \\
=&\mathbf{p}_e{}^{\intercal}(\mathbf{S}_{q_1} \circ \mathbf{S}_{q_2}) \\
=&\sum_{i=1}^{d} \mathbf{p}_{e i}\mathbf{S}_{q_{1_i}}\mathbf{S}_{q_{2_i}} \\
\leq&\sum_{i=1}^{d} \mathbf{p}_{e i}\mathbf{S}_{q_{1_i}} \\
=&\phi(q_1, e)
\end{aligned}$$

$\phi(q_1 \wedge q_2, e) \leq \phi(q_2, e)$ can be proved similarly. □

6.5.2 Proof of Proposition 2

Commutativity: $\phi(q_1 \vee q_2, e) = \phi(q_2 \vee q_1, e)$

Proof. We have $\mathcal{D}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2}) = \mathbf{S}_{q_1} + \mathbf{S}_{q_2} - \mathbf{S}_{q_1} \circ \mathbf{S}_{q_2} = \mathbf{S}_{q_2} + \mathbf{S}_{q_1} - \mathbf{S}_{q_2} \circ \mathbf{S}_{q_1} = \mathcal{D}(\mathbf{S}_{q_2}, \mathbf{S}_{q_1})$. Therefore, $\phi(q_1 \vee q_2, e) = \mathbf{p}_e{}^{\intercal}\mathcal{D}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2}) = \mathbf{p}_e{}^{\intercal}\mathcal{D}(\mathbf{S}_{q_2}, \mathbf{S}_{q_1}) = \phi(q_2 \vee q_1, e)$. □

Associativity: $\phi((q_1 \wedge q_2) \wedge q_3, e) = \phi(q_1 \wedge (q_2 \wedge q_3), e)$

Proof.

$$\begin{aligned}
&\mathcal{D}(\mathcal{D}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2})), \mathbf{S}_{q_3}) \\
=&\mathcal{D}(\mathbf{S}_{q_1} + \mathbf{S}_{q_2} - \mathbf{S}_{q_1} \circ \mathbf{S}_{q_2}, \mathbf{S}_{q_3}) \\
=&(\mathbf{S}_{q_1} + \mathbf{S}_{q_2} - \mathbf{S}_{q_1} \circ \mathbf{S}_{q_2}) + \mathbf{S}_{q_3} - (\mathbf{S}_{q_1} + \mathbf{S}_{q_2} - \mathbf{S}_{q_1} \circ \mathbf{S}_{q_2}) \circ \mathbf{S}_{q_3} \\
=&\mathbf{S}_{q_1} + \mathbf{S}_{q_2} + \mathbf{S}_{q_3} - \mathbf{S}_{q_1} \circ \mathbf{S}_{q_2} - \mathbf{S}_{q_1} \circ \mathbf{S}_{q_3} - \mathbf{S}_{q_2} \circ \mathbf{S}_{q_3} + \mathbf{S}_{q_1} \circ \mathbf{S}_{q_2} \circ \mathbf{S}_{q_3} \\
=&\mathcal{D}(\mathbf{S}_{q_1}, \mathcal{D}(\mathbf{S}_{q_2}, \mathbf{S}_{q_3}))
\end{aligned}$$

Therefore

$$\begin{aligned}
&\phi((q_1 \vee q_2) \vee q_3, e)\\
=&\mathbf{p}_e{}^\intercal \mathcal{D}(\mathcal{D}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2})), \mathbf{S}_{q_3})\\
=&\mathbf{p}_e{}^\intercal \mathcal{D}(\mathbf{S}_{q_1}, \mathcal{D}(\mathbf{S}_{q_2}, \mathbf{S}_{q_3}))\\
=&\phi(q_1 \vee (q_2 \vee q_3), e)
\end{aligned}$$

□

Disjunction amplification: $\phi(q_1 \vee q_2, e) \geq \phi(q_1, e)$**,** $\quad \phi(q_1 \vee q_2, e) \geq \phi(q_2, e)$

Proof. $\phi(q_1 \vee q_2, e) \geq \phi(q_1, e)$ can be proved by

$$\begin{aligned}
&\phi(q_1 \vee q_2, e)\\
=&\mathbf{p}_e{}^\intercal \mathcal{D}(\mathbf{S}_{q_1}, \mathbf{S}_{q_2})\\
=&\mathbf{p}_e{}^\intercal (\mathbf{S}_{q_1} + \mathbf{S}_{q_2} - \mathbf{S}_{q_1} \circ \mathbf{S}_{q_2})\\
=&\sum_{i=1}^{d} \mathbf{p}_{ei}(\mathbf{S}_{q_{1_i}} + \mathbf{S}_{q_{2_i}} - \mathbf{S}_{q_{1_i}} \mathbf{S}_{q_{2_i}})\\
=&\sum_{i=1}^{d} \mathbf{p}_{ei}\mathbf{S}_{q_{1_i}} + \mathbf{p}_{ei}\mathbf{S}_{q_{2_i}}(1 - \mathbf{S}_{q_{1_i}})\\
\geq&\sum_{i=1}^{d} \mathbf{p}_{ei}\mathbf{S}_{q_{1_i}}\\
=&\phi(q_1, e)
\end{aligned}$$

