

C02029: Doctor of Philosophy

CRICOS Code: 009469A

Subject Code: 33874

August 2019

Network Embedding Learning in Knowledge Graph

Zili Zhou

School of Computer Science

Faculty of Engineering and Information Technology

University of Technology Sydney

NSW - 2007, Australia

Network Embedding Learning in Knowledge Graph

*A thesis submitted in partial fulfilment of the requirements
for the degree of*

Doctor of Philosophy
in
Analytics

by

Zili Zhou

to

School of Computer Science
Faculty of Engineering and Technology
University of Technology Sydney
NSW - 2007, Australia

August 2019

ABSTRACT

Knowledge Graph stores a large number of human knowledge facts in form of multi-relational network structure, is widely used as a core technique in real-world applications including search engine, question answering system, and recommender system. Knowledge Graph is used to provide extra info box for user query in Google search engine, the WolframAlpha site provides question answering service relying on Knowledge Graph, and the eBay uses Knowledge Graph as semantic enhance for their recommendation service.

Motivated by several characteristics of Knowledge Graph including incompleteness, structural inferability, and semantical application enhancement, a few efforts have been put into the Knowledge Graph analysis area. Some works contribute to Knowledge Graph construction and maintenance through crowdsourcing. Some previous network embedding learning models show good performance on homogeneous network analysis, while the performance of directly using them on Knowledge Graph is limited because the multiple relationship information of the Knowledge Graph is ignored. Then, the concept of Knowledge Graph embedding learning is given, by learning representation for Knowledge Graph components including entities and relations, the latent semantic information is extracted into embedding representation. And the embedding techniques are also utilized in collaborative learning for Knowledge Graph and external application scenarios, the target is to use Knowledge Graph as a semantic enhancement to improve the performance of external applications.

However, some problems still remain in Knowledge Graph completion, reasoning, and external application. First, a proper model is required for Knowledge Graph self-completion, and a proper integration solution is also required to add extra conceptual taxonomy information into the process of Knowledge Graph completion. Then, a framework to use sub-structure information of Knowledge Graph network into knowledge reasoning is needed. After that, a collaborative learning framework for knowledge graph completion and downstream machine learning tasks is needed to be designed. In this thesis, we take recommender systems as an example of downstream machine learning tasks.

To address the aforementioned research problems, a few approaches are proposed in the works introduced in this thesis.

- A bipartite graph embedding based Knowledge Graph completion approach for Knowledge Graph self-completion, each knowledge fact is represented in the form of bipartite graph structure for more reasonable triple inference.

-
- An embedding based cross completion approach for completing the factual Knowledge Graph with additive conceptual taxonomy information, the components of factual Knowledge Graph and conceptual taxonomy, entities, relations, types, are jointly represented by embedding representation.
 - Two sub-structure based Knowledge Graph transitive relation embedding approaches for knowledge reasoning analysis based on Knowledge Graph sub-structure, the transitive structural information contained in Knowledge Graph network sub-structure is learned into relation embedding.
 - Two hierarchical collaborative embedding approaches for proper collaborative learning on Knowledge Graph and Recommender System through linking Knowledge Graph entities with Recommender items, then entities, relations, items, and users are represented by embedding in collaborative space.

The main contributions of this thesis are proposing a few approaches which can be used in multiple Knowledge Graph related domains, Knowledge Graph completion, reasoning and application. Two approaches achieve more accurate Knowledge Graph completion, other two approaches model knowledge reasoning based on network sub-structure analysis, and the other approaches apply Knowledge Graph into a recommender system application.

CERTIFICATE OF ORIGINAL AUTHORSHIP

I, Zili Zhou declare that this thesis, is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the School of Computer Science at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise reference or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution. I certify that the work in this thesis has not previously been submitted for a degree nor has it been submitted as part of the requirements for a degree at any other academic institution except as fully acknowledged within the text. This thesis is the result of a Collaborative Doctoral Research Degree program with Shanghai University.

This research is supported by the Australian Government Research Training Program.

Production Note:

SIGNATURE: Signature removed prior to publication.

Zili Zhou

DATE: 21st August, 2019

DEDICATION

To my beloved wife ...

ACKNOWLEDGMENTS

First of all, I would like to thank Prof Guandong Xu, my supervisor, for his guidance, suggestion, and support throughout my doctoral program course at the University of Technology Sydney. Without his professional guidance and support, this work would not have been achieved.

I am also grateful to my co-supervisor Prof Jinyan Li for his constructive suggestions on my research works. I would like to pass my gratitude to my research mate and friend Dr. Shaowu Liu, for his help and suggestions on my works.

I am thankful to my beloved wife, Xiu Yan, and my baby sons, Yinuo Zhou and Yiheng Zhou, for their love and encouragement which helps me finish this thesis. I am also thankful to my parents and my parents in law for their support throughout my research studies.

Zili Zhou
Sydney, Australia
August, 2019

LIST OF PUBLICATIONS

RELATED TO THE THESIS :

1. Zhou, Z., Xu, G., Zhu, W., Li, J., & Zhang, W. (2017, May). Structure embedding for knowledge base completion and analytics. In 2017 International Joint Conference on Neural Networks (IJCNN) (pp. 737-743). IEEE.
2. Zhou, Z., Liu, S., Xu, G., Xie, X., Yin, J., Li, Y., & Zhang, W. (2018, June). Knowledge-Based Recommendation with Hierarchical Collaborative Embedding. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 222-234). Springer, Cham.
3. Zhou, Z., Liu, S., Xu, G., & Zhang, W. (2019, July). On Completing Sparse Knowledge Base with Transitive Relation Embedding. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, pp. 3125-3132).
4. Zhou, Z., Liu, S., Xu, G. & Zhang, W. Meta-structure Transitive Relation Embedding for Knowledge Graph Completion. Prepared to be submitted as a Conference Paper.
5. Zhou, Z., Liu, S., Xu, G. & Zhang, W. Cross Completion for Factual Knowledge Graph and Conceptual Taxonomy. Prepared to be submitted as a Journal Paper.
6. Zhou, Z., Liu, S., Xu, G. & Zhang, W. Knowledge Graph based Collaborative Network Embedding for Recommender System. Prepared to be submitted as a Journal Paper.

OTHERS :

7. Zhou, Z., Xu, G., Zhu, X., & Liu, S. (2017, October). Latent factor analysis for low-dimensional implicit preference prediction. In 2017 International Conference on Behavioral, Economic, Socio-cultural Computing (BESC) (pp. 1-2). IEEE.

-
8. Liu, S., Xu, G., Zhu, X., & Zhou, Z. (2017, October). Towards simplified insurance application via sparse questionnaire optimization. In 2017 International Conference on Behavioral, Economic, Socio-Cultural Computing (BESC) (pp. 1-2). IEEE.
 9. Yin, J., Zhou, Z., Liu, S., Wu, Z., & Xu, G. (2018, June). Social Spammer Detection: A Multi-Relational Embedding Approach. In Pacific-Asia Conference on Knowledge Discovery and Data Mining (pp. 615-627). Springer, Cham.

TABLE OF CONTENTS

List of Publications	ix
List of Figures	xv
List of Tables	xvii
1 Introduction	1
1.1 Motivation	2
1.2 Research Objectives	4
1.3 Research Problems	4
1.4 Proposed Approaches	6
1.5 Thesis Outlines	10
2 Literature Review	13
2.1 Knowledge Graph and Semantic Web	14
2.2 Network Embedding Representation Learning	17
2.3 Knowledge Graph Embedding	19
2.3.1 Translational Distance Models	22
2.3.2 Semantic Matching Models	22
2.3.3 Relation-based Models	24
2.4 Knowledge Graph based Application Systems	24
3 Bipartite Graph Embedding based Knowledge Graph Completion	27
3.1 Introduction	27
3.2 Preliminaries	29
3.2.1 Structured Embedding (SE) and TransE	29
3.2.2 Other Models	29
3.3 Methodology	30
3.3.1 Structured Embedding Framework	30

TABLE OF CONTENTS

3.3.2	Bipartite Graph Network Architecture	32
3.3.3	Training	34
3.3.4	Latent Information	36
3.4	Experiment and Analysis	37
3.5	Summary	38
4	Embedding based Cross Completion for Factual Knowledge Base and Conceptual Taxonomy	39
4.1	Introduction	39
4.2	Preliminaries	41
4.2.1	Factual Knowledge Graph and Conceptual Taxonomy	41
4.2.2	Structure Embedding based Factual Knowledge Graph Completion	42
4.2.3	Collaborative Filtering based Conceptual Taxonomy Completion .	42
4.2.4	Problem Definition	43
4.3	Methodology	43
4.3.1	Integration of Factual Knowledge Graph and Conceptual Taxonomy	43
4.3.2	Collaborative Embedding Model	44
4.3.3	Adaptive True Knowledge Prediction	47
4.4	Experiment and Analysis	49
4.4.1	Dataset	49
4.4.2	Baselines	49
4.4.3	Comparison	50
4.5	Summary	51
5	Sub-structure based Knowledge Graph Transitive Relation Embedding	53
5.1	Introduction	53
5.2	Preliminaries	57
5.2.1	Knowledge Graph Completion	57
5.2.2	Embedding-based Knowledge Graph Completion	57
5.2.3	Relation-based Knowledge Graph Completion	58
5.2.4	Limitations of Current Models	58
5.3	On Completing Sparse Knowledge Base with Transitive Relation Embedding	58
5.3.1	Triangle Pattern	59
5.3.2	Transitive Relation Embedding	61
5.3.3	Training	62
5.3.4	Joint Prediction Strategy	63

5.3.5	Advantages of Proposed Model	64
5.4	Meta-structure based Transitive Relation Embedding for Knowledge Graph Completion	65
5.4.1	Meta-structure	66
5.4.2	Transitive Relation Embedding	67
5.4.3	Training and Prediction	69
5.4.4	Joint Prediction Strategy	71
5.4.5	Advantages of Proposed Model	72
5.5	Experiment and Analysis	73
5.5.1	Entity Link Prediction	75
5.5.2	Relation Link Prediction	77
5.5.3	Accurate Prediction on Extremely Sparse KG with Transitive Relation Embedding	81
5.6	Summary	82
6	Hierarchical Collaborative Embedding for Knowledge Graph and Recommender System	85
6.1	Introduction	85
6.2	Preliminaries	90
6.2.1	Implicit Feedback Recommendation	90
6.2.2	Collaborative Filtering using Implicit Feedback	91
6.2.3	Knowledge Graph	91
6.3	Hierarchical Collaborative Embedding	92
6.3.1	Knowledge Graph Structured Embedding	92
6.3.2	Knowledge Conceptual Level Connection	93
6.3.3	Collaborative Learning	95
6.4	Collaborative Network Embedding with Different Measurement Scale . .	95
6.4.1	Proximity Probability Embedding on RS Bipartite Network	97
6.4.2	Proximity Probability Embedding on KG Multi-relational Network	97
6.4.3	Collaborative Network Embedding	98
6.4.4	Advantages of Proposed Model	99
6.5	Experiment and Analysis	100
6.5.1	Dataset	100
6.5.2	Baselines	102
6.5.3	Comparison	103

TABLE OF CONTENTS

6.6	Summary	105
7	Conclusion and Future Work	107
7.1	Contributions	107
7.2	Possible Future Work	110
	Bibliography	111

LIST OF FIGURES

FIGURE	Page
1.1 Approach Framework Structure	7
3.1 Bipartite Graph Network Architecture	32
3.2 The process of getting one unit value of predict right entity vector by using weight sum of all units in left entity vector, all the units in predict right entity vector are computed as the process described in figure.	33
3.3 The process of getting one unit value of predict left entity vector by using weight sum of all units in right entity vector, all the units in predict left entity vector are computed as the process described in figure.	34
4.1 Cross Completion System for Factual Knowledge Graph and Conceptual Taxonomy	41
4.2 Completing Factual Knowledge Graph with information in Conceptual Taxonomy	45
4.3 Completing Conceptual Taxonomy with information in Factual Knowledge Graph	45
5.1 Basic Idea of Triangle Pattern	54
5.2 Example of Meta-structure	56
5.3 Meta-structure	56
5.4 Triangle Pattern with Formulated Relation Definition	59
5.5 Triangle Relation Pattern in Meta-structure	67
5.6 Hit@10 Result Comparison between MSTRE and MSTRE+Baselines	80
5.7 MRR Result Comparison between MSTRE and MSTRE+Baselines	80
6.1 Conceptual Level of Hierarchical Collaborative Embedding	87
6.2 Structure of Collaborative Network Embedding	88

LIST OF TABLES

TABLE	Page
2.1 List of popular Knowledge Graph Embedding methods	21
3.1 Ranking results on FB200 (146 entities)	37
3.2 Ranking results on FB500 (485 entities)	37
4.1 Precision, Recall and F1-score results of Cross Completion on Factual Knowl- edge Graph and Conceptual Taxonomy	49
5.1 Relation Inference Example	63
5.2 Space and Time Complexity of TRE and baselines	64
5.3 Space and Time Complexity of MSTRE and baselines	72
5.4 Dataset Size of FB15k, WN18 and DBP	74
5.5 Result of FB15K Entity Prediction with TRE	76
5.6 Result of WN18 Entity Prediction with TRE	76
5.7 Result of FB15K Entity Prediction with MSTRE	76
5.8 Result of FB15K, WN18 and DBP Relation Prediction with TRE $(h,?,t)$	78
5.9 Result of FB15K, WN18 and DBpedia Relation Prediction with MSTRE $(h,?,t)$	78
5.10 Result of FB15K, WN18 and DBpedia Relation Prediction $(h,?,t)$ for Entity Pairs with MSTRE	81
5.11 Result of Sparse FB15K Relation Prediction with TRE	82
6.1 Dataset Size of Movielens and Book-crossing	101
6.2 Mean MAP and Recall results of Hierarchical Collaborative Embedding on Github Dataset	104
6.3 Precision and Recall results of Collaborative Network Embedding on Movie- lens Dataset	104
6.4 Precision and Recall results of Collaborative Network Embedding on BookCross- ing Dataset	105

INTRODUCTION

Knowledge Graph is a multi-relational network structure system constructed for large-scale real-world human knowledge storage, retrieval, and external application. With the multi-relational network structure, there is a large number of relationship types contained in Knowledge Graph. Nowadays, the Knowledge Graph has been used as a key technique in various application scenarios. Google Knowledge Graph¹ provides extra entity infobox related to a user query in Google search engine service. WolframAlpha² online Question Answering system is built based on Knowledge Graph. Another example is that eBay improves its recommendation service based on explainable reasoning through Knowledge Graph [89].

The data resource of Knowledge Graph is automatically extracted from open sources including books, papers, articles, websites. This means the number of relationship types is not predefined and the relationships are always being updated. The multi-relationship of Knowledge Graph brings new challenges for Knowledge Graph analysis, it also makes the research on Knowledge Graph more attractive, because with this kind of automatically extracted structured human knowledge, we have an opportunity to reveal the human knowledge reasoning patterns with analysis methodologies. As a result of Knowledge Graph analysis, Knowledge Graph can be used as a semantic enhancement for external application scenarios, such as Recommender System, Knowledge Graph is proved useful to improve the accuracy and explainability of Recommender System.

¹googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html

²www.wolframalpha.com

A lot of efforts have been put into the analytics of Knowledge Graph, researchers intended to seek an efficient model for automatic human knowledge reasoning by utilizing the Multi-relational network structure of Knowledge Graph. Network Embedding Learning has been acknowledged as an effective type of models for leveraging network structure into network analytics. For a multi-relational network like Knowledge Graph, multi-relation network embedding methods have also been proved useful for human knowledge semantic inference through the structure of knowledge graph.

On the other hand, beyond the self-completion task of Knowledge Graph, network embedding methods are also used as a key technique in heterogeneous network analysis task, latent semantic information of different type elements of the heterogeneous network can be collaboratively learned with proper designed embedding models.

This thesis focuses on studying network embedding models to achieve Knowledge Graph completion tasks and Recommender System improvement.

1.1 Motivation

Knowledge Graph based Systems aim to extract semantic reasoning information of human knowledge from Knowledge Graph special network structure for two main reasons, 1) appending more knowledge facts into current Knowledge Graph based on network structure semantic reasoning, 2) improving external system performance based on semantic enhancement information provided by Knowledge Graph.

Despite the potential usefulness of Knowledge Graph based Systems, there are still following several issues in this field,

- **Knowledge Graph incompleteness.** Current Knowledge Graphs are incomplete, there are missing knowledge facts which have not been collected. Although a huge amount of human knowledge facts have been collected from multiple open resources, such as Wikipedia infobox, Websites, Articles, Papers, existed Knowledge Graphs are still incomplete. Only a part of human knowledge can be collected because all human knowledge is an almost infinite set, the knowledge facts uncollected are considered missing for Knowledge Graph. While the knowledge facts stored in current Knowledge Graph can be used to learn knowledge reasoning patterns, missing knowledge facts can be inferred by patterns to complete the Knowledge Graph.

This motivates a few works in this thesis, models are designed in these works aiming to learn knowledge reasoning patterns for Knowledge Graph Completion. More specifically, the knowledge facts can be considered as links in Knowledge Graph, Knowledge Graph completion is a task to predict missing links based existed links in current Knowledge Graph. To get higher quality Knowledge Graphs with more correct knowledge facts contained, an automatic Knowledge Graph Completion solution is needed. Knowledge reasoning patterns should be learned from the knowledge facts contained in current Knowledge Graph to infer missing knowledge facts automatically.

Although there are some existing method for KG completion, important features of knowledge reasoning process are ignored, first, the dual-way reasoning should be considered for a fact triple, forward from head entity to tail entity, and backward from tail entity to head entity, then, the conceptual taxonomy containing type information of entities can provide clues for knowledge reasoning. In this thesis, one work is proposed to achieve dual-way reasoning in KG completion process, another work focuses on importing the conceptual taxonomy into factual based KG completion, both two works target at better completion performance.

- **Structural inferability.** Knowledge Graph is multi-relational network, because of multiple relation types, relational structural inferable information contained in Knowledge Graph can be used for knowledge inference. To analyze the inference mechanism of Knowledge Graph, the sub-structures of Knowledge Graph can be extracted and they can be used to represent structural inferable information of Knowledge Graph. While no thorough solution for sub-structure analysis is available on either Knowledge Graph or other types of the multi-relational network before some of the works in this thesis.

In some works in this thesis, a series of multi-relational network sub-structure analysis solutions are proposed to fill the blank of this area, the proposed methods are used for knowledge reasoning and completion on Knowledge Graph.

- **Semantic enhancement.** Knowledge Graph can be used as a semantic enhancement for Recommender System to improve several performances of Recommender System, such as accuracy, recall, explainability, etc. We can turn Knowledge Graph and Recommender System into network representation, but in most cases, the network structures of Knowledge Graph and Recommender System are not exactly same. Knowledge Graph is in multi-relational network structure, while Recom-

mender System may be represented in other structures, such as homogeneous network, a bipartite network or heterogeneous network with a limited number of predefined relations.

In this case, a collaborative learning solution is needed for being adapted with both Knowledge Graph and Recommender System, in the meantime, the semantic information of both sides should be adequately employed. In some works of this thesis, collaborative learning solutions are proposed to learn knowledge reasoning patterns and Recommender System required patterns jointly. The proposed model can adopt different structures of Knowledge Graph and Recommender System, it can also adequately employ useful semantic information on both sides.

1.2 Research Objectives

The goal of the thesis is to improve the effectiveness of both Knowledge Graph self completion and Recommender System by studying new methods and techniques for semantic information representation in Knowledge Graph based systems.

This thesis aims to achieve the following objectives,

- **Knowledge Graph completion based on KG latent semantic information.**
Developing an approach to better adapt the multi-relational structure of Knowledge Graph for predicting missing knowledge facts more accurately.
- **Knowledge reasoning through inferable network structural information.**
Proposing a method to analyze the sub-structure in the multi-relational network for more accurate and explainable knowledge reasoning on Knowledge Graph.
- **Inferable knowledge semantic enhancement for Recommender System.**
With Knowledge Graph acts as semantic enhancement of Recommender System, providing a solution to extract semantic information from joint heterogeneous network consists of Knowledge Graph and Recommender System for improving performance of Recommender System.

1.3 Research Problems

Knowledge Graph is in multi-relational network structure, Knowledge Graph completion essentially is a task to predict missing links in the multi-relational network based on

links in the current network. There are the following problems in this task.

- The nodes in the multi-relational network are connected to each other by multi-relational links, the links are directed and have multiple types, each link represents a knowledge fact. **How to properly model multi-relational network structure Knowledge Graph?**
- Except the factual information represented by multi-relational links, conceptual information is also useful for knowledge completion as side information. Specifically, the conceptual information indicates the category of entities. The structure of conceptual information is different from factual information. **How to properly integrate conceptual information into Knowledge Graph completion?**

The multi-relational network contains complex sub-structures which can help to achieve accurate and explainable knowledge reasoning. If the sub-structures are considered into knowledge reasoning analysis, actually large numbers of sub-structures with a variety of complex structures can be extracted from the multi-relational network. Analyzing the sub-structures of multi-relational network raises new problems.

- It is challenging to find the sub-structure shape which is both simple to be modelled and effective for knowledge reasoning. **How to define modelable sub-structure for knowledge reasoning?**
- Despite some knowledge facts can be effectively inferred by simple sub-structure, some other knowledge facts can only be inferred through complex network sub-structure reasoning. **How to compose complex sub-structure for complex knowledge reasoning?**

Using Knowledge Graph as semantic enhancement side information for Recommender System is an effective solution for Recommender System performance improvement. However, integrating the Knowledge Graph and Recommender System into the joint heterogeneous network, the inhomogeneity between Knowledge Graph multi-relational network and Recommender System network causes the following problems.

- The network structure of the Knowledge Graph is different from the Recommender System network, it is a challenge to give a collaborative learning solution for adequately employing semantic information in both Knowledge Graph and Recommender System. **How to collaboratively learn latent semantic information**

of components in an integrated system consisting of Knowledge Graph and Recommender System?

- As the prediction measurement of knowledge graph and recommender systems have different scales, simple integrating knowledge graph and recommender system limits the performance of the integrated system. A proper method to unique the measurement scale of both sides is required. **How to unify the different measurements in Knowledge Graph and Recommender System for collaborative embedding learning?**

1.4 Proposed Approaches

As shown in Fig. 1.1, to address the research problems mentioned in the last section, in works of this thesis, based on network embedding representation learning methods, we propose the following categories of methods and techniques.

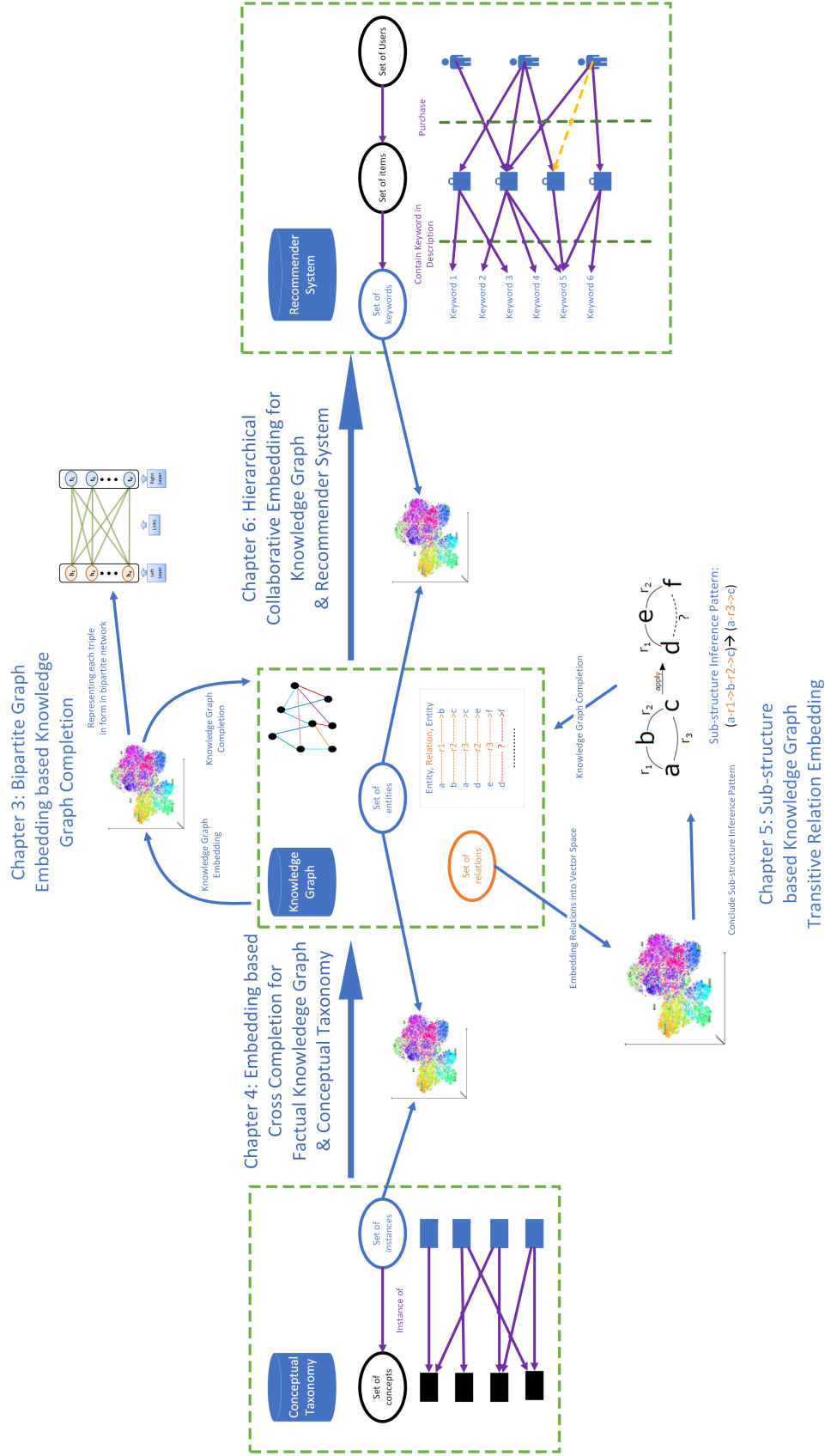


Figure 1.1: A framework structure demonstration for all approaches in this thesis.

Bipartite graph embedding based Knowledge Graph self completion. First, a bipartite network based multi-relational embedding learning method is designed to learn the proper representation for nodes and links in Knowledge Graph by considering each triplet of the Knowledge Graph as a special bipartite network, as a result, the designed method can achieve reasonably and accurate missing link prediction. The proposed method improves the current KB embedding methods by using a bipartite graph network model, which is widely used in many fields including image data compression, collaborative filtering.

The bipartite network based multi-relational embedding learning the model uses one matrix for each relation, the relation transform between two entities can be done directly by forward and backward propagation of bipartite graph network, no need for subspace projection. Because of using the bipartite graph network, the relation transforms between entities are nonlinear (network layer propagation), the multiple relations match or multiple entities match problems can be dealt with. The learned entity and relation embeddings can be used for problems such as knowledge base completion.

Embedding based cross completion for factual Knowledge Graph and conceptual taxonomy. Then, to integrate entity conceptual categorize information into factual Knowledge Graph completion task, a collaborative embedding learning based cross completion solution for factual Knowledge Graph and conceptual taxonomy is proposed to improve the completion performance of both sides by complementing semantic information from each other.

Integrating multiple types of knowledge bases can gain extra knowledge information which can't be gathered from single base. The factual Knowledge Graph and conceptual taxonomy with heterogeneous structure are integrated based on their mutual entities. For accurate cross completion, a collaborative embedding learning method is design to jointly learn latent semantic information of both factual Knowledge Graph and conceptual taxonomy. The proposed solution makes large progress for solving the knowledge bases acquisition and alignment problem by improving self-completion accuracy of Knowledge Graph containing factual information through using conceptual information as side information

Sub-structure based Knowledge Graph Transitive Relation Embedding. After that, to take information from multi-relational network sub-structure into knowledge reasoning process and complete Knowledge Graph, a series multi-relational network sub-structure analysis method are studied. Taking advantage of embedding representation learning, embedding based sub-structure representation learning method are proposed

to extract latent semantic knowledge reasoning information contained in sub-structure for accurate and explainable Knowledge Graph completion.

Entity embedding required multi-relational network embedding learns vector representations of entities and relations for computing the plausibility of candidate knowledge pieces. Despite its effectiveness in most cases, the embedding-based approach can fail for infrequent entities. Recently, relation-based methods have emerged to address issues of embedding-based approach by looking at short paths and relation patterns. Nevertheless, relation-based methods often assume the predictions can be made with individual patterns, which is not necessarily true for complex reasoning. Two works are introduced in this thesis to propose new models based on sub-structure capturing for complex reasoning,

- One new model exploiting the entity-independent transitive relation patterns, namely Transitive Relation Embedding (TRE). The TRE model alleviates the sparsity problem for predicting on infrequent entities while enjoys the generalization power of embedding.
- The other model analyze complex sub-structure constructing by multiple triangle patterns, namely, Meta-structure based Transitive Relation Embedding (MSTRE). The MSTRE model takes advantage of relation-based approach for predicting infrequent entities while complex reasoning can also be made with meta-structure.

Hierarchical Collaborative Embedding for Knowledge Graph and Recommender System. This thesis also contains works focusing on collaborative network embedding learning, a type of methods learn embedding representation for nodes and links in a heterogeneous network consisting of both multi-relational network and network in other structure.

Data sparsity is a common issue in recommender systems, particularly collaborative filtering. In real recommendation scenarios, user preferences are often quantitatively sparse because of the application of nature. To address the issue, the Knowledge Graph is used as a semantic enhancement for Recommender System, the integrated system consisting of Recommender System and Knowledge Graph is a heterogeneous network, a collaborative semantic information learning mechanism is proposed to achieve collaborative learning on the integrated system. Specifically, an embedding based model is proposed, namely Hierarchical Collaborative Embedding (HCE), it leverages both network structure and text info embedded in knowledge bases to supplement traditional collaborative filtering. The HCE model jointly learns the latent representations from user

preferences, linkages between items and knowledge base, as well as the semantic representations from the knowledge base. Collaborative network embedding learning extracts semantic information from Knowledge Graph and Recommender System through their network structure, using Knowledge Graph as a semantic enhancement, it can improve the performance of Recommender System. Experiment results on GitHub dataset demonstrated that semantic information from knowledge base has been properly captured, resulting in improved recommendation performance.

The different measurements in the Knowledge Graph and recommender system also limit the performance of collaborative embedding learning of integrated system, an approach is also proposed in this thesis to unify the measurement for collaborative embedding learning.

1.5 Thesis Outlines

An overview outlines of this thesis is presented as follows.

Chapter 2. A literature review is given in this chapter to show the background of the Knowledge Graph embedding learning area. This chapter focuses on current issues in several fields including Knowledge Graph completion, Knowledge Graph reasoning, and Knowledge Graph based Recommender System. The solutions and techniques for the issues are proposed in the following chapters.

The main contributions of the thesis are introduced in Chapters 3, 4, 5 and 6. To achieve the defined objectives, the main contribution part of the thesis introduces 4 categories of approaches,

- bipartite graph embedding based Knowledge Graph self completion,
- embedding based cross completion for factual Knowledge Graph and conceptual taxonomy,
- sub-structure based Knowledge Graph transitive relation embedding,
- hierarchical collaborative embedding for Knowledge Graph and Recommender System.

Chapter 3 introduces work on Knowledge Graph self completion with Bipartite Graph Embedding model. This work takes advantage of Bipartite Graph Embedding model to produce more accurate self completion result for Knowledge Graph.

Chapter 4 presents a cross completion solution for factual Knowledge Graph and conceptual taxonomy. Taking complement semantic information from each other, both factual Knowledge Graph and conceptual taxonomy can obtain extra useful information for completion task.

Chapter 5 gives two works which use sub-structure analysis for knowledge reasoning. By modeling knowledge relation transitivity in form of sub-structures patterns including triangle pattern and meta-structure, the knowledge reasoning patterns between relations can be revealed. The learned knowledge reasoning patterns can be used to improve the result of Knowledge Graph completion task.

Chapter 6 proposes two works integrating Knowledge Graph with Recommender System, the embedding learning model is designed to collaboratively learn semantic information of both Knowledge Graph and Recommender System. One work focuses on collaborative embedding learning through flexible conceptual link level between Knowledge Graph and Recommender System. The other work focuses on collaborative embedding learning on for Knowledge Graph and Recommender System with different measurement scale.

Chapter 7 concludes this thesis, the contribution of the works are summarized and some future research directions are also listed.

LITERATURE REVIEW

Human Knowledge is a familiarity, awareness, or understanding of someone or something, such as facts, information, descriptions, or skills, which is acquired through experience or education by perceiving, discovering, or learning. Knowledge Graph is a multi-relational network which is used to store complex human knowledge into structured form. In Knowledge Graph, each knowledge fact is simplified into the form of entity-relation-entity triple, considering entities as nodes and relations as edges, human knowledge can be stored and linked by the multi-relational structure of Knowledge Graph.

Embedding Representation Learning is a good solution for latent semantic information extraction, which is proved useful in textual semantic information analysis, including Word2Vec. Inspired by Embedding Representation Learning, network embedding learning methods are proposed to learn latent semantic information of nodes by learning an embedding representation for each node in the graph, the semantic information is extracted from the structural connection of the network. However these models are designed for homogeneous network, rich semantic information contained in multi-relationship is ignored if network embedding models are directly applied to multi-relational network structure Knowledge Graph.

To fully leverage the rich semantic information of multi-relationship in the embedding learning process, Knowledge Graph embedding, a new type of representation learning models which takes the multi-relational structural information into the learning process, was proposed to achieve proper embedding for Knowledge Graph components, entities and relations.

Knowledge Graph can be used for semantic enhancement for several application systems, one typical application system which can get benefit from using Knowledge Graph is Recommender System. With extra semantic additional information from Knowledge Graph, the accuracy and explainability of recommendation result can be improved.

In the rest part of this section, a detailed literature review of each topic is given.