$\phi(q_1 \vee q_2, e) \geq \phi(q_2, e)$ can be proved similarly. □

6.5.3 Proof of Proposition 3

Involution: $\phi(q, e) = \phi(\neg\neg q, e)$

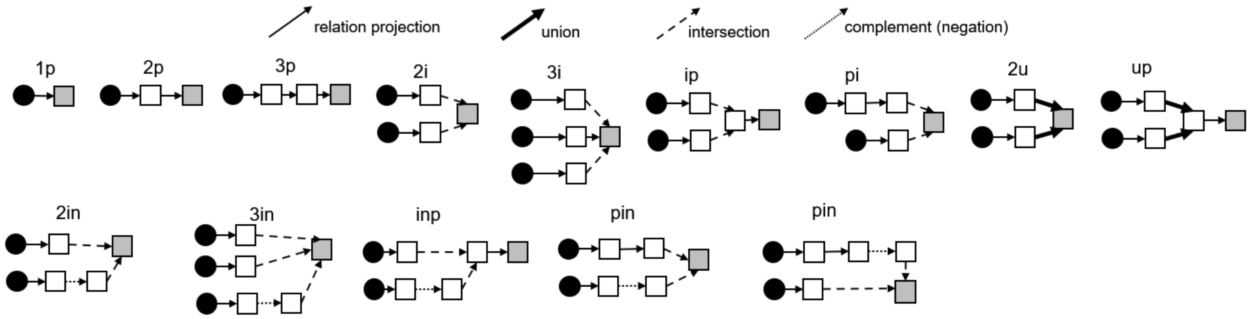

Figure 6.4: Query structure types used in training and evaluation. Naming convention: p for relation projection, i for conjunction (intersection), n for negation (complement), u for disjunction (union).

Proof.

$$\mathcal{N}(\mathcal{N}(q)) = \mathbf{1} - (\mathbf{1} - \mathbf{S}_q) = \mathbf{S}_q$$

Therefore $\phi(\neg\neg q, e) = \mathbf{p}_e{}^{\intercal}\mathcal{N}(\mathcal{N}(\mathbf{S}_q)) = \phi(q, e)$ □

Non-Contradiction: $\phi(q, e) \uparrow \Rightarrow \phi(\neg q, e) \downarrow$

Proof. The Łukasiewicz negation $c(x) = 1 - x$ is monotonically decreasing with regard to x. Therefore, $\phi(\neg q, e)$ is monotonically decreasing with regard to $\phi(q, e)$. □

6.6 Experiments

In this section, we evaluate the ability of `FuzzQE` to answer complex FOL queries over incomplete knowledge graphs.

6.6.1 Evaluation Setup

6.6.1.1 Datasets

We evaluate our model on two benchmark datasets provided by [71], which contain 14 types of logical queries on FB15k-237 [90] and NELL995 [104] respectively. These queries are generated based on the official training/validation/testing edge splits of those knowledge graphs. The knowl-

Table 6.5: Knowledge graph dataset statistics as well as training, validation, and test edge splits.

Dataset	Entities	Relations	Training Edges	Validation Edges	Test Edges	Total Edges
FB15k-237	14505	237	272115	17526	20438	310079
NELL	63361	200	114213	143234	14267	142804

Table 6.6: Number of training, validation, and test queries for different query structures. For columns that list multiple query structures, the number in the table represents the number of each query structure.

	Training		Validation		Test	
Dataset	1p/2p/3p/2i/3i	2in/3in/inp/pin/pni	1p	others	1p	others
FB15k-237	149,689	149,68	20,101	5,000	2,812	5,000
NELL995	107,982	10,798	16,927	4,000	17,034	4,000

edge graph statistics are summarized in Table 6.5. The 14 types of query structures in the datasets are shown in Fig. 6.4. We list the number of training/validation/test queries in Table 6.6. Note that these datasets provided by BetaE [71] are an improved and expanded version of the datasets provided by Query2Box [69]. Compared to the earlier version, the new datasets [71] contain 5 new types of queries that involve negation. The validation/test set of the original 9 query types are regenerated to ensure that the number of answers per query is not excessive, making this task more challenging. We exclude FB15k [9] as this dataset suffers from major test leakage [90].

6.6.1.2 Evaluation Protocol

We follow the evaluation protocol in [71]. To evaluate the model's generalization capability over incomplete knowledge graphs, the datasets are masked out so that each validation/test query answer pair involves imputing at least one missing edge. For each answer of a test query, we use the Mean Reciprocal Rank (MRR) as the major evaluation metric. We use the *filtered* setting [9] and filter out other correct answers from ranking before calculating the MRR.

6.6.1.3 Baselines

We consider three logical query embedding baselines for answering complex logical queries on knowledge graphs: GQE [35], Query2Box [69], and BetaE [71]. We also compare with one recent state-of-the-art query optimization model CQD [2]. For BetaE and CQD, we compare with the model variant that generally provides better performance, namely $\text{BetaE}_{\text{DNF}}$ and CQD-BEAM. Note that CQD cannot process complex logical queries during training and is thus trained with knowledge graph edges. To the best of our knowledge, BetaE is the only available baseline that can handle negation. Therefore, for GQE, Query2Box, and CQD, we compare with them only on EPFO queries (queries with $\exists, \wedge, \vee$ and without negation).

6.6.1.4 Model Configurations and Hyperparameters

We use AdamW [58] as the optimizer. Training terminates with early stopping based on the average MRR on the validation set with a patience of 15k steps. We run each method up to 450k steps. We repeat each experiment three times and report the average results. For GQE [35], Query2Box [69], and BetaE [71], we use the implementation from https://github.com/snap-stanford/knowledgegraphReasoning. For CQD, we use the implementation at https://github.com/uclnlp/cqd.

As in [69, 71], for fair comparison, we use the same embedding dimensionality d and the number of negative samples k for all the methods. With reference to [71], we set the embdding dimensionality to $d = 800$ and use $k = 128$ negative samples per positive sample. We fine-tune other hyperparameters and the choice of the subspace mapping function $\mathbf{g} : \mathbb{R}^d \rightarrow [0, 1]^d$ by grid search based on the average MRR on the validation set. We search hyperparameters in the following range: learning rate from $\{0.001, 0.0005, 0.0001\}$, number of relation bases from $\{30, 50, 100, 150\}$, batch size b from {128, 512, 1000}. **g** is chosen from from {Logistic function, Bounded rectifier}.