2.1 Knowledge Graph and Semantic Web

Knowledge graphs, structured storage of human knowledge, are a backbone of many information systems.[62] The idea of using formalized human knowledge to improve the performance of intelligent systems starts in the 1980s.[71] Recently, more and more research works focus on Linked Open Data[7], such as DBpedia¹ [39] and Google Knowledge Graph², which represents general world knowledge as graph structures.

Knowledge Graph is a type of Semantic Web, Semantic Web is a type of information web designed for semantic information storage and quick query, and this technique is widely employed in Knowledge Graph construction process, the huge amount of human knowledge is collected and stored into semantic web, the quick query can also be achieved based on RDF standard, rich semantic information can be extracted by analyzing the Semantic Web containing human knowledge. With human knowledge collected for constructing Knowledge Graph, Semantic Web represents human knowledge into structured data based on RDF standard³ [52]. In RDF standard, each knowledge fact is represented in form of triple (subject entity, relation, object entity), for example, a knowledge fact “Sydney is a city of Australia” can be represented in form of triple (“Sydney”, “is a city of”, “Australia”). A large number of such triples are collected, considering entities as nodes and relations as edges connecting nodes, a Linked Data [7] based semantic web can be constructed. Many semantic web datasets from different topical domains are interlinked by Linked Data.[73]

The term “Knowledge Graph” is widely used after Google used semantic web for knowledge representation in their Web Search System and named it “Knowledge Graph”. Nowadays, from a broader view, any structured knowledge vault represented in form of network structure is considered in the category of Knowledge Graph. Almost any kind of dataset following RDF standard are included in this definition, to specify the Knowledge

¹An online large-scale factual knowledge graph <http://wiki.dbpedia.org>

²Large scale linked open data knowledge graph constructed by Google <https://googleblog.blogspot.com/2012/05/introducing-knowledge-graph-things-not.html>

³RDF semantic web standard for structured knowledge representation <http://www.w3.org/RDF/>

Graphs which will be studied in this thesis, we add following restricts to the Knowledge Graph definition for narrowing down the range. Firstly, the Knowledge Graph describes general knowledge entities in the real world. Then, the Knowledge Graph should be a multi-relational network which has multiple types of relations for linking entities (such as special heterogeneous networks with only 2 or 3 types of relations). Besides that, the entities in the Knowledge Graph should be allowed to be arbitrarily interrelated with each other. We list some popular existing Knowledge Graphs in following.

Cyc [40] is one of the oldest knowledge graphs, which is curated and maintained by CyCorp Inc.⁴ OpenCyc is a publicly available reduced version of Cyc, 120,000 instances, 2.5 million knowledge facts and 19,000 relation types are collected in OpenCyc. OpenCyc can be linked to other Linked Open Data datasets through Semantic Web schema.

Freebase⁵ [8] is constructed through crowdsourcing, it is built as a public and editable knowledge graph. Many kinds of entities are contained in Freebase, such as persons, cities, movies, etc. The last version of Freebase contains 50 million entities, 3 billion knowledge facts, and 38,000 relation types.

Wikidata [83] is another editable knowledge graph depends on crowdsourcing, it is operated by Wikimedia foundation⁶. The rich metadata is included in Wikidata, such as the source and date of a knowledge fact. Wikidata contains 16 million entities, 66 million facts, and 1,600 relations.

DBpedia⁷ [39] is a knowledge graph extracted from semi-structured data in Wikipedia. The data resource is mainly constructed through key-value pairs in infoboxes contained in Wikipedia page, using the key as ontology and value as property. DBpedia 2015-04 contains 4.8 million entities, 176 million facts and 2,800 relation types.

YAGO⁸ [47, 78] is built from both the category system in Wikipedia and the lexical resource of WordNet [28, 53]. The infobox properties are manually mapped to a fixed set of attributes, YAGO aims at automatic knowledge extraction from various data sources. 4.6 million entities, 26 million facts, and 77 relations are collected into YAGO.

NELL [17] is a project named Never Ending Language Learning. NELL apply coupled supervised learning based textual analysis on large-scale web sites to extract new entities and relations. The learning process of NELL keeps running, and it still keeps extracting entities and relations from open source today. There are over 2 million entities 433,000

⁴<http://www.cyc.com/>

⁵<http://www.freebase.com>

⁶<http://wikimediafoundation.org/>

⁷<http://wiki.dbpedia.org>

⁸<https://www.mpi-inf.mpg.de/home/>

extracted relations in NELL, the NELL ontology defines 425 types of relations.

Google Knowledge Graph was introduced by Google in 2012. It mainly uses semi-structured data resources, including Wikipedia, structured markup on webpages, contents from the social network, etc. The detailed construction information of Google Knowledge Graph is not available publicly. According to the description in one previous work related to Google Knowledge Graph [26], 570 million entities, 18 billion facts, and 35,000 relation types are collected in the Google Knowledge Graph.

Microsoft Satori⁹ is a Knowledge Graph project of Microsoft. Similar to Google, the detailed construction information of Satori is not given by Microsoft. According to Microsoft's online document¹⁰, the Satori contains 300 million entities and 800 million facts.

Facebook Entities Graph is a Knowledge Graph built by Facebook. Many items appeared in the social network can be represented by entities, such as movies, books, bands, etc. By parsing textual information and linking entity to Wikipedia, the social network can be connected to the general Knowledge Graph.

The Knowledge Graph can be constructed by either manual curation of a small group of experts or automatic information extraction from open source. However, most of the recent Knowledge Graph studies focus on the latter. Manually curation of experts is limited by knowledge boundary of experts, while the automatic extracted Knowledge Graph contains a large scale of knowledge from open sources without such limitation. And the recent researches prefer to study multi-relational network structure Knowledge Graph, complex semantical knowledge reasoning can be analyzed based on interactions between multiple types of relations. Theoretically, the network with more than one type of relationship should be considered as a multi-relational network, there is no clear boundary defining the least number of relation for the multi-relational network, while the Knowledge Graph with more relation types is closer to a typical multi-relational network. A large number of relations types can be automatically collected into the Knowledge Graph by automatic information extraction, which makes the constructed Knowledge Graph closer to a typical multi-relational network.

⁹<https://blogs.bing.com/search/2013/03/21/understand-your-world-with-bing/>

¹⁰<http://research.microsoft.com/en-us/projects/trinity/query.aspx>

2.2 Network Embedding Representation Learning

The performance of machine learning models heavily relies on data representation, the features extracted from the original data and applied to the machine learning model. [6] Embedding learning is a typical representation learning, which is widely used in various of scenarios including text embedding[3, 50, 79], network embedding[29, 80], etc. A lot of application systems including Knowledge Graph can be represented in the form of network structure. Network based analysis can leverage latent information contained in network structure, while there are following challenges in network analysis.[23]

- First, the computational complexity of network analysis is quite high, with a large number of nodes and edges, traditional network feature extraction methods such as path enumeration or neighborhood node propagation are expensive in time cost.
- Second, the parallelizability is low, the nodes are densely linked with each other, distributing nodes in different shards or servers may cause expensive communication between shards or servers, which slow down the efficiency of parallel computing.
- Then, with traditional network feature extraction methods, it is hard to directly use the network data as input for machine learning models, which normally use vectors or matrices as input.

Network embedding learning is a good solution for large scale network analysis with bottleneck because of these challenges. The task of network embedding is to learn low-dimensional vector space representation for network nodes, learning embedding for each node. There are a few categories of network embedding methods.

Structure and Property Preserving Network Embedding is a type of methods which mapping components of the network into low dimensional embedding space based on network topology structural information. With this basic idea, several embedding models in this type are proposed. Following network structural information are used into network embedding process, which includes nodes and links [80], neighborhood structure [64], high-order proximities of nodes [84], and community structure [88]. Besides the structural information, some other network characteristics are also used. Link formation in the network is used for analysis based on network transitivity [35]. Structural balance property is used in the evolution of signed networks [18]. Some recent works also try to tackle inhomogeneity between network spaces [60, 87].

Network Embedding with Side Information is a type of methods which learn proper embedding for the network by leveraging rich side information. Node content or labels in information network are used in [82], and [96] leverages node and edge attributes from a social network into the analysis. Node types in the heterogeneous network are proved useful for network embedding [20]. Some works also do some exploration into new directions, such as multimodal and multisource fusion techniques [55, 96].

Advanced Information Preserving Network Embedding tries to provide end to end solution for specific application scenarios by designing framework of representation for the particular target scenarios.[41] One application scenarios is computer vision [99]. Another typical application is Natural Language Processing [97]. There are some other application scenarios, such as cascading prediction [41], anomaly detection [33], network alignment [48], collaboration prediction [21].

Besides the network embedding categories, some common used model in network embedding is also listed here.

Matrix Factorization The adjacency matrix is a common type of representation for network topology structure, matrix factorization can be used on the adjacency matrix to compress the representation of each node into a low-dimensional vector space, and the latent information of network topology can be extracted. Based on this idea, the network embedding task can be solved by matrix factorization techniques, there are two types of matrix factorization techniques which are widely used in this area, Singular Value Decomposition (SVD) [60] and Non-negative matrix factorization [88]. Singular Value Decomposition has the advantage in low-rank approximation optimality, and the advantage of non-negative matrix factorization is that it can be used as an additive model.

Random Walk method can be used in neighborhood structure based network embedding, the local structural characteristics of nodes are extracted from the original network structure. Word2Vector [51] is an effective word embedding model, the basic idea of Word2Vector is to map the target word into low-dimensional vector space, based on the neighborhood context, the words previous and after the target word. The assumption is that if the contexts of the two words are similar, the semantic meanings of these two words are similar, the embeddings should be close to each other. The network embedding problem can be solved with a similar idea, considering the nodes as words, the embedding of nodes can be learned based on the proper definition of neighborhood context. DeepWalk [64] and Node2Vec [29] use the random walk to generate paths from the original network, and then the paths are used as context to learn network embedding.

Deep Neural Networks is also a choice for network embedding. Some works, such as SDNE [84], SDAE [16], SiNE [87], try to fit deep models to network data, network structure and property-level constraints are applied on deep models. This type of methods can be used in some application scenarios including cascading prediction [41] and network alignment [48].

2.3 Knowledge Graph Embedding

Various models have been proposed in the area of network embedding, most of the models focus on the homogeneous network or simple heterogeneous network, However, directly using these models on the Knowledge Graph is not an effective solution, because the Knowledge Graph is a type of multi-relational network, and there are some extra challenges in multi-relational network embedding problem.

Given a multi-relational Knowledge Graph consisting of entities and relations, considering entities as the node of network and relations as edge connecting nodes. Because of multiple types of relations in Knowledge Graph, the structure of the Knowledge Graph is multi-relational. Each knowledge fact can be represented with a triple consisting of two nodes and one edge in the network. In this case, the knowledge facts stored in the Knowledge Graph can be represented by a collection of triples.

The basic idea of Knowledge Graph Embedding is similar to general Network Embedding, mapping the components into low dimensional vector space for new data representation. However, the Knowledge Graph Embedding methods consider two key points. Firstly, besides learning embedding on nodes, the embedding learning on edges should not be ignored because of rich semantic information contained in multiple relationships of Knowledge Graph. The other point is that the Knowledge Graph Embedding methods can simulate the knowledge reasoning process, the knowledge facts in each triple can be reasoned through embedding models.

A Knowledge Graph Embedding model design follows three steps. First, the embedding representation form should be defined for entities and relations, such as vector. Then, a score function should be defined to predict a score indicating the plausibility that a given knowledge fact is true. Finally, learning the proper embedding for entities and relations based on the assumption that the true knowledge fact should always have a higher score than false knowledge fact based on the defined score function.

Specifically, given a Knowledge Graph with a set of true knowledge facts, $\mathcal{D}^+$, the false knowledge fact set, $\mathcal{D}^-$, can be generated by negative sampling, then pairwise

ranking loss function is used for training. In each iteration of Knowledge Graph embedding training process, drawing a pair of triples $((h, r, t), (h', r', t'))$, the knowledge fact represented by (h, r, t) is true, and the knowledge fact represented by (h', r', t') is false, the target is to make the score function value of true fact higher than the false one. The proper embedding can be learned through minimizing the loss function defined in Eq. 2.1,

$$(2.1) \quad \min_{\theta} \sum_{(h,r,t) \in \mathcal{D}^+} \sum_{(h',r',t') \in \mathcal{D}^-} \max(0, \gamma - f_r(h, t) + f_{r'}(h', t'))$$

We list embedding representation form, score function and reference of several popular Knowledge Graph Embedding models in Tab. 2.1. These models can be categorized into two types, translational distance models and semantic matching models, detailed descriptions of these two types of models are given in the next two sections. After that, we introduce a new type of Knowledge Graph Embedding models, relation-based models, which gives a new direction for research in this area.

Table 2.1: List of popular Knowledge Graph Embedding methods

	Entity embedding	Relation embedding	Score function	Reference
TransE	$h, t \in \mathbb{R}^d$	$r \in \mathbb{R}^d$	$-\ h + r - t\ _{1/2}$	[12]
TransH	$h, t \in \mathbb{R}^d$	$r, w_r \in \mathbb{R}^d$	$-\ (h - w_r^T t w_r) + r - (t - w_r^T t w_r)\ _2^2$	[90]
TransR	$h, t \in \mathbb{R}^d$	$r \in \mathbb{R}^k, M_r \in \mathbb{R}^{k \times d}$	$-\ M_r h + r - M_r t\ _2^2$	[44]
TransD	$h, w_h, t, w_t \in \mathbb{R}^d$	$r, w_r \in \mathbb{R}^k$	$-\ (w_r w_h^T + I)h + r - (w_r w_t^T + I)t\ _2^2$	[36]
SE	$h, t \in \mathbb{R}^d$	$M_r^1, M_r^2 \in \mathbb{R}^{d \times d}$	$-\ M_r^1 h - M_r^2 t\ _1$	[13]
RESCAL	$h, t \in \mathbb{R}^d$	$M_r \in \mathbb{R}^{d \times d}$	$h^T M_r t$	[59]
DistMult	$h, t \in \mathbb{R}^d$	$r \in \mathbb{R}^d$	$h^T \text{diag}(r) t$	[95]
HolE	$h, t \in \mathbb{R}^d$	$r \in \mathbb{R}^d$	$r^T (h \star t)$	[58]
ComplEx	$h, t \in \mathbb{C}^d$	$r \in \mathbb{C}^d$	$\text{re}(h^T \text{diag}(r) \bar{t})$	[91]
SME	$h, t \in \mathbb{R}^d$	$r \in \mathbb{R}^d$	$(M_u^1 h + M_u^2 r + b_u)^T (M_v^1 t + M_v^2 r + b_v)$ $((M_u^1 h) \circ (M_u^2 r) + b_u)^T ((M_v^1 t) \circ (M_v^2 r) + b_v)$	[11]
NTN	$h, t \in \mathbb{R}^d$	$r, b_r \in \mathbb{R}^k, M_r \in \mathbb{R}^{d \times d \times k}$ $M_r^1, M_r^2 \in \mathbb{R}^{k \times d}$	$r^T \tanh(h^T M_r t + M_r^1 h + M_r^2 t + b_r)$	[77]
MLP	$h, t \in \mathbb{R}^d$	$r \in \mathbb{R}^d$	$w^T \tanh(M^1 h + M^2 r + M^3 t)$	[26]

2.3.1 Translational Distance Models

Translational distance model uses the translation in the vector space for score function definition, the components of Knowledge Graph are mapped into vector space, the embeddings of components are their position in vector space, the score function value of a knowledge fact is based on distance measurement in the vector space.

TransE [12] uses a simple but effective assumption to define its score function for objective learning in Knowledge Graph embedding framework. First, TransE model uses a vector to represent each entity and a vector to represent each relation, the entity vector and the relation vector have same dimension number, in a knowledge fact triple (h, r, t) , h , r and t , are subject entity, relation and object entity, and $\vec{h}$, $\vec{r}$, and $\vec{t}$ are their vector representation. After that, TransE model assumes that if a knowledge fact is true, then the triple of this fact, (h, r, t) , should satisfy the condition that $\vec{h} + \vec{r} \approx \vec{t}$. With a simple translation in vector space, TransE model assumes the true fact can fit such condition, reversely the false fact can't fit the condition. The score function can be designed based on distance, $-\|h + r - t\|_{1/2}$.

TransH [90], **TransR** [44], and **TransD** [36] are extended models of TransE model. They follow the basic idea of TransE, using vector space translation and distance measurement to classify true knowledge fact from false fact. However, these three models all argue that the semantic meaning of the entity may change when it is composed of different types of relations. In this case, before they compute the translation distance, they do a relation specific projection for each entity. TransH projects the entity into a relation specific hyperplane, TransR projects the entity into a relation specific subspace, and TransD is similar to TransR, but TransR projects the entity with relation specific projection matrix, TransD simplifies the projection matrix into the projection vector.

Structured Embedding [13] also relies on the idea of distance measurement and relation projection, while it uses two separate matrices for projection, M_r^1 is used to project subject entity h , M_r^2 is used to project object entity t . Then it assumes that if the fact is true, the projected positions of subject entity and object entity should be close to each other. The final score function is defined as $-\|M_r^1 h - M_r^2 t\|_1$.

2.3.2 Semantic Matching Models

With different assumption, the score function definition of the semantic matching model is different from the translational distance model. Instead of vector space translation distance, semantic matching models use the semantic matching score to define the score

function for objective learning in Knowledge Graph embedding framework.

RESCAL [59] uses a vector to represent each entity, and a matrix is used to represent each relation. Given a knowledge fact triple (h, r, t) , $\vec{h}, \vec{t}$ is the representations for entities h, t , M_r is the representation for relation r . the score function of RESCAL is defined as a bilinear function $f(h, r, t) = \vec{h}^T M_r \vec{t}$. The latent semantic information is contained in embedding representations of entities and relations, the bilinear function is designed to predict whether the entities and relations in a triple match each other semantically. The semantic matching score value is high if the knowledge fact is true, otherwise, the semantic matching score value is low.

DistMult [95], **HolE** [58], and **Complex** [91] share the idea that using semantic matching score as in RESCAL, but each model represent relation in different form based on different semantic matching assumption. DistMult model uses one diagonal matrix $diag(r)$ instead of normal matrix M_r to represent each relation, which simplify the parameters of model, the score function is defined as $f(h, r, t) = \vec{h}^T diag(r) \vec{t}$. HolE uses one vector to represent each relation, $\vec{r}$, it first composes the vector representations of entities by the circular correlation operation $\vec{h} \star \vec{t}$, then predict the semantic matching degree between relation vector and $\vec{h} \star \vec{t}$, $f(h, r, t) = \vec{r}^T (\vec{h} \star \vec{t})$. Complex model is similar to DistMult model, while it uses complex vector space for embedding representation instead of real vector space used in DistMult.

SME [11], **NTN** [77], and **MLP** [26] also use semantic matching score, while the semantic matching score is predicted through Neural Network structure, using knowledge fact triple as input and output semantic matching score. SME model consists of three layer, in middle layer of neural network, it propagates subject entity and relation into a new representation $g_u(h, r) = M_u^1 h + M_u^2 r + b_u$, it also propagates object entity and relation into another new representation $g_v(h, r) = M_v^1 t + M_v^2 r + b_v$. The final score function is defined as dot product of two new representations $f(h, r, t) = g_u(h, r)^T g_v(h, r)$. NTN model uses entities h, t as input, the entity h, t are combined by a relation-specific tensor M_r as connection weigh between input layer and middle layer, after activated by $\tanh$ function, the result is used into dot product with relation embedding r . The final score function of NTN can be defined as following, $f(h, r, t) = r^T \tanh(h^T M_r t + M_r^1 h + M_r^2 t + b_r)$. MLP model is simple in assumption, it just combines entity h, t and relation r together with connection weigh for each of them M_1, M_2, M_3 , then by a semantic matching score can be computed through tanh activation and connection weigh w before output layer. The final score function is in form of $f(h, r, t) = w^T \tanh(M_1 h + M_2 r + M_3 t)$.

2.3.3 Relation-based Models

Relation-based Knowledge Graph offers a new direction for KG completion task, the Knowledge Graph reasoning of relation-based models relies on relations or relation paths, relation-based models try to leverage information contained in edges and paths of the Knowledge Graph network to simulate the chain of human knowledge reasoning. Yoon et al. [102] added a role-specific mapping matrix for each entity to preserve the logical properties among relations. Lin et al. [43] studied the path between entities, while only line form paths are considered. For these works, the embedding vector for each entity is still required to be learned. Dong et al. [27] intend to research complex link structure of KG and other heterogeneous networks, but the Meta-path used in this work need manual designing for each application scenarios.

2.4 Knowledge Graph based Application Systems

With multiple choices in Knowledge Graph embedding and self completion, application out of Knowledge Graph is a key step to integrate Knowledge Graph into some application systems of the real world as a semantic enhancement.

Because this thesis mainly discusses embedding learning in Knowledge Graph, this part of the literature review focuses on embedding learning related Knowledge Graph applications. In most of Knowledge Graph application based on embedding methods, the entities in application systems can be recognized and linked to entity nodes in the Knowledge Graph network. Because of the structural inhomogeneity between the Knowledge Graph and application system, collaborative methods are designed to jointly learn embedding representation for components of both Knowledge Graph and application system.

A few examples of Knowledge Graph based application systems in several areas are given, including relation extraction system, question answering system, and recommender system.

Relation Extraction System is a task to extraction knowledge relations and facts from textual original data. Specifically, given a paragraph of text, textual mention, the task is to find knowledge relations which are written in complex natural language based on existing structured knowledge facts in current Knowledge Graph. In most of the cases, the task can be proceeded by two steps, 1) recognizing entities from textual data, 2) finding textual format knowledge relation based on recognized entities and structured knowledge facts related to these entities.

One typical work in this area is using collaborative learning for relation extraction and Knowledge Graph [92]. In this work, a collaborative learning framework is proposed to combine the relation extraction process with Knowledge Graph embedding methods, such as TransE. As shown in 2.2, the objective function of this work consists of two parts, $S_{m2r}(m, \hat{r}_{h,t})$ represents the matching score of textual mention m and relation $\hat{r}_{h,t}$, $S_{KG}(h, \hat{r}_{h,t}, t)$ represents the score function value of existing knowledge fact triple $(h, \hat{r}_{h,t}, t)$.

$$\begin{aligned}
 S_{m2r+KG}(h, \hat{r}_{h,t}, t) &= \sum_{m \in \mathcal{M}_{h,t}} S_{m2r}(m, \hat{r}_{h,t}) + S_{KG}(h, \hat{r}_{h,t}, t) \\
 S_{m2r}(m, \hat{r}_{h,t}) &= f(m)^T r \\
 S_{KG}(h, \hat{r}_{h,t}, t) &= - \|h + r - t\|_{1/2}
 \end{aligned}
 \tag{2.2}$$

Another typical Knowledge Graph based relation extraction method is proposed in [70], matrix factorization techniques are used to solve the relation extraction problem. The matrix consists of two dimensions, one dimension represents pairs consisting of subject entity and object entity, the other dimension represents textual mentions and structured relations in Knowledge Graph. With matrix factorization operation, the embeddings for entities, textual mentions, and relations are all learned. And in [19], as an extended work, the tensor factorization techniques are used instead of matrix factorization. A tensor with three dimensions are constructed, textual mentions and relations are still represented by one dimension, the other two dimensions represent subject entities and object entities respectively. The tensor factorization techniques are required for embedding learning.

Question Answering System is also a type of application systems which highly rely on the semantic enhancement from Knowledge Graph. In [9, 14], a collaborative learning model is proposed for joint learning on question answering task and Knowledge Graph. With structured Knowledge Graph, as well as the question and answer text corpus, the key idea is to learn low-dimensional embedding for words in text corpus and Knowledge Graph components jointly. The similarity of question and answer is defined in Eq. 2.3. W is a matrix to represent the embedding of words, entities, and relations, $\phi(q)$ is a sparse vector representing the occurrence of words in the question, and $\psi(a)$ is a sparse vector representing the occurrence of entities and relations in the answer.

$$S(q, a) = (W\phi(q))^T (W\psi(a)) \tag{2.3}$$

Recommender System with Knowledge Graph is a hot topic drawing lots of attention recently. Some previous works, such as [103], attempted manually defined the path between recommender system network and Knowledge Graph network to improve the accuracy of recommendation. A heterogeneous network consists of several predefined types of nodes is constructed for analysis, such as a movie recommender system network can be constructed by users, movies, actors, etc. Then some manually defined paths are used as features for computing similarity between nodes, for example, a meta-path user-movie-actor-movie indicates that a user may have an interest in the movies with mutual actors. Some embedding methods can be applied, such as [27], while the generalized capability is limited because of manually defined paths.

Another recent work [105] proposed a collaborative embedding learning framework for joint embedding learning between Recommender System and Knowledge Graph. In this work, a collaborative embedding framework is proposed, the key idea is to build an end-to-end solution for jointly learning embedding for components of both recommender system and knowledge graph instead of learning embedding of each side separately. The BPRMF[68] model and TransE/TransR [12, 44] model are integrated into the final joint objective function for joint learning.

BIPARTITE GRAPH EMBEDDING BASED KNOWLEDGE GRAPH COMPLETION

3.1 Introduction

Knowledge Graph completion is always an important task in the area of Knowledge Graph Analysis, because of the incompleteness of the currently popular Knowledge Graphs. Although the Knowledge Graphs are incomplete, it still contains a large amount of structured human knowledge facts, which can be used to learn knowledge reasoning patterns. The reasoning patterns are helpful for predicting missing knowledge facts. and finally, the Knowledge Graph can be completed by predicted facts.

Embedding based representation learning is proved to be a useful solution for Knowledge Graph completion task, multiple embedding based methods have been proposed in this area. Using the structured Knowledge Graph fact triples as input to learning proper embedding representation for the Knowledge Graph components including entities and relations, some embedding based models can effectively find missing knowledge facts based on learned embedding representation. Among these embedding based methods, Structured Embedding (SE) [13] and TranE [12] are explainable and effective.

In particular, SE gives every relation two matrix, R_{lhs} and R_{rhs} for transforming left and right entity, e_h and e_t , into a subspace. The entity vectors of e_h and e_t are h and t , then SE assumes that $R_{lhs}h \approx R_{rhs}t$, if triplet (e_h, r, e_t) exists in the knowledge base. Since one relation has two directions of transform, i.e., forward transform and

backward transform, SE method uses two matrices to represent two transforms of one relation, however, ignoring the interaction between the forward transform and backward transform. Differently, TransE uses a vector to represent each entity and relation. TransE assumes that $h + r \approx t$ (h and t denote the vectors representing entity h and t , r representing relation r). TransE uses vector "+" operation to represent relation transforms. The key limitation of TransE is that it can not handle the complex relations in KBs (e.g. N-to-1, 1-to-N, N-to-N relation).

Inspired by some graphical models [31, 37], especially Restricted Boltzmann Machine [30], a bipartite graph embedding based model for Knowledge Graph completion task is proposed, Bipartite Graph Network Structured Embedding (BGNSE), which is simple, universal, and efficient. In particular, BGNSE, uses one complete bipartite graph to represent each triplet in KB. In one triplet, each entity is mapped into one layer of a bipartite graph, and links between layers is to represent relation. The forward and backward relation transforms are performed by forward and backward propagation of bipartite graph network, which share the same bipartite graph transition matrix. The Bipartite Graph Network can use only one matrix (link weight matrix of Bipartite Graph Network) for each relation representation instead of two matrices used by SE. This is because the relation is represented by the link weights between two layers of the bipartite graph, which is shared by both relation transform directions (forward transform and backward transform). On the other hand, TransE model is simpler and uses one vector for each relation transform, but it can not distinguish complex relations (e.g. N-to-1, 1-to-N, N-to-N relation). Because the BGNSE uses network propagation, i.e., dot operation between matrix and vector, as relation transform, the complex relations in KBs can be accordingly handled and derived.

In reasoning process of BGNSE, the reasoning is dual-way, in the forward reasoning, the head entity is projected by relation matrix, and in the backward reasoning, the tail entity is projected by relation matrix. When the same entity composes with different relations, this kind of matrix projection can transfer the original vector representation of the same entity into different target vector representations based on different relations, so it can solve the complex relations problem better than direct translation operation as TransE model do.

3.2 Preliminaries

3.2.1 Structured Embedding (SE) and TransE

As mentioned above, the SE and TransE model are both effective solutions for KB embedding problem. Although there are some issues with these two models, the framework and score function are well designed.

SE [13] is a classic model for Multi-relational data embedding. Given a triplet (h, r, t) of KB, it use a vector to represent each entity, vector $\vec{h}$ for entity h , vector $\vec{t}$ for entity t . SE use two matrices for one relation r , R_{lhs} and R_{rhs} , to project entity h and t into a subspace. The score function used in the model is defined as

$$(3.1) \quad score(h, r, t) = \|R_{lhs}\vec{h} - R_{rhs}\vec{t}\|_1.$$

The model assumes that if the triplet (h, r, t) is true according to KB, the score function value of triplet, $score(h, r, t)$, should be low. The model offers a good framework for solving KBs Structure Embedding problem, while the meaning of projection subspace is unclear.

TransE [12] is another effective embedding model. With the similar framework, the author give a different definition of score function. Using vectors $\vec{h}$ and $\vec{t}$ for entities h and t , using another vector $\vec{r}$ for each relation r . Using the score function

$$(3.2) \quad score(h, r, t) = \|\vec{h} + \vec{r} - \vec{t}\|_2^2.$$

TransE uses simple and understandable relation transform, but has a new problem. TransE can't handle complex KB relation problems, such as multiple relations match or multiple entities match.

3.2.2 Other Models

Unstructured Model (UM) UM [10] is a model which is close to TransE, but with a simpler score function

$$(3.3) \quad score(h, r) = \|\vec{h} - \vec{r}\|_2^2.$$

The UM model can't identify the difference between relations, it can only output the relation of two entities is close or not.

TransH TransH [90] is also a simple and effective model, it addresses some issues of TransE. TransH uses a hyperplane w_r to represent relation r and vector h and vector t

for entities h and t . h and t are projected to hyperplane w_r , $\vec{h}_\perp$ and $\vec{t}_\perp$. Then the score function is defined as

$$(3.4) \quad score(h, r, t) = \|\vec{h}_\perp + \vec{r} - \vec{t}_\perp\|_2^2.$$

The projection operations are defined as $\vec{h}_\perp = \vec{h} - w_r^T \vec{h} w_r$ and $\vec{t}_\perp = \vec{t} - w_r^T \vec{t} w_r$.

Neural Tensor Network (NTN) NTN [77] defines the score function as follows,

$$(3.5) \quad score(h, r, t) = \vec{u}_r^T \tanh(\vec{h}^T R \vec{t} + R_1 \vec{h} + R_2 \vec{t} + \vec{b}_r),$$

where u_r is a relation-specific linear layer, R is a 3-way tensor, R_1 and R_2 are weight matrices for entities h and r . The model is considered too complex for applying in knowledge base efficiently. [44]

TransR and CTransR TransR and CTransR [44] is another approach attempting to solve the issues of above models. It assumes that the entities and relations are not in the same semantic space. TransR uses matrix M_r for projecting entities from entity space to relation space by $\vec{h}_r = \vec{h} M_r$ and $\vec{t}_r = \vec{t} M_r$, where $\vec{h}$ and $\vec{t}$ are entity embeddings. The score function is

$$(3.6) \quad score(h, r, t) = \|\vec{h}_r + \vec{r} - \vec{t}_r\|_2^2.$$

3.3 Methodology

To get a reasonable and computable representation of entities and relations in knowledge bases, we offer a structured embedding framework to define the abstract score function and learning framework for our embedding solution. We also propose a new method, Bipartite Graph Network Structured Embedding (BGNSE). BGNSE method embeds entities and relation of KB triplet into the bipartite graph and uses propagation between bipartite graph network layers to represent the relation transform between entities.

3.3.1 Structured Embedding Framework

Because of the issues of other methods, we intend to build a structured embedding framework which includes the following features.

Directly transform entities to each other. The score function of each triplet is used for determining the triplet is true or not. We use a score function inspired by score functions of previous works, but we do some modification to transform entities to

each other directly based on the relation between them. For each triplet (e_h, r, e_t) in the knowledge base, the score function of a give entity-relation triplet is defined as:

$$(3.7) \quad S(e_h, r, e_t) = \|R(h) - t\|_p + \|R_{rev}(t) - h\|_p,$$

where h and t mean the d dimensional representation vectors of entities e_h and e_t . R is forward relation transform function of relation r , and R_{rev} is backward relation transform function of relation r . R projects left entity vector h to a predict right entity vector $R(h)$ which should be approximate to real right entity vector t , R_{rev} projects right entity vector t to a predict left entity vector $R_{rev}(t)$ which should be approximate to real left entity vector h . We use p-norm to determine the approximation between predict entity vectors and real entity vectors.

Forward and backward relation transforms should be related. The relations of KB are directed. For example, if A "is_part_of" B, B can not have "is_part_of" relation to A. But the two directions of relation transform should be related, the relation from B to A is obviously not "is_part_to", maybe "contain" or "include", but it should be strongly related to "is_part_to" relation. In that case, the R transform function and R_{rev} reverse transform function in equation (7) should be extremely relevant, and R and R_{rev} can be transformed into each other by the specific operation. We introduce our method in section "Bipartite Graph Network Architecture" that using one Bipartite Graph Network to represent both R transform and R_{rev} transform, R transform and R_{rev} transform share the link weight matrix of the same bipartite graph.

Can be used to represent the complex knowledge relations. Knowledge bases contain some complex relations, such as multiple relations match or multiple entities match. For example, there is more than one relationship between two entities, or there is more than one entity has the same relation to the same entity. Some previous methods, such as TransE, have issues with complex knowledge relations. This is because TransE uses simple entity vector transform. Our method can distinguish the difference between two relations between the same entity pair by using Bipartite Graph Network. The detail method introduction is in section "Bipartite Graph Network Architecture".

The results contain latent information of knowledge. Some latent information hidden in the human knowledge base can be used to discover the nature of human knowledge. By structure embedding operations, the latent information can be extracted by deep analyzing the relation structure of big knowledge bases. The results of our method will extract the latent features of the original knowledge base into a compressed low dimensional representation.