The best hyperparameter combination on FB15k-237 is learning rate 0.001, number of relation bases 150, batch size 512, **g** as a logistic function. The best combination on NELL995 is learning rate 0.0005, number of relation bases 30, batch size 1000, **g** as a bounded rectifier. For baselines GQE , Q2B, and BetaE, we use the best combinations reported by [71]. For CQD, we use the ones reported in [2]. We follow the setting in the official code repository for any hyperparameter unspecified in the chapter.

Each single experiment is run on CPU Intel® Xeon® E5-2650 v4 12-core and a single NVIDIA®

Table 6.7: **MRR results (%) on answering FOL queries.** Report MRR results (%) on test FOL queries. Avg_{EPFO} and Avg_{Neg} denote the average MRR on EPFO queries (queries with $\exists$, $\wedge$, $\vee$ and without negation) and queries containing negation respectively. Results of GQE, Query2Box, and BetaE are taken from [71].

Type of Model	Model	Avg_{EPFO}	Avg_{Neg}	1p	2p	3p	2i	3i	pi	ip	2u	up	2in	3in	inp	pin	pni
								FB15k-237									
Query Embedding	GQE	16.3	N/A	35.0	7.2	5.3	23.3	34.6	16.5	10.7	8.2	5.7	N/A	N/A	N/A	N/A	N/A
	Query2Box	20.1	N/A	40.6	9.4	6.8	29.5	42.3	21.2	12.6	11.3	7.6	N/A	N/A	N/A	N/A	N/A
	BetaE	20.9	5.5	39.0	10.9	10.0	28.8	42.5	22.4	12.6	12.4	9.7	5.1	7.9	7.4	3.5	3.4
	FuzzQE	**24.2**	**8.5**	42.2	**13.3**	**10.2**	**33.0**	**47.3**	**26.2**	**18.9**	**15.6**	**10.8**	**9.7**	**12.6**	**7.8**	**5.8**	**6.6**
Query Optimization	CQD	21.7	N/A	**46.3**	9.9	5.9	31.7	41.3	21.8	15.8	14.2	8.6	N/A	N/A	N/A	N/A	N/A
								NELL995									
Query Embedding	GQE	18.6	N/A	32.8	11.9	9.6	27.5	35.2	18.4	14.4	8.5	8.8	N/A	N/A	N/A	N/A	N/A
	Query2Box	22.9	N/A	42.2	14.0	11.2	33.3	44.5	22.4	16.8	11.3	10.3	N/A	N/A	N/A	N/A	N/A
	BetaE	24.6	5.9	53.0	13.0	11.4	37.6	47.5	24.1	14.3	12.2	8.5	5.1	7.8	10.0	3.1	3.5
	FuzzQE	**29.3**	**8.0**	58.1	**19.3**	**15.7**	39.8	**50.3**	**28.1**	**21.8**	**17.3**	**13.7**	**8.3**	**10.2**	**11.5**	**4.6**	**5.4**
Query Optimization	CQD	28.4	N/A	**60.0**	16.5	10.4	**40.4**	49.6	28.6	20.8	16.8	12.6	N/A	N/A	N/A	N/A	N/A

GP102 TITAN Xp (12GB) GPU. RAM size is 256GB. The operating system is Ubuntu 18.04.01. Our framework is implemented wtih Python 3.9 and Pytorch 1.9.

6.6.2 Main Results: Trained with FOL queries

We fist test the ability of FuzzQE to model arbitrary FOL queries when complex logical queries are available for training. Results are reported in Table 6.7.

6.6.2.1 Comparison with Query Embedding

As shown in Table 6.7, FuzzQE consistently outperforms all the logical query embedding baselines. For EPFO queries, FuzzQE improves the average MRR of best baseline BetaE [71] by 3.3% (ca. 15% relative improvement) on FB15k-237 and 4.7% (ca. 19% relative improvement) on NELL995. For queries with negation, FuzzQE significantly outperforms the only available baseline BetaE. On average, FuzzQE leads to 3.0% (54% relatively) improvement in MRR on FB15k-237 and 1.9% (32% relatively) on NELL995. On average, FuzzQE improves the MRR by 3.0% (54% relatively) on FB15k-237 and 2.1% (36% relatively) on NELL995 for queries containing negation. We hypothesize that this significant enhancement comes from the principled design of our negation operator that satisfies the axioms, while BetaE fails to satisfy the non-contradiction

Table 6.8: **MRR results (%) of logical query embedding models that are trained with only link prediction.** This task tests the ability of the model to generalize to arbitrary complex logical queries, when no complex logical query data is available for training. Avg$_{\text{EPFO}}$ and Avg$_{\text{Neg}}$ denote the average MRR on EPFO ($\exists, \wedge, \vee$) queries and queries containing negation respectively.

Model	Avg$_{\text{EPFO}}$	Avg$_{\text{Neg}}$	1p	2p	3p	2i	3i	pi	ip	2u	up	2in	3in	inp	pin	pni
								FB15k-237								
GQE	17.7	N/A	41.6	7.9	5.4	25.0	33.6	16.3	10.9	11.9	6.2	N/A	N/A	N/A	N/A	N/A
Query2Box	18.2	N/A	42.6	6.9	4.7	27.3	36.8	17.5	11.1	11.7	5.5	N/A	N/A	N/A	N/A	N/A
BetaE	15.8	0.5	37.7	5.6	4.4	23.3	34.5	15.1	7.8	9.5	4.5	0.1	1.1	0.8	0.1	0.2
`FuzzQE`	**21.8**	**6.6**	**44.0**	**10.8**	**8.6**	**32.3**	**41.4**	**22.7**	**15.1**	**13.5**	**8.7**	**7.7**	**9.5**	**7.0**	**4.1**	**4.7**
								NELL995								
GQE	21.7	N/A	47.2	12.7	9.3	30.6	37.0	20.6	16.1	12.6	9.6	N/A	N/A	N/A	N/A	N/A
Query2Box	21.6	N/A	47.6	12.5	8.7	30.7	36.5	20.5	16.0	12.7	9.6	N/A	N/A	N/A	N/A	N/A
BetaE	19.0	0.4	53.1	6.0	3.9	32.0	37.7	15.8	8.5	10.1	3.5	0.1	1.4	0.1	0.1	0.1
`FuzzQE`	**27.1**	**7.3**	**57.6**	**17.2**	**13.3**	**38.2**	**41.5**	**27.0**	**19.4**	**16.9**	**12.7**	**9.1**	**8.3**	**8.9**	**4.4**	**5.6**

property.