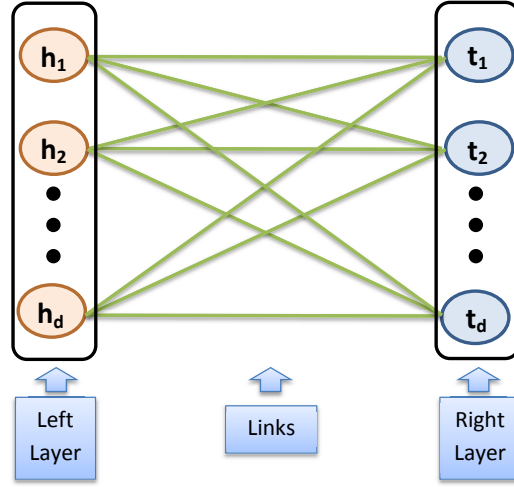


Figure 3.1: Bipartite Graph Network Architecture

3.3.2 Bipartite Graph Network Architecture

Graphical Network has been widely used in many cases including some classic machine learning model. Bayesian Network [63] and Boltzmann Machine [1] are both widely used graphical models. Restricted Boltzmann Machine (RBM) [30] put units in two layers and only remain the links between layers, the units in the same layer are unlinked. This method makes the model effective and can be used in some application [72]. The success of RBM gives us hints to use bipartite graph network in our model.

Our new structured embedding method, Bipartite Graph Network Structured Embedding (BGNSE), uses entity-relation-entity triplet contained in knowledge bases as input, and it outputs the structured embeddings representation for entities and relationships. The output represents a triplet by using one entity vectors for each entity and one matrix for the relation in one triplet. BGNSE uses one complete Bipartite Graph Network to represent the transform operations of one triplet. Each entity vector is put into each layer of a bipartite graph, use cells of vector for units of the layer. The relation matrix is used as the weights of links between two layers of the bipartite graph. Then our structured embedding framework can use bipartite graph network architecture (shown in figure 1) to define the front and reverse relation transform functions used in score function, R and R_{rev} in equation (7).

Given triplet (e_h, r, e_t) , using d -dimensional vector h to represent the entity vector of entity e_h , using d -dimensional vector t to represent the entity vector of entity e_t . The i_{th} cells of vector h is h_i , and the j_{th} cells of vector t is t_j . Put vector h into left layer

of bipartite graph and put vector t into right layer of bipartite graph as figure 1. Using links between two layers to represent the relation between two entities, the link weight between h_i cells and t_j cells is $w_{i,j}$. Equation (8) and (9) can be gotten.

The values of right entity vector t , t_j ($j \in [1, d]$), are approximate to predict right entity vector values computed by weight sum of values in left entity vector h as figure 2, the weights are the bipartite graph link weights between cell t_j and all cells of vector h .

Reversely, the values of left entity vector h , h_i ($i \in [1, d]$), are approximate to predict left entity vector values computed by weight sum of values in right entity vector t as figure 3, the weights are the bipartite graph link weights between cell h_i and all cells of vector t .

$$(3.8) \quad t_j \approx \sum_{i=1}^d (h_i \times w_{i,j}), \quad j \in [1, d].$$

$$(3.9) \quad h_i \approx \sum_{j=1}^d (t_j \times w_{i,j}), \quad i \in [1, d].$$

Obviously, relation r can be represented by link weight matrix of bipartite graph W (the link weight between h_i cell and t_j cell is $w_{i,j}$, the i_{th} row j_{th} column value of matrix W). The front transform function R and reverse transform function R_{rev} of relation r in equation (7) can be represented by $R(h) = hW$ and $R_{rev}(t) = tW^T$. The equation (7) turns

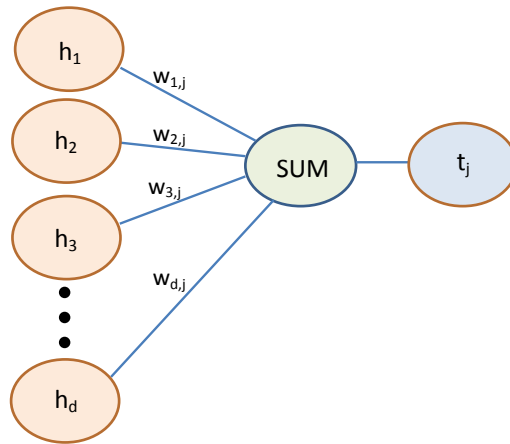


Figure 3.2: The process of getting one unit value of predict right entity vector by using weight sum of all units in left entity vector, all the units in predict right entity vector are computed as the process described in figure.

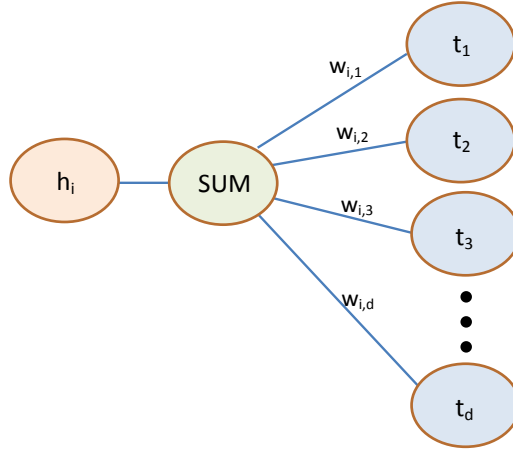


Figure 3.3: The process of getting one unit value of predict left entity vector by using weight sum of all units in right entity vector, all the units in predict left entity vector are computed as the process described in figure.

into following definition.

$$(3.10) \quad S(e_h, r, e_t) = \|hW - t\|_p + \|tW^T - h\|_p,$$

The front transform function R and reverse transform function R_{rev} are strongly related. R uses dot result of left entity vector h and bipartite graph link weight matrix W to get the predict right entity vector. R_{rev} uses dot result of right entity vector t and the transpose of the matrix W to get the predict left entity vector. Our method BGNSE uses only one bipartite graph link weight matrix to represent one relation. Because BGNSE uses dot operation between entity vector and link weight matrix to get the predicted result, it is obvious that BGNSE can solve the complex relation problems in KBs.

3.3.3 Training

In our model, the following parameters should be trained, entity vectors (h and t) and bipartite graph network link weight matrix (W). We choose $p = 1$ for our score function to simplify the training process. The final score function for training is defined as follows.

$$(3.11) \quad S(e_h, r, e_t) = \|hW - t\|_1 + \|tW^T - h\|_1,$$

Given the knowledge base triplet dataset X , true triplet $(e_h, r, e_t) \in X$. We pick up false triplets as negative training sample, $(e_h, r, e) \notin X$ and $(e, r, e_t) \notin X$. The purpose of our training is to get the result that

$$(3.12) \quad S(e_h, r, e_t) < S(e_h, r, e), \forall : (e_h, r, e) \notin X$$

and

$$(3.13) \quad S(e_h, r, e_t) < S(e, r, e_t), \forall (e, r, e_t) \notin X.$$

We want the score function result of the true triplet is smaller than false triplet, as shown in equation (12) and (13).

We use stochastic gradient descent (SGD) for the parameters, the training of our method. We build an objective function

$$(3.14) \quad O = \begin{cases} m + S - S_{neg} + P & \text{if } m + S - S_{neg} \geq 0; \\ 0 & \text{if } m + S - S_{neg} < 0, \end{cases}$$

where S means the score function of triplet in KB (true triplet), S_{neg} means the score function of negative triplet not in KB (false triplet). We use $P = \lambda \|W\|_1$ in objective function as sparse penalty function to keep the relation transform matrix W sparse and avoid the model from overfitting. We use the L1 regularization penalty function here, the same penalty function is also used in Sparse Coding method [38]. The detail of using L1 regularization can be found in [56]. $\max(a, b)$ function gets the max value from a and b . We add a limitation to the objective function that no optimization is applied if $m + S - S_{neg} < 0$. $m + S - S_{neg} < 0$ means that the margin between the score function results of true triplet and false triplet is large than m . If the optimization is still applied when $m + S - S_{neg} < 0$, the objective function value of some pairs of true and false triplet will overfit, but the others won't fit well. In this case, we add limitation for training to minimize the objective function for the most number of the training set. Parameter m is the margin between score function results of the true and false triplet. m can be set depends on the detail need of a result. The larger m is set, the difference between score function results of the true and false triplet is more evident. While the value of m should be tuned carefully, a too-large value for m can lead to a local optimization problem (only results for several specific triplets are correct, others are incorrect), the global effectiveness of training results can be compromised.

By minimizing the objective function, the structured embeddings can be gotten. The detail train algorithm of our model is defined as in algorithm 1.

Algorithm 1 SGD for BGNSE

```

for Fix number of iterations do
    Select a triplet  $(e_h, r, e_t)$  from  $X$  at random.
    Select an entity  $e$  from entities at random.
    if  $(e_h, r, e) \notin \text{KB triplets}$  then
        if  $m + S(e_h, r, e_t) - S(e_h, r, e) \geq 0$  then
             $\text{minimize}\{m + S(e_h, r, e_t) - S(e_h, r, e) + P\}$ .
    if  $(e, r, e_t) \notin \text{KB triplets}$  then
        if  $m + S(e_h, r, e_t) - S(e, r, e_t) \geq 0$  then
             $\text{minimize}\{0, m + S(e_h, r, e_t) - S(e, r, e_t) + P\}$ .

```

We also make a constraint after each iteration, the L2-Regularization of each entity vector should be regularized to 1, to avoid vectors of entities become too close to zero, the zero-close value entity vectors can also lead to a small result value for the objective function.

3.3.4 Latent Information

The meaning of using Bipartite Graph Network for structured embedding is not only getting a representation embedding for each relation. The process of training Bipartite Graph Network also allows our model to learn some abstract latent information from the structure of the whole knowledge base. The bipartite graph network link weight matrix representing relation transform has to be trained for fitting all the triplets containing this relation, abstract features of relation are extracted into relation bipartite graph network link weight matrix during the training process.

If we use simple sparse representation for knowledge entities and relations embeddings, such as one-hot vectors for entities and sparse matrices for relations, the result can represent the relations already contained in KB, but the latent information of knowledge can not be represented. The hidden or missed relation of a knowledge base can not be found by sparse representation, because the pattern of relations is not learned.

The original KB stores the entities and the relations between them with sparse representation, high dimensional sparse vectors and matrices will be used if we represent KB entities and relations directly. The structure embedding methods embedding the KB data into a low dimensional compressed representation containing latent information for knowledge completion and analytics such as missed knowledge relation found and relation similarity prediction. BGNSE takes advantage of Bipartite Graph Network, the result of BGNSE is a highly compressed and abstract representation of knowledge,

Table 3.1: Ranking results on FB200 (146 entities)

	train data (left / right)	test data (left / right)
TransE	16.00 / 24.45	21.07 / 30.80
SE	15.55 / 18.26	21.69 / 23.85
BGNSE	10.60 / 8.18	13.32 / 16.31

Table 3.2: Ranking results on FB500 (485 entities)

	train data (left / right)	test data (left / right)
TransE	86.20 / 104.46	94.23 / 111.13
SE	29.73 / 26.76	35.68 / 34.99
BGNSE	29.61 / 25.07	32.68 / 33.31

the latent information contained by BGNSE result can be used for knowledge nature discovery.

The relation matrices used in BGNSE model can fit the dataset during the training. The relation matrices are low dimensional compressed matrices. The result relation matrices have to fit most of the data in KB, the result relation matrices have to contain the abstract pattern of knowledge relation. The accuracy of knowledge abstract representation will increase by using a larger size of KB datasets.

3.4 Experiment and Analysis

TransE and SE are two classic multi-relational embedding methods. The model of TransE method is understandable, the efficiency of TransE is high enough for most of the embedding problems. SE methods can handle complex relation in relational triplet sets, while TransE can not handle complex relation problem. The datasets we used in our experiments contain a high proportion of complex relations. We use TransE, SE and our method to compare the differences between our method and two types of classic embedding methods.

For these reasons, we conduct experiments using TransE and SE as baselines on two datasets extracted from Freebase dataset. The results of the experiment are tabulated in Table 1 and 2, respectively. Because the complicated nature of relations contained in our datasets, the TransE performs worse due to its intrinsic weakness as described above. SE and BGNSE perform closely, the average rank result of BGNSE is slightly better than SE.

3.5 Summary

We proposed BGNSE, an approach for multi-relational data structured embedding. In our method, due to the advantage of Bipartite Graph Network, our method only needs to learn one matrix to model structured embeddings and handle the complex KB relations without the need of subspace projection. The forward transform and backward transforms of one relation are strongly related. Forward and backward transform matrix can turn into each other by the transpose of itself because the relation transform is represented by a bipartite graph network link weight matrix. The experiments have justified the effectiveness of our approach.

Furthermore, we believe this is a knowledge relation representation method which is more natural and reasonable because the relation transform matrix can learn abstract latent information of relation during the learning steps of the bipartite graph network model.

EMBEDDING BASED CROSS COMPLETION FOR FACTUAL KNOWLEDGE BASE AND CONCEPTUAL TAXONOMY

4.1 Introduction

In this chapter, one work of mine which is prepared to be submitted as a journal paper is introduced.

In the past years, many knowledge base completion methods have been proposed and shown effectiveness on a single knowledge base completion task. Nevertheless, existing methods suffer from the limited information problem, in which the total amount of information in a single knowledge base is always limited. Regardless of the completion method, the amount of information in a single knowledge base creates an upper bound to the complete performance. In other words, if a piece of knowledge is entirely missing in the current knowledge base then there is no way to discover it with existing completion methods.

However, the missing knowledge may still exist in other knowledge bases as knowledge bases are often overlapped and complementary to each other. For example, we can learn the following knowledge about the entity *Facebook* from *DBPedia* ¹ [39]: “*Facebook* is a *public company*” and “*Instagram* and *Whatsapp* are products of *Facebook*”. However, additional knowledge such as “*Facebook* is an *Internet company* and a *technology company*” does not present in *DBPedia* but is available in *Probase* ² [85, 93]. These

¹An online large-scale factual knowledge graph <http://wiki.dbpedia.org>

²An online large-scale conceptual taxonomy <https://concept.research.microsoft.com>

missing pieces of knowledge prevent traditional completion methods from identifying the potential links from *Facebook* to other semantically similar companies such as *Google*, *Apple* and *Microsoft*.

In this example, *DBPedia* is a factual knowledge graph consists of (subject, predictor, object) triples and *Probase* is a conceptual taxonomy consists of (concept, instance) pairs. If we consider the knowledge base as a graph structure, then the above example shows that the entity *Facebook* in a factual knowledge graph could be more accurately positioned by considering relevant knowledge in conceptual taxonomy, and vice versa.

Although integrating multiple knowledge bases is a promising direction towards better completion quality, it remains a challenging task of linking heterogeneous knowledge bases. Specifically, existing completion methods are designed for a single knowledge base completion and will fail when knowledge is stored in different structures across knowledge bases, e.g., triplets and pairs. To overcome this barrier, we propose to jointly complete multiple knowledge bases through a collaborative embedding model which not only connects heterogeneous knowledge bases but also enables the sharing of knowledge even if the knowledge pieces are presented in different structures, e.g., triplets and pairs. To be specific, we target on a joint completion framework where representations of entities are learned by embedding for triplets and matrix factorization for pairs. In addition, depending on the overlapping and complement of two knowledge bases, the number of knowledge pieces to be shared for different entities may vary from small to large numbers. Therefore, an adaptive prediction method is introduced to adjust the completions accordingly.

The main contributions are listed as follows:

- A novel collaborative embedding model is proposed to jointly complete a factual knowledge graph and a conceptual taxonomy. Knowledge sharing has been made possible by bridging the two knowledge bases through entity representations even if the two knowledge bases are heterogeneous. To the best of our knowledge, this is the first attempt at knowledge base completion across heterogeneous knowledge bases through a principled collaborative learning framework.
- An adaptive prediction method is introduced to dynamically adjust the number of completions according to the overlapping and complementary of two knowledge bases. In this way, the completion quality has been improved for both the proposed collaborative embedding model as well as for traditional methods.

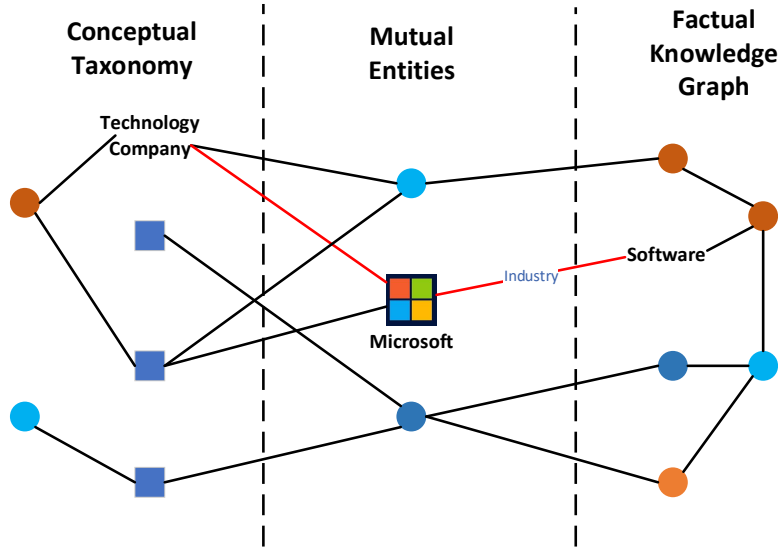


Figure 4.1: Cross Completion System for Factual Knowledge Graph and Conceptual Taxonomy

- Extensive experiments have been conducted on two real-world large-scale knowledge bases to assess the effectiveness of the proposed collaborative embedding model. Special implementation treatment has been made to the optimization process such that parallel computing becomes possible for efficient model training.