6.6.2.2 Comparison with Query Optimization: CQD

We next compare `FuzzQE` with a recent query optimization baseline, CQD [2] on EPFO queries. On average, `FuzzQE` provides 2.5% and 0.9% absolute improvement in MRR on FB15k-237 and NELL995 respectively. It is worth noting that `FuzzQE` outperforms CQD on most complex query structures on NELL995 even with slightly worse 1p performance. We hypothesize that the 1p performance difference on NELL995 comes from the differenct ability of different relation projection/link prediction models to encode sparse graphs.

A major motivation for learning logical query embedding is its high inference efficiency. We compare with CQD with regard to the time for answering a query. For CQD, we use its official experiment setting [2]. The beam search candidate number for CQD is set as 64, i.e. CQD finds top 64 entity candidates for each sub-query and uses it as seeds for search in the next round. For `FuzzQE`, we retrieve top 64 entity candidates for each query as well. We use FAISS [45] to speed up dense similarity search, where *exact measurement matching* is adopted instead of *approximate measurement matching*. FAISS cannot be applied to CQD, because (i) CQD is not a logical query embedding framework that retrieves entity answers by dense similarity search, and (ii) scoring an

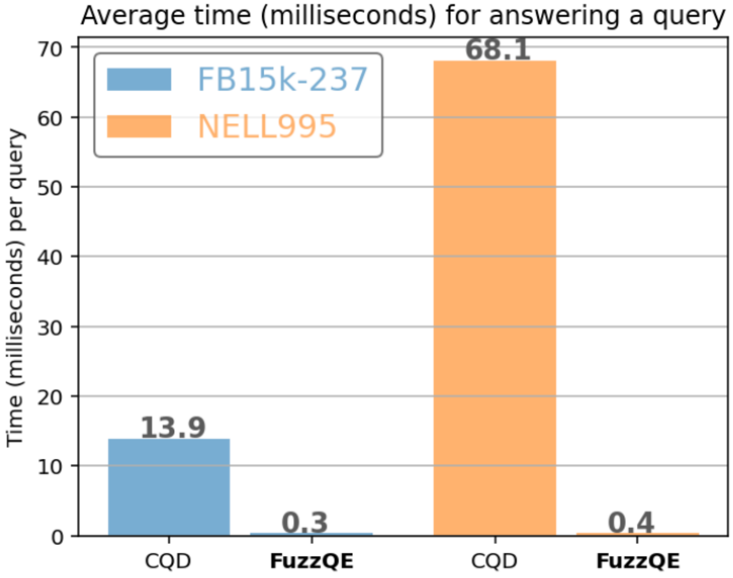

Figure 6.5: Average time (milliseconds) for answering an FOL query on a single NVIDIA® GP102 TITAN Xp (12GB) GPU. FB15k-237 contains 14,505 entities. NELL995 contains 63,361 entities, roughly 4 times the number of FB15k-237.

entity for a query involves computation in the complex number domain.

Fig 6.5 shows the average time of CQD and FuzzQE for answering a complex FOL query. On a NVIDIA® GP102 TITAN Xp (12GB), the average time for CQD to answer a FOL query on FB15k-237 is 13.9 ms (milliseconds), while FuzzQE takes only 0.3 ms. On NELL995, where the number of entities is 4 times the number in FB15k-237, the average time for CQD is 68.1 ms, whereas FuzzQE needs only 0.4 ms. CQD takes 170 times longer than FuzzQE. The reason is that CQD is required to score all the entities for each node in the dependency graph to obtain the top-k candidates for beam search. Consistent with the observation in [2], the main computation bottleneck of CQD are multi-hop queries (e.g. $3p$ queries), since the model is required to invoke the link prediction model for each node in the dependency graph to obtain the top-k candidates for the next step. We also note that as the number of entities increases, the time required by CQD to answer a query significantly grows. In contrast, the inference time of `FuzzQE` is almost independent of the number of entities and the complexity of the query.

6.6.3 Trained with only Link Prediction

This experiment tests the ability of the model to generalize to arbitrary complex logical queries when it is trained with only the link prediction task. To evaluate it, we train FuzzQE and other logical query embedding models using only knowledge graph edges (i.e. 1p queries). For baseline models GQE, Query2Box, and BetaE, we adapt them following the experiment settings of the Q2B-AVG-1P model discussed in [69]. Specifically, we set all the sub-query weights to 1.0 for this experiment.

As shown in Table 6.8, FuzzQE is able to generalize to complex logical queries of new query structures even if it is trained on link prediction and provides significantly better performance than baseline models. Compared to the best baseline, FuzzQE improves the average MRR by 3.6% for EPFO queries on FB15k-237 and 5.4% on NELL995. Regarding queries with negation, our model drastically outperforms the only available baseline BetaE across datasets. In addition, compared with the ones trained with complex FOL queries (in Table 6.7), It is worth nothing that FuzzQE trained with only link prediction can outperform BetaE that are trained with extra complex logical queries (in Table 6.7). This demonstrates the superiority of the logical operators in FuzzQE, which are designed in a principled and learning-free manner. Meanwhile, FuzzQE can still take advantage of additional complex queries as training samples to enhance entity embeddings.

6.7 Conclusion

We propose a novel logical query embedding framework FuzzQE for answering complex logical queries on knowledge graphs. Our model FuzzQE borrows operations from fuzzy logic and implements logical operators in a principled and learning-free manner. Extensive experiments show the promising capability of FuzzQE on answering logical queries on knowledge graphs.

CHAPTER 7

Conclusion and Future Directions

In this dissertation, we provide the following contributions regarding three key aspects of representation learning based query answering on knowledge graphs:

1. We propose two new methods to encode uncertain facts in the embedding space, enable reasoning on uncertain knowledge graphs, and address fact uncertainty in query ansewring.
2. We enhance query answering accuracy by leveraging complementary knowledge from knowledge graphs in various languages. We propose an ensemble learning framework that can populate entity alignment and assess the credibility of different knowledge sources thereby leading to a more accurate final prediction.
3. We propose a fuzzy logic based logical query embedding framework for answering logical queries on knowledge graphs, where logical operators are implemented in a principled and learning-free manner. In addition, we identify some basic properties that an embedding model ought to possess as well as analyze whether existing models can fulfill these conditions. This analysis provides theoretical guidance for future research on embedding-based logical query answering models.

Chapters 3 and 4 introduce our uncertain knowledge graph embedding models UKGE and BEUrRE respectively. Unlike previous models that characterize facts using binary classification techniques, UKGE learns embeddings according to confidence scores of facts. We also introduce probabilistic soft logic to infer the confidence scores and provide extra training supervision, and we propose two variants of UKGE based on different regression functions. Chapter 4 extends the previous chapter's technique to improve reasoning on sparse uncertain knowledge graphs by providing

BEUrRE, which is a novel uncertain knowledge graph embedding model with probabilistic semantics. BEUrRE considers each entity as a binary random variable and models each entity as a box (i.e. axis-aligned hyperrectangle) in the vector space, with relations between two entities representing affine transforms on the subject and object entity boxes. The geometry of the boxes endows the model with calibrated probabilistic semantics and facilitates incorporating relation property constraints. The results are encouraging and suggest various extensions, including deeper transformation architectures and alternative geometries to allow imposing additional rules. We are also interested in extending the use of the proposed technologies into more downstream tasks, such as knowledge association [82] and event hierarchy induction [96] as well as for ontology construction and population, since box embeddings are capable of capturing concepts' granularities.

Chapter 5 enhances deterministic knowledge graph completion by transferring complementary knowledge across language-specific knowledge graphs. The proposed framework KEns embeds all knowledge graphs in a shared embedding space, where the association of entities is captured based on self-learning. KEns then performs ensemble inference to combine prediction results from embeddings of multiple language-specific knowledge graphs, whereby multiple ensemble techniques are investigated. Experiments demonstrate that our model delivers significant performance improvements to query answering on knowledge graphs. This work also suggests that future research exploits the potential of KEns for query answering on low-resource knowledge graphs and to extend the ensemble transfer mechanism to populate sparse domain knowledge in biological [36] and medical knowledge bases [109].

Chapter 6 presents the logical query embedding framework FuzzQE for answering first-order logical queries on knowledge graphs. This model FuzzQE borrows operations from fuzzy logic and implements logical operators in a principled and learning-free manner, and extensive experiments show promising capability of FuzzQE for answering logical queries on knowledge graphs. The results are encouraging and suggest various extensions, including introducing logical rules into learning, as well as studying the potential use of predicate fuzzy logic systems and other deeper transformation architectures. Future research could also use the defined logical operators to incorporate logical rules to enhance reasoning on knowledge graphs. Furthermore, we are interested in jointly learning embeddings for logical queries, natural language questions ,and entity labels to enhance question answering on knowledge graphs.

BIBLIOGRAPHY

[1] Ralph Abboud, İsmail İlkan Ceylan, Thomas Lukasiewicz, and Tommaso Salvatori. Boxe: A box embedding model for knowledge base completion. In *Proceedings of the 34th Annual Conference on Neural Information Processing Systems, NeurIPS*, 2020.

[2] Erik Arakelyan, Daniel Daza, Pasquale Minervini, and Michael Cochez. Complex query answering with neural link predictors. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[3] Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *CoRR*, abs/1607.06450, 2016.

[4] Stephen Bach, Bert Huang, Ben London, and Lise Getoor. Hinge-loss markov random fields: Convex inference for structured prediction. 2013.

[5] Stephen H Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. Hinge-loss markov random fields and probabilistic soft logic. *Journal of Machine Learning Research*, 2017.

[6] Vevake Balaraman, Simon Razniewski, and Werner Nutt. Recoin: Relative completeness in wikidata. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, page 1787–1792, Republic and Canton of Geneva, CHE, 2018. International World Wide Web Conferences Steering Committee.

[7] Ivana Balazevic, Carl Allen, and Timothy M. Hospedales. Tucker: Tensor factorization for knowledge graph completion. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP*, pages 5184–5193. Association for Computational Linguistics, 2019.

[8] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 2008.

[9] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2787–2795, 2013.

[10] Khaoula Boutouhami, Jiatao Zhang, Guilin Qi, and Huan Gao. Uncertain ontology-aware knowledge graph embeddings. In Xin Wang, Francesca A. Lisi, Guohui Xiao, and Elena Botoeva, editors, *Semantic Technology*, pages 129–136. Springer Singapore, 2020.

[11] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[12] Yixin Cao, Zhiyuan Liu, Chengjiang Li, Juanzi Li, and Tat-Seng Chua. Multi-channel graph neural network for entity alignment. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*, pages 1452–1461. Association for Computational Linguistics, 2019.