4.2 Preliminaries

This section briefly summarizes necessary background of two types of knowledge bases and single knowledge base completion methods that form the basis of this work.

4.2.1 Factual Knowledge Graph and Conceptual Taxonomy

There are 2 types of knowledge bases widely used in multiple knowledge based related systems, factual knowledge graph and conceptual taxonomy. The factual knowledge graph is a semantic web consist of entities and relations, where entities represent anything in the world including people, things, events, etc., and relations connect entities that have interactions with each other. The other type of knowledge base, conceptual taxonomy, are constructed with hypernyms-hyponyms structure. There are two types of

elements in conceptual taxonomy, concept, and instance, “IsA” relation is used as links between concepts and instances.

4.2.2 Structure Embedding based Factual Knowledge Graph Completion

In this word, we mainly use two models, TransE [12] and TransR [44], which are representative of all structure embedding models.

TransE uses d length vectors to represent both entities and relations. In d -dimensional space, every entity is represented by a point, every relation is an arrow. Given a true triple (h, r, t) , h is subject entity, r is predictor relation, t is object entity, translating from h point with r arrow should be close to t point. Following score function can be used to measure the triple is true or false.

$$(4.1) \quad f_r^{\text{TransE}}(h, t) = \|E_h + R_r - E_t\|_2^2.$$

With smaller score function value, the triple is more likely to be true.

TransR inherits the main framework of TransE, while it added one step before space translating. TransR assumes the semantic meaning of entities are different for a different context. To be specific, for two different relations, entities should be mapped into different relation subspaces. Given a true triple (h, r, t) , TransR firstly maps entity h and entity t into subspace of relation r as following equation.

$$(4.2) \quad E_h^r = M_r E_h, E_t^r = M_r E_t.$$

After that the score function of TransR can be computed based on translating in subspace.

$$(4.3) \quad f_r^{\text{TransR}}(h, t) = \|E_h^r + R_r - E_t^r\|_2^2.$$

Similar to TransE score function, with smaller score function value, the triple is more likely to be true.

4.2.3 Collaborative Filtering based Conceptual Taxonomy Completion

Collaborative Filtering (CF) [34, 61] is widely used in recommendation system. CF assumes that two people having similar tastes are likely to have the same preference for a new item. There are several solutions to solve conceptual taxonomy completion

problem, but only CF conceptual taxonomy completion method uses the idea of embedding learning, and it can be integrated with KG embedding learning for collaborative learning. According to one previous work [42], which uses CF as a solution for conceptual taxonomy completion, the structure of conceptual taxonomy is similar to the recommendation system. Considering instance as user and concept as item, the instances with similar semantic meaning are likely to have a similar concept.

For ease of reference, notations used throughout this word are summarized as following. C_i represents vector of user i ; E_i represents vector of entity i ; R_r represents vector of relation r ; M_r represents subspace mapping matrix of relation r ; $\mathcal{X}_{i,j,j'}$ represents instance-concept match term of training function; $\mathcal{Y}_{h,h',r,t}$ represents knowledge graph embedding term of training function; $\mathcal{Z}$ represents regularization term of training function; f_r represents knowledge graph triple score function used in training function; f_r^{TransE} represents TransE score function; f_r^{TransR} represents TransR score function.

4.2.4 Problem Definition

The research problem of this word can be defined as follows: given two heterogeneous large scale high-quality knowledge bases with complementary information, how to leverage information of both two knowledge bases to provide semantic reinforcement for each other, and eventually, properly achieve dual knowledge bases completion.

4.3 Methodology

In this section, we propose the Knowledge Bases Cross Completion system to connect two types of knowledge bases with heterogeneous structure. We leverage complementary information of both knowledge bases to automatically complete the missing knowledge of each other.

4.3.1 Integration of Factual Knowledge Graph and Conceptual Taxonomy

Factual knowledge graph and conceptual taxonomy have different structure. To integrate these two types of knowledge bases, we extract the mutual entities, some entities are used in both knowledge bases. For mutual entity extraction, the representations of the same entity may be different in different KG database, such as the Factual Knowledge Graph DBpedia and Conceptual Taxonomy Probase used in this work. To extract the mutual

entity, first, the entities with exact same representation in two KG database are extract, then, we treat entity textual representation of Probase as text, entity recognition tool DBpedia spotlight [49] can be used to recognize and match DBpedia entity with Probase entity.

As shown in Fig. 4.1, there are three parts in an integrated knowledge base, factual knowledge graph part, conceptual taxonomy part, mutual entity part. Factual knowledge graph part, as shown in the right part of the figure, circles represent entities, the lines between entities represent the relation from one entity to another. Conceptual taxonomy part, as shown in the left part of the figure, squares represent conceptual terms, circles represent instances (we consider instance as an entity) which only appear in conceptual taxonomy, the lines between instance and concept represent a “IsA” relation between them. Mutual entity part, as shown in the middle part of the figure, each element in this part represents an entity appear in both bases. We give an example in Fig. 4.1, entity “Microsoft” is not only an instance of conceptual taxonomy but also a subject entity of factual knowledge graph.

4.3.2 Collaborative Embedding Model

The target of cross completion is to complete knowledge bases with extra information about each other. As shown in Fig. 4.2, we give an example of completing factual knowledge graph with extra information of conceptual taxonomy. In conceptual Taxonomy, “Google” and “Facebook” share three conceptual terms, “Technology Company”, “Internet Company” and “Web Service Provider”. In factual knowledge graph, we have triple (“Google”, “Industry”, “software”), based on extra information of conceptual Taxonomy, we can add a triple (“Facebook”, “Industry”, “software”) to factual knowledge graph.

We show another example in Fig. 4.3, the conceptual taxonomy is completed with extra information of factual knowledge graph. In factual knowledge graph, “iPad” and “iPhone” have same “Operating System”, same “CPU” and same “Manufacturer”. We know “iPad” is an “Apple Device” from conceptual taxonomy, we can predict that “iPhone” is also an “Apple Device”.

To achieve the cross completion target as described in above examples, we propose **Collaborative Embedding Model (CEM)**. CEM jointly learns the embedding of elements, including concepts, instances, entities, and relations. We induce all the elements of factual knowledge graph and conceptual taxonomy into three types, concept, relation and entity (factual knowledge graph entity and conceptual taxonomy instance). Each entity is represented with a vector, each concept is represented with a vector. There are two

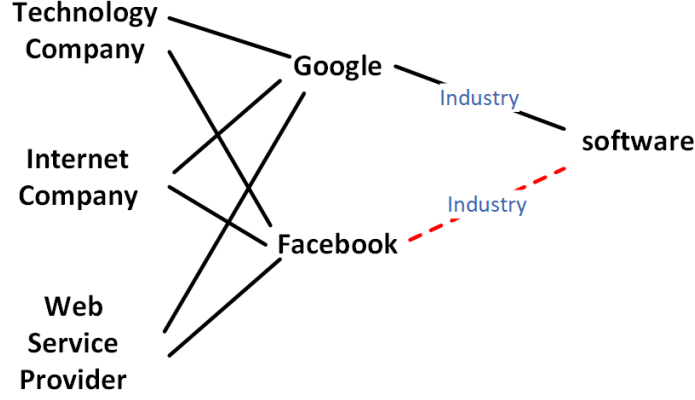


Figure 4.2: Completing Factual Knowledge Graph with information in Conceptual Taxonomy

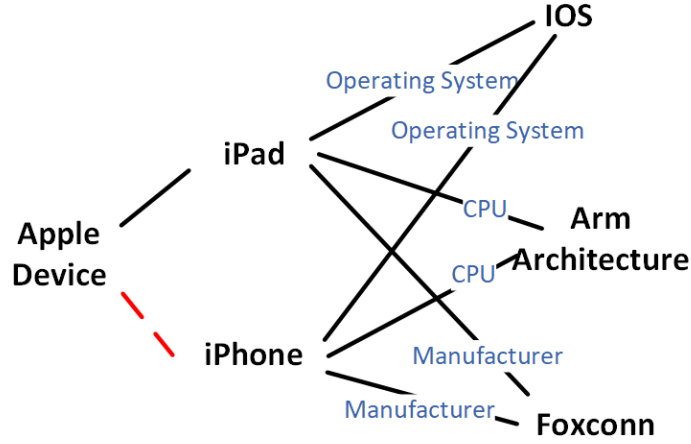


Figure 4.3: Completing Conceptual Taxonomy with information in Factual Knowledge Graph

types of factual knowledge graph score function can be adapted in collaborative learning, score function of TransE and TransR. If we use TransE score function, we represent relation with a vector. If we use TransR score function, we represent relation with a vector and a subspace mapping matrix. As in Eq. 4.4, we put three terms in likelihood equation, $\mathcal{X}_{i,j,j'}$ is conceptual taxonomy term, $\mathcal{Y}_{h,h',r,t}$ for factual knowledge graph term,

Algorithm 2 *CEM* Algorithm

Input: Factual Knowledge Graph, Conceptual Taxonomy.

Training:

Step 1: Draw union set of concepts, subjects $\mathcal{U}$.

Step 2: Repeat

for each $e_i \in \mathcal{U}$ **do**

 Draw pairwise instance-concept triple with e_i as instance into set $\mathcal{D}_i$.

for each $(e_i, c_j, c_{j'}) \in \mathcal{D}_i$ **do**

Compute interaction of user and item:

$$\mathcal{X}_{i,j,j'} = \ln \sigma(E_i^T C_j - e_i^T C_{j'})$$

$$\mathcal{Z}_{i,j,j'} = \|E_i + C_j + C_{j'}\|$$

 Draw pairwise entity-relation quadruple with e_i as true subject into set $\mathcal{S}_i$.

for each $(e_i, h', r, t) \in \mathcal{S}_i$ **do**

Compute score of entity-relation quadruple:

$$\mathcal{Y}_{e_i, h', r, t} = \ln \sigma(f_r(h', t) - f_r(e_i, t))$$

$$\mathcal{Z}_{e_i, h', r, t} = \|E_i + E_{h'} + R_r + E_t\|$$

Compute regularization:

$$\mathcal{Z} = \sum_{i,j,j'} \mathcal{Z}_{i,j,j'} + \sum_{h',r,t} \mathcal{Z}_{e_i, h', r, t}$$

$$\textbf{maximize } \sum_{j,j'} \mathcal{X}_{i,j,j'} + \sum_{h',r,t} \mathcal{Y}_{e_i, h', r, t} + \mathcal{Z}$$

 Stop when the value of likelihood equation $\sum_{j,j'} \mathcal{X}_{i,j,j'} + \sum_{h',r,t} \mathcal{Y}_{e_i, h', r, t} + \mathcal{Z}$ convergence.

$\mathcal{Z}$ for constraint term.

$$\begin{aligned} \mathcal{L} &= \sum_{(i,j,j') \in \mathcal{D}} \mathcal{X}_{i,j,j'} + \sum_{(h,h',r,t) \in \mathcal{S}} \mathcal{Y}_{h,h',r,t} + \mathcal{Z} \\ (4.4) \quad \mathcal{X}_{i,j,j'} &= \ln \sigma(E_i^T C_j - E_i^T C_{j'}) \\ \mathcal{Y}_{h,h',r,t} &= \ln \sigma(f_r(h', t) - f_r(h, t)) \\ \mathcal{Z} &= \frac{\lambda_C}{2} \|C\|_2^2 + \frac{\lambda_E}{2} \|E\|_2^2 + \frac{\lambda_R}{2} \|R\|_2^2 \end{aligned}$$

In each $\mathcal{X}_{i,j,j'}$, we pick up an instance i , a true concept j and a false concept j' , the target is to maximize intersection of instance i and concept j and minimize intersection of instance i and concept j' . In each $\mathcal{Y}_{h,h',r,t}$, we pick up a triple (h, r, t) and a corrupted subject h' , the target is to minimize the score function of (h, r, t) and maximize the score function of (h', r, t) . In some previous factual knowledge graph completion works, both subject and object are replaced by corrupted subject and object, we only replace the subject, by observing dataset, we find the (predictor relation, object entity) pairs existing in current base is a match, if we change object entity, the pair may be meaningless. For example, the pair (“Location”, “China”) means the location of the subject is China, but the pair (“Location”, “banana”) does not mean anything. The proper representation can

be learned by maximizing likelihood equation. If we use TransE score function in our model, we use f_r^{TransE} to replace f_r in equation. If we use TransR score function, we use f_r^{TransR} to replace f_r .

In each step of training, we choose an entity and gather all related factual knowledge graph triples and conceptual taxonomy pairs, then we maximize the likelihood equation. The detail training process is introduced in Alg. 2.

4.3.3 Adaptive True Knowledge Prediction

After learning the proper representation of knowledge base elements, next step is to complete the current bases based on learned representation. For conceptual taxonomy, we pick up one instance, we compute the intersections of this instance and all concepts, then we sort the concepts in descending order. For factual knowledge graph, we pick up one (predictor relation, object entity) pair, we compute the score function values of this pair and all subject entities, then we sort all subject entities in ascending order. The top few concepts and subject entities in the sorted list are likely to be true, the key problem is how many concepts and subject entities should be selected. In some previous works, a parameter K is configured to select top K terms in the list. There is an obvious shortcoming, different instances have a different number of concepts, different (predictor relation, object entity) pairs match different number of subject entities. In some other works, an adaptive K value prediction method is proposed, and the K value is predicted based on the variation tendency of sorted intersection values or sorted score function values, while this method is not effective in all scenarios.

Algorithm 3 Factual Knowledge Graph Adaptive Prediction Algorithm

Input: Current factual knowledge graph, Representation of concept and instance.

Predicting:

Step 1: Draw all concepts $\mathcal{C}$.

Step 2: Draw all instances $\mathcal{I}$.

Step 3: Repeat

for each $e_i \in \mathcal{I}$ **do**

for each $c_j \in \mathcal{C}$ **do**

$$Score_{(e_i, c_j)} = E_i^T C_j$$

compute mean score of known concept-instance pairs, $mean_{score}$

select concepts with scores higher than or equal to $mean_{score}$ as predicted true concepts. $\{c_j \in \mathcal{C} | Score_{(e_i, c_j)} \geq mean_{score}\}$.

Algorithm 4 Conceptual Taxonomy Adaptive Prediction Algorithm

Input: Factual Knowledge Graph, Conceptual Taxonomy.

Predicting:

Step 1: Draw all subjects $\mathcal{S}$.

Step 2: Draw all predictor-object pairs $\mathcal{P}$.

Step 3: Repeat

for each $(r, t) \in \mathcal{P}$ **do**

for each $h \in \mathcal{S}$ **do**

$Score_{(h,r,t)} = f_r(h, t)$

compute mean score of known subject-predictor-object triples, $mean_{score}$

select subjects with scores lower than or equal to $mean_{score}$ as predicted true subjects. $\{h \in \mathcal{S} | Score_{(h,r,t)} \leq mean_{score}\}$.

We apply a new adaptive method in our experiments. For each instance of conceptual taxonomy, we compute the intersections of this instance and all concepts, we select the concepts which are already known as true based on training data, we compute the mean intersection value of known concepts, then we select all concepts with intersection values larger than mean known concept value as true concepts. Similarly, for each (predictor relation, object entity) pair in the factual knowledge graph, we select the subject entities which are already known as true based on training data, we compute the mean score function value of known subject entities, then we select all concepts with score function values larger than mean known subject entity value as true subject entities. The detail is shown in Alg. 3 and Alg. 4.

The adaptive true knowledge prediction method proposed in this work can achieve better knowledge fact prediction accuracy than previous methods, such as top-K model with fixed K parameter or adaptive top-K model with the adaptive K parameter. For each triple missing an entity, the previous model intends to find K most possible entities from candidate entities to fill in the missing part of triple, while it is not effective enough. Given a triple from KG, we can calculate the score function value to indicate the possibility that the triple is true. Generally, the triple with higher score is more likely to be true, but the true triple threshold value for each triple is different, which means for some triples, filling in a candidate entity with relatively low score may be true, and for some other triples, filling in a candidate entity with relatively high score may be false. Through our observation, the key factor here is mean score function value, filling all the candidate entities into the missing part of triple, calculate the mean score function value for all the candidate entities, by using the mean value as threshold to distinguish true triple from false triple is more effective in most cases.