[13] Jiaao Chen, Jianshu Chen, and Zhou Yu. Incorporating structured commonsense knowledge in story completion. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6244–6251. AAAI Press, 2019.

[14] Minmin Chen, Kilian Q Weinberger, Zhixiang Xu, and Fei Sha. Marginalizing stacked linear denoising autoencoders. *Journal of Machine Learning Research*, 2015.

[15] Muhao Chen, Yingtao Tian, Kai-Wei Chang, Steven Skiena, and Carlo Zaniolo. Co-training embeddings of knowledge graphs and entity descriptions for cross-lingual entity alignment. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 3998–4004. International Joint Conferences on Artificial Intelligence Organization, 2018.

[16] Muhao Chen, Yingtao Tian, Xuelu Chen, Zijun Xue, and Carlo Zaniolo. On2vec: Embedding-based relation prediction for ontology population. In *Proceedings of the SIAM Conference on Data Mining*, 2018.

[17] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1511–1517. ijcai.org, 2017.

[18] Muhao Chen, Yingtao Tian, Mohan Yang, and Carlo Zaniolo. Multilingual knowledge graph embeddings for cross-lingual knowledge alignment. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1511–1517. International Joint Conferences on Artificial Intelligence, 2017.

[19] Xuelu Chen, Muhao Chen, Weijia Shi, Yizhou Sun, and Carlo Zaniolo. Embedding uncertain knowledge graphs. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pages 3363–3370. AAAI Press, 2019.

[20] Karel Chvalovskỳ. On the independence of axioms in bl and mtl. *Fuzzy Sets and Systems*, 197:123–129, 2012.

[21] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. In *International Conference on Learning Representations (ICLR)*, 2018.

[22] Graham Cormode and S. Muthukrishnan. An improved data stream summary: the count-min sketch and its applications. *J. Algorithms*, 55(1):58–75, 2005.

[23] Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *6th International*

Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. OpenReview.net, 2018.

[24] Shib Sankar Dasgupta, Michael Boratko, Dongxu Zhang, Luke Vilnis, Xiang Lorraine Li, and Andrew McCallum. Improving local identifiability in probabilistic box embeddings. In *Advances in Neural Information Processing Systems*, 2020.

[25] Shib Sankar Dasgupta, Xiang Li, Michael Boratko, Dongxu Zhang, and Andrew McCallum. Box-to-box transformation for modeling joint hierarchies, 2021.

[26] Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. Convolutional 2d knowledge graph embeddings. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pages 1811–1818. AAAI Press, 2018.

[27] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, and Rayid Ghani, editors, *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 601–610. ACM, 2014.

[28] John Duchi, Elad Hazan, et al. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul), 2011.

[29] János C. Fodor and Marc Roubens. *Fuzzy Preference Modelling and Multicriteria Decision Support*, volume 14 of *Theory and Decision Library*. Springer, 1994.

[30] Yoav Freund, Raj Iyer, Robert E. Schapire, and Yoram Singer. RankBoost: An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research (JMLR)*, 4(6):933–969, 2004.

[31] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[32] Tal Friedman and Guy Van den Broeck. Symbolic querying of vector spaces: Probabilistic databases meets relational embeddings. In Ryan P. Adams and Vibhav Gogate, editors, *Proceedings of the Thirty-Sixth Conference on Uncertainty in Artificial Intelligence, UAI 2020, virtual online, August 3-6, 2020*, volume 124 of *Proceedings of Machine Learning Research*, pages 1268–1277. AUAI Press, 2020.

[33] Octavian Ganea, Gary Becigneul, and Thomas Hofmann. Hyperbolic entailment cones for learning hierarchical embeddings. In *International Conference on Machine Learning*, pages 1646–1655, 2018.

[34] Kelvin Guu, John Miller, and Percy Liang. Traversing knowledge graphs in vector space. In Lluís Màrquez, Chris Callison-Burch, Jian Su, Daniele Pighin, and Yuval Marton, editors,

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015, pages 318–327. The Association for Computational Linguistics, 2015.

[35] William L. Hamilton, Payal Bajaj, Marinka Zitnik, Dan Jurafsky, and Jure Leskovec. Embedding logical queries on knowledge graphs. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 2030–2041, 2018.

[36] Junheng Hao, Chelsea Ju, Muhao Chen, Yizhou Sun, Carlo Zaniolo, and Wei Wang. Biojoie: Joint representation learning of biological knowledge bases. In *Proceedings of the 11st ACM Conference on Bioinformics, Computational Biology and Biomedicine (BCB)*. ACM, 2020.

[37] Olaf Hartig and Ralf Heese. The sparql query graph model for query optimization. In *European Semantic Web Conference*, pages 564–578. Springer, 2007.

[38] He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. In Regina Barzilay and Min-Yen Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 1766–1776. Association for Computational Linguistics, 2017.

[39] Ruining He, Wang-Cheng Kang, and Julian J. McAuley. Translation-based recommendation. In Paolo Cremonesi, Francesco Ricci, Shlomo Berkovsky, and Alexander Tuzhilin, editors, *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017*, pages 161–169. ACM, 2017.

[40] Jiafeng Hu, Reynold Cheng, Zhipeng Huang, Yixang Fang, and Siqiang Luo. On embedding uncertain graphs. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, 2017.

[41] Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. Knowledge graph embedding based question answering. In J. Shane Culpepper, Alistair Moffat, Paul N. Bennett, and Kristina Lerman, editors, *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pages 105–113. ACM, 2019.

[42] Rodolphe Jenatton, Nicolas L Roux, Antoine Bordes, and Guillaume R Obozinski. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems (NIPS)*, 2012.

[43] Guoliang Ji, Shizhu He, Liheng Xu, Kang Liu, and Jun Zhao. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*, pages 687–696. The Association for Computer Linguistics, 2015.

[44] Yantao Jia, Yuanzhuo Wang, Hailun Lin, Xiaolong Jin, and Xueqi Cheng. Locally adaptive translation for knowledge graph embedding. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, 2016.

[45] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *arXiv preprint arXiv:1702.08734*, 2017.

[46] Rudolf Kadlec, Ondrej Bajgar, and Jan Kleindienst. Knowledge base completion: Baselines strike back. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*, 2017.

[47] A.K Katsaras and D.B Liu. Fuzzy vector spaces and fuzzy topological vector spaces. *Journal of Mathematical Analysis and Applications*, 58(1):135–146, 1977.

[48] Angelika Kimmig, Stephen Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. A short introduction to probabilistic soft logic. In *Proceedings of the NIPS Workshop on Probabilistic Programming: Foundations and Applications*, 2012.

[49] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2014.

[50] Erich-Peter Klement, Radko Mesiar, and Endre Pap. *Triangular Norms*, volume 8 of *Trends in Logic*. Springer, 2000.

[51] George Klir and Bo Yuan. *Fuzzy sets and fuzzy logic*, volume 4. Prentice hall New Jersey, 1995.

[52] Alice Lai and Julia Hockenmaier. Learning to predict denotational probabilities for modeling entailment. In *EACL*, 2017.

[53] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.

[54] Chengjiang Li, Yixin Cao, Lei Hou, Jiaxin Shi, Juanzi Li, and Tat-Seng Chua. Semi-supervised entity alignment via joint knowledge embedding model and cross-graph model. 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 2723–2732. Association for Computational Linguistics, 2019.

[55] Hang Li, Tie-Yan Liu, and Chengxiang Zhai. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 2009.

[56] Xiang Li, Luke Vilnis, Dongxu Zhang, Michael Boratko, and Andrew McCallum. Smoothing the geometry of probabilistic box embeddings. *ICLR*, 2019.

[57] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, 2015.

[58] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

[59] Thomas Lukasiewicz and Umberto Straccia. Managing uncertainty and vagueness in description logics for the semantic web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2008.

[60] Tom Mitchell, William Cohen, Estevam Hruschka, Partha Talukdar, B Yang, J Betteridge, A Carlson, B Dalvi, M Gardner, B Kisiel, et al. Never-ending learning. *Communications of the ACM*, 2018.

[61] Maximilian Nickel, Lorenzo Rosasco, Tomaso A Poggio, et al. Holographic embeddings of knowledge graphs. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pages 1955–1961. AAAI Press, 2016.

[62] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. A three-way model for collective learning on multi-relational data. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 809–816. Omnipress, 2011.

[63] Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 6338–6347. Curran Associates, Inc., 2017.

[64] Dhruvesh Patel, Shib Sankar Dasgupta, Michael Boratko, Xiang Li, Luke Vilnis, and Andrew McCallum. Representing joint hierarchies with box embeddings. *Automated Knowledge Base Construction*, 2020.

[65] Shichao Pei, Lu Yu, Robert Hoehndorf, et al. semi-supervised entity alignment via knowledge graph embedding with awareness of degree difference. In *Proceedings of the Web Confererence (WWW)*, pages 3130–3136. ACM, 2019.

[66] Jay Pujara, Eriq Augustine, and Lise Getoor. Sparsity and noise: Where knowledge graph embeddings fall short. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

[67] Lev Ratinov and Dan Roth. Design challenges and misconceptions in named entity recognition. In *Proceedings of the SIGNLL Conference on Computational Natural Language Learning (CoNLL)*, 2009.

[68] Thomas Rebele, Fabian Suchanek, Johannes Hoffart, Joanna Biega, Erdal Kuzey, and Gerhard Weikum. Yago: A multilingual knowledge base from wikipedia, wordnet, and geonames. In *Proceedings of the International Semantic Web Conference (ISWC)*, volume 9982 of *Lecture Notes in Computer Science*, pages 177–185. Springer, 2016.

[69] Hongyu Ren, Weihua Hu, and Jure Leskovec. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[70] Hongyu Ren, Weihua Hu, and Jure Leskovec. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *8th International Conference on Learning Representations*. OpenReview.net, 2020.

[71] Hongyu Ren and Jure Leskovec. Beta embeddings for multi-hop logical reasoning in knowledge graphs. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[72] Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer, 2018.

[73] Michael Schmidt, Michael Meier, and Georg Lausen. Foundations of sparql query optimization. In *Proceedings of the 13th International Conference on Database Theory*, pages 4–33, 2010.

[74] Jiaming Shen, Zeqiu Wu, Dongming Lei, Jingbo Shang, Xiang Ren, and Jiawei Han. Setexpan: Corpus-based set expansion via context feature selection and rank ensemble. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, volume 10534 of *Lecture Notes in Computer Science*, pages 288–304. Springer, 2017.

[75] Baoxu Shi and Tim Weninger. Proje: Embedding projection for knowledge graph completion. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[76] Anshumali Shrivastava and Ping Li. Asymmetric LSH (ALSH) for sublinear time maximum inner product search (MIPS). In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2321–2329, 2014.

[77] Richard Socher, Danqi Chen, Christopher D Manning, and Andrew Ng. Reasoning with neural tensor networks for knowledge base completion. In *Advances in Neural Information Processing Systems (NIPS)*, 2013.

[78] DN Sosa, A Derry, M Guo, E Wei, C Brinton, and RB Altman. A literature-based knowledge graph embedding method for identifying drug repurposing opportunities in rare diseases. In *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, volume 25, pages 463–474, 2020.

[79] Robert Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[80] Fabian M Suchanek, Serge Abiteboul, et al. Paris: Probabilistic alignment of relations, instances, and schema. *Proceedings of the VLDB Endowment (PVLDB)*, 5(3), 2011.