Algorithm	Precision	Probase Recall	F1-Score	Precision	DBpedia Recall	F1-Score
CF	.6822 ± .0079	.3011 ± .0060	.4178 ± .0058	-	-	-
TransE	-	-	-	.9657 ± .0012	.1443 ± .0110	.2509 ± .0165
TransR	-	-	-	.9548 ± .0032	.1880 ± .0153	.3139 ± .0212
CEM(CF+TransE)	.6766 ± .0098	.3166 ± .0063	.4314 ± .0074	.9661 ± .0036	.1397 ± .0099	.2440 ± .0150
CEM(CF+TransR)	.6669 ± .0105	.3409 ± .0090	.4511 ± .0091	.9252 ± .0056	.2447 ± .0100	.3869 ± .0122

Table 4.1: Precision, Recall and F1-score results of Cross Completion on Factual Knowledge Graph and Conceptual Taxonomy

4.4 Experiment and Analysis

In this section, we introduce the detail of our experiment. We firstly introduce two knowledge base dataset used in the experiment. The results of baselines and our proposed method are compared to show the effectiveness of the proposed method.

4.4.1 Dataset

We use *DBpedia* as the factual knowledge graph and *Probase* as the conceptual taxonomy in our experiments. The English version of the *DBpedia* currently describes 6.6M entities consisting of 1.5M persons, 840K places, 496K works, 286K organizations, 306K species, 58K plants and 6K diseases. *Probase* contains 5,376,526 unique concepts, 12,501,527 unique instances, and 85,101,174 “IsA” relations.

With two large-scale knowledge bases containing massive real-world knowledge information, the task of this work is to cross complete two heterogeneous knowledge bases by using complementary knowledge data of both bases, use the bases as the semantic reinforcement of each other to gain better completion result than single knowledge base completion.

4.4.2 Baselines

We use several baselines in our experiments to prove the effectiveness of our proposed model, the baselines include Collaborative Filtering (CF) based Conceptual Taxonomy Completion solution[42], TransE/TransR Structure Embedding based Factual Knowledge Graph Completion solution [12]/[44], CF+TransE based two knowledge bases Cross Completion solution. We choose CF-based Conceptual Taxonomy Completion solution and TransE/TransR Structure Embedding based Factual Knowledge Graph Completion solution to show the difference between cross completion solution and single knowledge

base completion solutions. CF+TransE is a model proposed by us, it is similar to proposed method CF+TransR, but it uses TransE instead of TransR, we compare it with CF+TransR to show the effectiveness of considering relation subspace mapping in the factual knowledge graph.

4.4.3 Comparison

The proposed cross completion solution is CF+TransR method, which integrates CF and TransR model in collaborative learning to jointly learn proper representation for knowledge base elements. We run the baselines and proposed models on same datasets *Probase* and *DBpedia* in the experiments, we run all experiments on two types of servers, servers with 20 core CPU and 128GB memory, servers with NVIDIA Quadro M4000 8GB memory GPU.

In experiment evaluation, we compute precision, recall, and F1-score for predicted new concepts/subject entities. For each instance in *Probase* training data, we predict a list of true conceptual terms for this instance from all conceptual terms. Similarly, for each (predictor relation, object entity) pair in *DBpedia* training data, we predict a list of true subject entities for this pair from all entities. For specific instance or (predictor relation, object entity) pair, the set of conceptual terms or subject entities matching in test data is called real set, the set of conceptual terms or subject entities in the predicted list is called predicted set, the intersection of the real set and predicted set is called the predicted true set. The precision is computed as $|predicted\ true\ set|/|predicted\ set|$, the recall is computed as $|predicted\ true\ set|/|real\ set|$. We compute mean precision and mean recall values of all instance as final precision and recall value of *Probase* completion, and we compute mean precision and mean recall values of all (predictor relation, object entity) pair as final precision and recall value of *DBpedia* completion. F1-score is computed with mean precisions and mean recalls as $(precision * recall)/(precision + recall)$. As shown in Tab. 4.1, we get the result of baselines and proposed model. Firstly, single knowledge base completion methods have an obvious disadvantage, each of them can only complete one knowledge base, if both bases need to be completed, two separate algorithms need to run, and joint learning is not supported. The recall value of the proposed method is significantly improved, which means the proposed method finds more real knowledge information both in *Probase* and *DBpedia*. The precision of the proposed method is minorly compromised, which is acceptable, more real knowledge information is found, more noise data also sneak into prediction result. Actually, most knowledge base completion process need an expert or some rule-based system to double-check the

new knowledge gained by automatic completion algorithm. In this case, if precision is not largely declined, recall value improvement is more important. In F1-scores, we can see that the proposed method CF+TransR has an outstanding result synthetically consider both recall and precision. Comparing the result of CF+TransE and CF+TransR, the result of CF+TransR is much better, which means considering subspace mapping of the relation is essential.

4.5 Summary

In this work, we proposed a Multiple Knowledge Base Cross Completion framework, which integrates multiple knowledge bases. The complementary information of multiple knowledge bases is leveraged to improve the results of knowledge bases completion. Collaborative Embedding Model is proposed for joint learning of heterogeneous structured knowledge bases. Adaptive True Knowledge Prediction is proposed for adaptively select true knowledge from the candidate list. Experiment was conducted on real-world knowledge bases showing that semantic information from both knowledge bases has been properly captured, resulting in improved completion performance. To the best of our knowledge, this is the first attempt of using collaborative embedding model to perform cross completion of multiple knowledge bases, use their unique knowledge information as semantic reinforcement of each other.

SUB-STRUCTURE BASED KNOWLEDGE GRAPH TRANSITIVE RELATION EMBEDDING

5.1 Introduction

The last few years have seen a growing trend in constructing knowledge bases (KB) such as Freebase [8], WordNet [28], DBpedia [4] and Google Knowledge Graph [76]. KB typically stores knowledge in form of the entity-relation-entity triplet, e.g., “*Sydney*”-“*is in*”-“*Australia*”. Collectively, a large number of triplets connect entities into a massive graph structure, which has a wide range of applications, such as recommender system [86, 89, 101, 105], information retrieval [22, 94, 98] and question answering system [32, 46, 100], to name a few.

Despite its important role played in real-world applications, KB is often incomplete by its nature [54] which is one of the main barriers to broader adoption. To address this issue, a considerable amount of literature on *knowledge base completion* has been published with an emphasis on embedding approach [12, 36, 44, 45, 59, 90, 91, 95]. The main idea of this approach is to learn low-dimensional representations of KB entities and relations which can then be used to infer missing triplets. The embeddings can then be used for measuring the plausibility of candidate knowledge pieces for the purpose of completion. Nevertheless, the embedding approach looks at the global structures from the entire KB, thus its effectiveness depends on the sparsity of KB and falls for infrequent entities, i.e., reliable embedding representations can’t be learned for entities

that only appeared a few times.

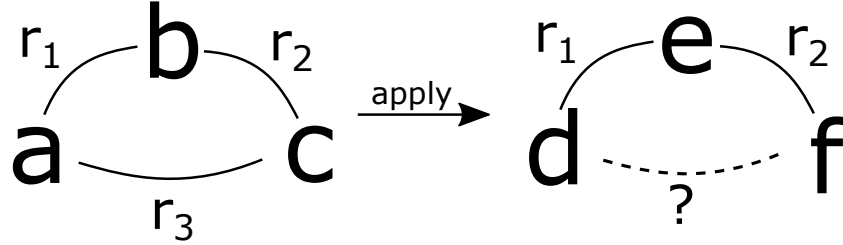


Figure 5.1: Entities a , b , c are connected through relations r_1 , r_2 , r_3 . If these three relations, regardless of which entities are connected, appeared together frequently, then we may believe there is a pattern. The pattern is then applied to an incomplete triangle to predict the missing relation between entities d and f .

In the first work introduced in this chapter, Transitive Relation Embedding, we argue that the local structure can be used to alleviate the sparsity problem by improving the completion of infrequent entities through frequent relation patterns. A typical local structure is transitivity among relations as illustrated in Figure 5.1. The basic idea is that the missing relation between two entities could be inferred from a path connecting them. Although the idea is straightforward, it has several nice properties. Firstly, the relation patterns are independent of entities, thus makes it possible to predict missing relations for infrequent entities, which was a difficult task for the embedding approach. Secondly, identifying relation patterns is less computational expensive [81] comparing to the embedding approach because it does not require the learning of embedding representations for individual entities. Last but not least, relation patterns have great interpretability.

Nevertheless, the plain idea illustrated in Figure 5.1 has its flaws. Firstly, it favors frequent relation patterns thus unable to predict true relations that have never or infrequently appeared in relation patterns. Secondly, it learns strictly triangle relation pattern and does not generalize. To address these issues, we propose a new model called Transitive Relation Embedding (TRE). The idea behind TRE is to learn embedding representations for each relation from transitive relation patterns, which can be then used to predict missing relations. The main difference between TRE and traditional embedding models is that it does not require the learning of entity representations whilst be able to predict missing triplets involving infrequent entities.

In the first work of this chapter, Transitive Relation Embedding model achieves quite an improvement for Knowledge Graph Completion based on Reasoning. However in Transitive Relation Embedding model, only a single triangle pattern is considered in

relation inference, it is not effective enough for relations which need to be inferred by multiple triangle patterns jointly, thus may fail when the inference requires reasoning on multiple patterns. We illustrate this idea in Fig. 5.2. In this example, the two entities Isaac Newton and English Physicist are connected in two path patterns, a.k.a. triangle pattern. However, neither of these two patterns can infer the missing triplet Isaac Newton–IsA–English Physicist, instead, the inference requires reasoning from both triangle patterns. Despite the simplicity shown in this toy example, the actual reasoning process in large KG may need considering multiple triangle patterns simultaneously.

The second work of this chapter is prepared to be submitted as a journal paper. In the second work, Meta-structure Transitive Relation Embedding, we propose a new method to learn from multiple triangle patterns. To be specific, instead of looking at short paths or individual triangle patterns, we define a new pattern called meta-structure that assembles multiple triangle patterns. The meta-structure is illustrated in Fig. 5.3. With assistance from meta-structure, learning from multiple triangle patterns becomes possible. Nevertheless, applying meta-structure faces several challenges. Firstly, not all entities are covered by meta-structures, thus the method cannot predict for entities what are not connected. Secondly, meta-structure learning does not generalize well as the embedding-based approach does.

To implement the idea of meta-structure and overcome the aforementioned issues, we fit meta-structure into embedding learning, namely *Meta-structure Transitive Relation Embedding* (MSTRE). Unlike traditional embedding-based methods, MSTRE only focuses on relation embeddings learned from meta-structures without overheads on entity embeddings.

We summarize the main contributions of these two works as follows:

- For the first time, the data sparsity problem in knowledge base completion is tackled by learning relation embeddings from transitive relation patterns.
- The Transitive Relation Embedding (TRE) model significantly improves completion performance on sparse knowledge bases comparing to state-of-the-art embedding models.
- A new method is proposed to encode complex relation patterns via meta-structure.
- We proposed the Meta-structure Transitive Relation Embedding (MSTRE) model which enables learning relation embeddings from meta-structures.

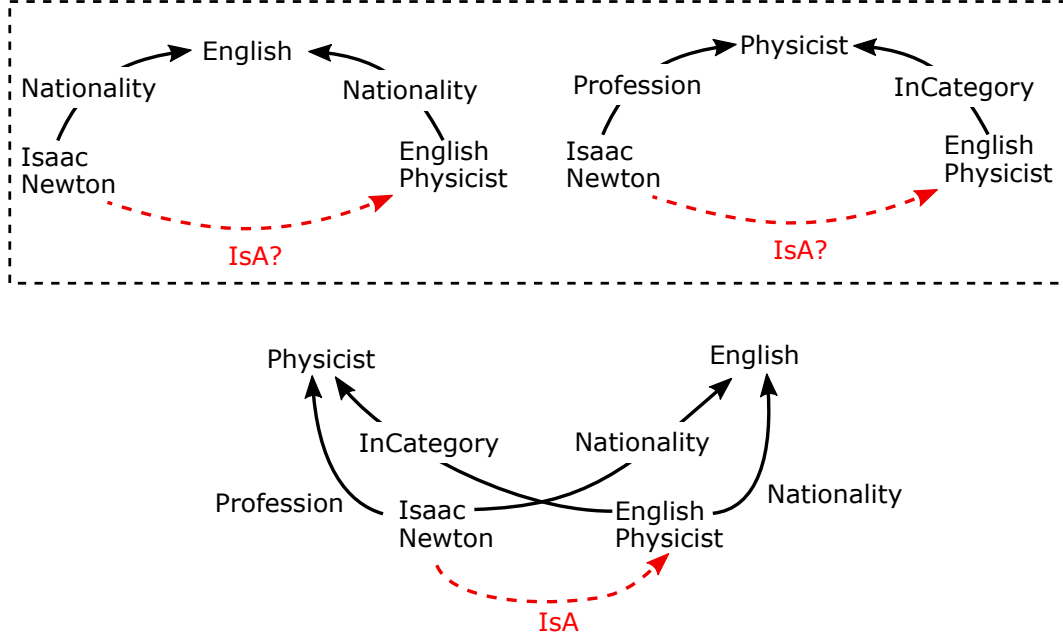


Figure 5.2: This example shows two potential triangle patterns and the task is to infer the missing triplet *Isaac Newton-IsA-English Physicist*. However, neither of the two triangle patterns can infer this missing relation independently. Instead, they must be combined for proper reasoning.

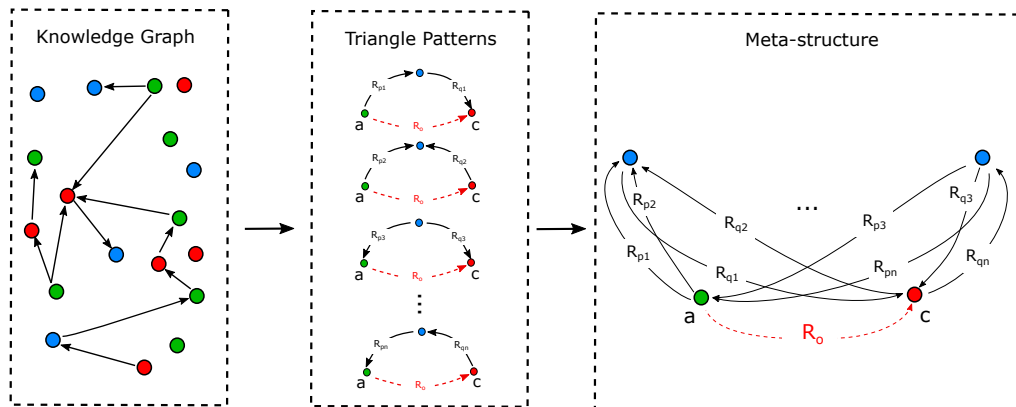


Figure 5.3: We extract triangle patterns from KG, and then merge patterns of the same entity pair into a meta-structure for joint relation embedding learning.

- We conducted extensive experiments on 3 datasets and 7 embedding models to evaluate the proposed TRE and MSTRE models in terms of accuracy and computational complexity.

5.2 Preliminaries

We briefly summarise the background of KG completion, traditional embedding-based KG completion methods and previous works in the field of path-based KG completion.

5.2.1 Knowledge Graph Completion

The Knowledge Graphs represent each human knowledge in form of triplet, (h, r, t) . h is the subject entity, t is the object entity, $h, t \in \mathcal{E}$, $\mathcal{E}$ is the set of all entities. r is the predicate relation from entity h to entity t , $r \in \mathcal{R}$, $\mathcal{R}$ is the set of all relations. The triplet set $\mathcal{D}$ contains all triplets currently existing in KG, while the $\mathcal{D}$ is incomplete, some triplets are missed, the KG completion task is to infer the missing triplets based on given triplet set $\mathcal{D}$.

5.2.2 Embedding-based Knowledge Graph Completion

Embedding-based model is a traditional type of KG completion model, entities and relations of KG are embedded into the same dimensional vector space. Each entity e in entity set $\mathcal{E}$ is represented as a vector $\vec{e}$ and each relation r in relation set $\mathcal{R}$ is represented as a vector $\vec{r}$. The basic idea of the embedding-based model is to learn proper embeddings for entities and relations in KG. One scoring function is defined for each model, for example for TransE model [12], the assumption is that $\mathbf{h} + \mathbf{r} \approx \mathbf{t}$, h is subject entity, t is object entity and r is predicate relation. The scoring function can be defined as:

$$(5.1) \quad f_r(h, t) = -\|\mathbf{h} + \mathbf{r} - \mathbf{t}\|_{1/2}$$

Different assumptions and scoring functions are defined for other embedding models [36, 44, 90, 91, 95], nevertheless, embedding vector of entities and relations are required to be learnt in all embedding based models.

5.2.3 Relation-based Knowledge Graph Completion

Relation-based Knowledge Graph offers a new direction for KG completion task. Yoon et al. [102] added a role-specific mapping matrix for each entity to preserve the logical properties among relations. Lin et al. [43] studied the path between entities, while only line form paths are considered. For these works, the embedding vector for each entity is still required to be learned. Dong et al. [27] intend to research complex link structure of KG and other heterogeneous networks, but the Metapath used in this work need manual designing for each application scenarios. Other works [29, 69, 80] focus on analyzing the embedding learning for large-scale graph, but targeting the homogeneous network, can't fit the multi-relational network directly.