[81] Haitian Sun, Andrew O. Arnold, Tania Bedrax-Weiss, Fernando Pereira, and William W. Cohen. Faithful embeddings for knowledge base queries. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[82] Zequn Sun, Muhao Chen, Wei Hu, Chengming Wang, Jian Dai, and Wei Zhang. Knowledge association with hyperbolic knowledge graph embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5704–5716, 2020.

[83] Zequn Sun, Wei Hu, and Chengkai Li. Cross-lingual entity alignment via joint attribute-preserving embedding. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 628–644. Springer, 2017.

[84] Zequn Sun, Wei Hu, Qingheng Zhang, and Yuzhong Qu. Bootstrapping entity alignment with knowledge graph embedding. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 4396–4402. International Joint Conferences on Artificial Intelligence Organization, 2018.

[85] Zequn Sun, Chengming Wang, Wei Hu, Muhao Chen, Jian Dai, Wei Zhang, and Yuzhong Qu. Knowledge graph alignment network with gated multi-hop neighborhood aggregation. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pages 222–229. AAAI Press, 2020.

[86] Zequn Sun, Jiacheng Huang Wang, Wei Hu, Muhao Chen, and Yuzhong Qu. Transedge: Translating relation-contextualized embeddings for knowledge graphs. In *Proceedings of the International Semantic Web Conference (ISWC)*, pages 612–629. Springer, 2019.

[87] Zequn Sun, Qingheng Zhang, Wei Hu, Chengming Wang, Muhao Chen, Farahnaz Akrami, and Chengkai Li. A benchmarking study of embedding-based entity alignment for knowledge graphs. *Proceedings of the VLDB Endowment*, 13:2326–2340, 2020.

[88] Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *International Conference on Learning Representations (ICLR)*, 2019.

[89] Damian Szklarczyk, John H Morris, Helen Cook, Michael Kuhn, Stefan Wyder, Milan Simonovic, Alberto Santos, Nadezhda T Doncheva, Alexander Roth, Peer Bork, et al. The string database in 2017: quality-controlled protein–protein association networks, made broadly accessible. *Nucleic acids research*, 2016.

[90] Kristina Toutanova and Danqi Chen. 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, 2015.

[91] Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. Complex embeddings for simple link prediction. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 48, pages 2071–2080. PMLR, 2016.

[92] Bayu Distiawan Trsedya, Jianzhong Qi, and Rui Zhang. Entity alignment between knowledge graphs using attribute embeddings. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, pages 297–304. AAAI Press, 2019.

[93] Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. Order-embeddings of images and language. In *ICLR*, 2016.

[94] Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. Probabilistic embedding of knowledge graphs with box lattice measures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*, pages 263–272. Association for Computational Linguistics, 2018.

[95] Denny Vrandečić and Markus Krötzsch. Wikidata: a free collaborative knowledge base. *Communications of ACM*, 57(10):78–85, 2014.

[96] Haoyu Wang, Muhao Chen, Hongming Zhang, and Dan Roth. Joint constrained learning for event-event relation extraction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, 2020.

[97] Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743, 2017.

[98] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of AAAI Conference on Artificial Intelligence (AAAI)*, pages 1112–1119. AAAI Press, 2014.

[99] Zhichun Wang, Qingsong Lv, Xiaohan Lan, and Yu Zhang. Cross-lingual knowledge graph alignment via graph convolutional networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 349–357. Association for Computational Linguistics, 2018.

[100] Zhongyuan Wang and Haixun Wang. Understanding short texts. In *ACL*, 2016.

[101] Zhongyuan Wang, Haixun Wang, Ji-Rong Wen, and Yanghua Xiao. An inference approach to basic level of categorization. In *Proceedings of the ACM International Conference on Information and Knowledge Management (CIKM)*, 2015.

[102] David H Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[103] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q Zhu. Probase: A probabilistic taxonomy for text understanding. In *Proceedings of ACM SIGMOD International Conference on Management of Data (SIGMOD)*, 2012.

[104] Wenhan Xiong, Thien Hoang, and William Yang Wang. Deeppath: A reinforcement learning method for knowledge graph reasoning. In Martha Palmer, Rebecca Hwa, and Sebastian Riedel, editors, *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pages 564–573. Association for Computational Linguistics, 2017.

[105] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. Embedding entities and relations for learning and inference in knowledge bases. *International Conference on Learning Representations (ICLR)*, 2015.

[106] Hsiu-Wei Yang, Yanyan Zou, Peng Shi, Wei Lu, Jimmy Lin, and Xu Sun. Aligning cross-lingual entities with multi-aspect information. 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 4430–4440. Association for Computational Linguistics, 2019.

[107] Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. Question answering using enhanced lexical semantic models. In *Proceedings of the Annual Meeting of Associations for Computational Linguistics (ACL)*, 2013.

[108] Qingheng Zhang, Zequn Sun, Wei Hu, Muhao Chen, Lingbing Guo, and Yuzhong Qu. Multi-view knowledge graph embedding for entity alignment. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pages 5429–5435. International Joint Conferences on Artificial Intelligence Organization, 2019.

[109] Tianran Zhang, Muhao Chen, and Alex Bui. Diagnostic prediction with sequence-of-sets representation learning for clinical event. In *Proceedings of the 18th International Conference on Artificial Intelligence in Medicine (AIME)*. Springer, 2020.

[110] Hans-Jürgen Zimmermann. *Fuzzy Set Theory - and Its Applications*. Springer, 1991.

[111] Lei Zou, Jinghui Mo, Lei Chen, M. Tamer Özsu, and Dongyan Zhao. Gstore: Answering sparql queries via subgraph matching. *Proc. VLDB Endow.*, 4(8):482–493, May 2011.