5.2.4 Limitations of Current Models

For traditional embedding based KG completion methods, entity embeddings are required to be learned, and for infrequent entities only appear a few times in KG, proper embedding is hard to be learned. On the other hand, if a huge number of entities contained in KG, learning the embedding for each entity is a big challenge for efficient computing. Path-based and relation-based methods only consider the line segments in paths, and metapath-based method needs manual path design. General homogeneous network embedding methods don't consider multi-relation, obviously unfit for KG completion problem.

5.3 On Completing Sparse Knowledge Base with Transitive Relation Embedding

To solve the sparse KB completion inaccuracy problem, we proposed an embedding based relation inference model. The proposed model focuses on the following issues.

- Instead of training individual knowledge fact, the proposed model extracts knowledge information by using co-occurrence statistics of relations. We use these statistics as the input of embedding model to learn inference rules. We highly improve the accuracy in prediction, especially on sparse KG.
- Our proposed model focuses on explicit relation inference. The proper vector representations are learned for relations, the transitivity of Knowledge Base relations are contained in vector representations (from the relations A-B and B-C we can

infer the relation between A and C). And the vector representations can be used to calculate the possibility of given triangle pattern, the highly possible inference rules can be extracted explicitly, this makes the result of our model highly interpretable.

- In the proposed model, we only learn the embedding for relations, which makes the number of the parameters extremely small. This makes the training process efficient.

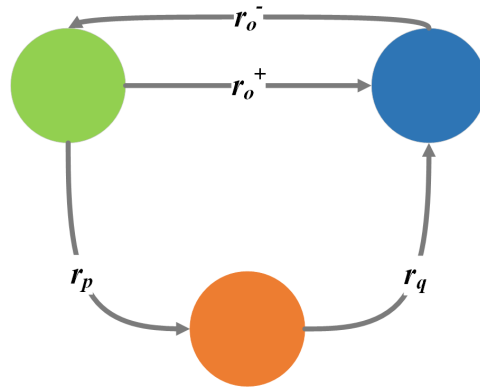


Figure 5.4: Triangle Pattern with Formulated Relation Definition

5.3.1 Triangle Pattern

Inspired by previous work on triangle pattern [81], we observed the triangle structure in our dataset and found that triangle pattern also exists in Knowledge Base. We extract 115,939 triangle patterns from FB15K, 1,068 triangle patterns from WN18 and 46,327 triangle patterns from DBP.

The triangle patterns in social network user relation graph and web page reference link graph can be used for social community detection and web page semantic structure discovery. This makes us assume that the Knowledge Base triangle pattern can be helpful for Knowledge Base hidden relation discovery, and go a step further, Knowledge Base automatic completion. However, Knowledge Base is a multi-relational graph, different than social network user relation graph and web page reference link graph. With entities as nodes, the edges in Knowledge Base represent multiple types of relations between entities. Thus, we focus on relation inference based on the triangle pattern of Knowledge Base.

To formulate the triangle patterns, as shown in Fig. 5.4, we define a restricted triangle structure with three nodes $a(\text{green}), b(\text{orange}), c(\text{blue})$. We represent the relation from a to b as r_p , the relation from b to c as r_q . If the relation between a and c is from a to c , we represent it as r_o^+ , otherwise, we use r_o^- for the relation from c to a . In each restricted triangle structure, either r_o^+ or r_o^- occurs between a and c . In each triangle structure, if one relation is missed, we can use the other two relations to predicted the missed one.

$$\begin{aligned}
 \text{Confidence}(r_o^+|r_p, r_q) &= \frac{\text{Frequency}(r_p, r_q, r_o^+)}{\text{Frequency}(r_p, r_q)} \\
 \text{Confidence}(r_p|r_o^+, r_q) &= \frac{\text{Frequency}(r_p, r_q, r_o^+)}{\text{Frequency}(r_o^+, r_q)} \\
 \text{Confidence}(r_q|r_o^+, r_p) &= \frac{\text{Frequency}(r_p, r_q, r_o^+)}{\text{Frequency}(r_o^+, r_p)} \\
 \text{Confidence}(r_o^-|r_p, r_q) &= \frac{\text{Frequency}(r_p, r_q, r_o^-)}{\text{Frequency}(r_p, r_q)} \\
 \text{Confidence}(r_p|r_o^-, r_q) &= \frac{\text{Frequency}(r_p, r_q, r_o^-)}{\text{Frequency}(r_o^-, r_p)} \\
 \text{Confidence}(r_q|r_o^-, r_p) &= \frac{\text{Frequency}(r_p, r_q, r_o^-)}{\text{Frequency}(r_o^-, r_q)}
 \end{aligned}
 \tag{5.2}$$

If we found a complete triangle structure, “a-r1-b-r2-c, a-r3-c”, we call it triangle pattern. Besides, if we found a triangle consisting of three entities with only two edges, such as “a-r1-b-r2-c”, we call it potential triangle pattern. We use existed triangle pattern in KB for model training, and we predict a new triangle pattern based on existed potential triangle pattern in KG.

Collecting all the triangle patterns in Knowledge Base, we count the co-occurrence frequency of potential triangle patterns as $\text{Frequency}(r_1, r_2)$, r_1, r_2 can be replaced by picking two from $r_p, r_q, r_o^+/r_o^-$. We also count the co-occurrence frequency of r_1, r_2 and r_3 in triangle patterns as $\text{Frequency}(r_1, r_2, r_3)$, r_1, r_2, r_3 can be replaced by $r_p, r_q, r_o^+/r_o^-$, the order can be changed. We can conclude a candidate inference rule, “ $n_1 - r_1 - n_2 - r_2 - n_3 \Rightarrow n_1 - r_3 - n_3$ ”. In textual expression, “If r_1 and r_2 occur on a specific role in the triangle pattern consists of three nodes n_1, n_2, n_3 , we can measure how likely r_3 occurs by a confidence”. As shown in Eq. 5.2, based on occurrence dependency between relations, the confidence can be computed as $\text{Confidence}(r_3|r_1, r_2) = \text{Frequency}(r_1, r_2, r_3)/\text{Frequency}(r_1, r_2)$. The confidence of each candidate inference rule is used in the embedding learning process to indicate how likely the rule is tenable.

5.3.2 Transitive Relation Embedding

To measure the probability of candidate inference rules based on the relation transitive inference, we propose the Transitive Relation Embedding model.

If we observe r_p and r_q in a triangle pattern, then we can predict the occurrence probability of r_o and represent it as $P(r_o|r_p, r_q)$. We have the following probabilities needed to be predicted in our model, $P(r_o^+|r_p, r_q)$, $P(r_o^-|r_p, r_q)$, $P(r_p|r_o^+, r_q)$, $P(r_p|r_o^-, r_q)$, $P(r_q|r_o^+, r_p)$ and $P(r_q|r_o^-, r_p)$.

$$\begin{aligned}
 \overrightarrow{V_{r_p, r_q}} &= M_1 \overrightarrow{r_p} + M_2 \overrightarrow{r_q}, \\
 \overrightarrow{U_{r_o}^+} &= M_3^+ \overrightarrow{r_o}, \\
 \overrightarrow{U_{r_o}^-} &= M_3^- \overrightarrow{r_o}
 \end{aligned}
 \tag{5.3}$$

$$\begin{aligned}
 P(r_o^+|r_p, r_q) &= \frac{\exp(\overrightarrow{U_{r_o}^+}^T \overrightarrow{V_{r_p, r_q}})}{\sum_{r_k}^R [\exp(\overrightarrow{U_{r_k}^+}^T \overrightarrow{V_{r_p, r_q}}) + \exp(\overrightarrow{U_{r_k}^-}^T \overrightarrow{V_{r_p, r_q}})]} \\
 P(r_o^-|r_p, r_q) &= \frac{\exp(\overrightarrow{U_{r_o}^-}^T \overrightarrow{V_{r_p, r_q}})}{\sum_{r_k}^R [\exp(\overrightarrow{U_{r_k}^+}^T \overrightarrow{V_{r_p, r_q}}) + \exp(\overrightarrow{U_{r_k}^-}^T \overrightarrow{V_{r_p, r_q}})]} \\
 P(r_p|r_o^+, r_q) &= \frac{\exp(\overrightarrow{U_{r_o}^+}^T \overrightarrow{V_{r_p, r_q}})}{\sum_{r_k}^R \exp(\overrightarrow{U_{r_o}^+}^T \overrightarrow{V_{r_k, r_q}})} \\
 P(r_p|r_o^-, r_q) &= \frac{\exp(\overrightarrow{U_{r_o}^-}^T \overrightarrow{V_{r_p, r_q}})}{\sum_{r_k}^R \exp(\overrightarrow{U_{r_o}^-}^T \overrightarrow{V_{r_k, r_q}})} \\
 P(r_q|r_o^+, r_p) &= \frac{\exp(\overrightarrow{U_{r_o}^+}^T \overrightarrow{V_{r_p, r_q}})}{\sum_{r_k}^R \exp(\overrightarrow{U_{r_o}^+}^T \overrightarrow{V_{r_p, r_k}})} \\
 P(r_q|r_o^-, r_p) &= \frac{\exp(\overrightarrow{U_{r_o}^-}^T \overrightarrow{V_{r_p, r_q}})}{\sum_{r_k}^R \exp(\overrightarrow{U_{r_o}^-}^T \overrightarrow{V_{r_p, r_k}})}
 \end{aligned}
 \tag{5.4}$$

We learn a k -dimensional vector embedding for each relation in Knowledge Base, each relation is represented as a point in a k -dimensional space, all relations share the same space. But in each triangle pattern, each relation has different role, r_p , r_q , r_o^+ or r_o^- . As shown in Eq. 5.3, for two relations occurs on positions of r_p and r_q , we map their embeddings into one joint role-specific space point as $\overrightarrow{V_{r_p, r_q}}$ by using role-specific matrices M_1 and M_2 . We also map the embedding of the relation in position r_o^+ / r_o^- into role-specific space as $\overrightarrow{U_{r_o}^+} / \overrightarrow{U_{r_o}^-}$. We use M_3^+ as role-specific matrix for r_o^+ , and use M_3^- for r_o^- .

As shown in Eq. 5.4, we can compute the probabilities of relation occurrence based on mapped vector space points $\overrightarrow{V_{r_p, r_q}}$ and $\overrightarrow{U_{r_o}^+} / \overrightarrow{U_{r_o}^-}$. We give the assumption that if the relation inference rule is likely to be true, the probability we predict in our model should be high, which means interaction of $\overrightarrow{V_{r_p, r_q}}$ and $\overrightarrow{U_{r_o}^+} / \overrightarrow{U_{r_o}^-}$ should be high. Our probability equation meets the restrictions in Eq. 5.5.

$$\begin{aligned}
 (5.5) \quad & \sum_{r_k}^R [P(r_k^+ | r_p, r_q) + P(r_k^- | r_p, r_q)] = 1, \\
 & \sum_{r_k}^R P(r_k | r_o^+, r_q) = 1, \sum_{r_k}^R P(r_k | r_o^-, r_q) = 1, \\
 & \sum_{r_k}^R P(r_k | r_o^+, r_p) = 1, \sum_{r_k}^R P(r_k | r_o^-, r_p) = 1.
 \end{aligned}$$

An important reason to use embedding model is that the result of embedding model is one vector for each relation, it can not only compute the probabilities for relation triples occurred in training data, it can also generalize to the relation triples never occurred in training data but need to be predicted in test data. For example, (r_1, r_3, r_5^+) , (r_1, r_4, r_5^+) and (r_2, r_3, r_6^+) occurred in training data, r_2 , r_4 and r_6 never occurred in the same triangle pattern, we can still compute probabilities for inference rule consists of r_2 , r_4 and r_6 , because we learn the transitive inference information of r_2 , r_4 and r_6 and represent it by embedding vector.

5.3.3 Training

With the above definitions, we can get a training likelihood equation for our detail embedding model. We define two triple sets, $\mathcal{S}^+$ and $\mathcal{S}^-$, $\mathcal{S}^+$ consists of relation triples with forward direction third relation r_o^+ (from a to c). Conversely, $\mathcal{S}^-$ consists of relation triples with backward direction third relation r_o^- (from c to a). As shown in Eq. 5.6, in the likelihood equation, we use KL-divergence for distributions of confidence and predicted probability. By maximizing the likelihood equation, the relation inference rule with higher confidence have more chance to result in high probability.

$$\begin{aligned}
 \mathcal{L} = & \sum_{r_p, r_q, r_o^+}^{\mathcal{S}^+} \{ \text{Confidence}(r_o^+ | r_p, r_q) \log[P(r_o^+ | r_p, r_q)] \\
 & + \text{Confidence}(r_p | r_o^+, r_q) \log[P(r_p | r_o^+, r_q)] \\
 & + \text{Confidence}(r_q | r_o^+, r_p) \log[P(r_q | r_o^+, r_p)] \} \\
 (5.6) \quad & + \sum_{r_p, r_q, r_o^-}^{\mathcal{S}^-} \{ \text{Confidence}(r_o^- | r_p, r_q) \log[P(r_o^- | r_p, r_q)] \\
 & + \text{Confidence}(r_p | r_o^-, r_q) \log[P(r_p | r_o^-, r_q)] \\
 & + \text{Confidence}(r_q | r_o^-, r_p) \log[P(r_q | r_o^-, r_p)] \}
 \end{aligned}$$

Table 5.1: Relation Inference Example

Potential triangle pattern	Inferred relation	Probability
r_p : film_release_region, r_q : languages_spoken	r_o^+ : language	0.9916
r_p : actor, r_q : languages	r_o^+ : language	0.9578
r_p : film_release_region, r_q : currency	r_o^+ : currency	0.9433
r_p : spouse, r_q : place_lived_location	r_o^+ : location_of_ceremony	0.9969
r_p : sibling, r_o^+ : ethnicity	r_q : ethnicity	0.9664
r_p : computer_videogame/sequel, r_q : computer_videogame/developer	r_o^- : games_developed	0.9987

5.3.4 Joint Prediction Strategy

We also observed a limitation of the proposed model. It can't predict links between a pair of entities if there is no existing potential triangle pattern in training data between them. We use a strategy to combine the prediction results of the proposed model and baseline models including TransE, TransH, TransR, RESCAL, TransD, DistMult, ComplEx. The final result of the combined model is improved.

In entity link prediction and relation link prediction tasks, we target to predict how likely the given entity-relation-entity triple is true or false. For each triple (a, r_o, c) , a and c are entities, r is one relation from a to c , we detect all the potential triangle pattern between entity pair a and c . Specifically, we find all the combination of r_p, b, r_q which can link a and c , $a - r_p - b - r_q - c$, r_p and r_q are relations, b is an entity. If there is any potential triangle pattern between entity pair a and c , we can compute the probability based on the triangle pattern inference embedding vectors of r_p , r_q and r_o , we predict the triple authenticity with computed probability, we can achieve more accurate prediction than baselines because of the advantages of triangle pattern inference embedding. Otherwise, if there is no potential triangle pattern between a and

c , we can't compute the probability. In this condition, we predict a new triple based on the score function value of baselines. In brief, our proposed method uses the same prediction result for entity pairs with no potential triangle pattern, it uses embedding based triangle pattern inference to achieve better prediction result for entity pairs with the potential triangle pattern.

Table 5.2: Space and Time Complexity of TRE and baselines

	Space complexity	Time complexity	Approx. Exec. Time (s)
TransE	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$	400
TransH	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$	400
TransR	$\mathcal{O}(nd + mdk)$	$\mathcal{O}(dk)$	9,000
RESCAL	$\mathcal{O}(nd + md^2)$	$\mathcal{O}(d^2)$	6,200
TransD	$\mathcal{O}(nd + mk)$	$\mathcal{O}(\max(d, k))$	400
DistMult	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$	1,400
ComplEx	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$	1,800
TRE	$\mathcal{O}(mk)$	$\mathcal{O}(k^2)$	360

5.3.5 Advantages of Proposed Model

Outperform on sparse KG. In sparse KG, some entities occur infrequently, traditional embedding models can't do an accurate prediction for these entities, while our proposed method focuses on relation inference, as long as there is a valid triangle pattern between two entities, we can give an accurate prediction, the entity occurrence frequency is irrelevant.

High interpretability. Our proposed model can achieve high interpretability because the probability of each candidate relation inference rule can be explicitly computed. Given two entities a, c , and a potential triangle pattern $a - r_1 - b - r_2 - c$ between them, we can explicitly compute the probability of relation r_3 between a and c , $P(r_3|r_1, r_2)$. We can conclude an interpretable rule, "If r_1 between a and b , r_2 between b and c , then the probability of there is r_3 between a, c is $P(r_3|r_1, r_2)$ ". In Tab. 5.1, we list some interpretable relation inference rule example computed by our proposed model on dataset FB15K.

Efficient parameter learning. Our proposed model only needs to learn to embed for relations in order to simplify the parameters. With less number of parameters, we can achieve the training efficiency with the proposed model. We list the space complexity, time complexity and running time for convergence of different methods in Tab. 5.2. n is entity number, m is relation number, d is the entity embedding dimension, k is

relation embedding dimension. We can see that TRE method has an advantage in space complexity. And on time complexity, TRE can converge in fewer iterations than other methods, we find that the execution time of TRE is shorter than others.

Why we focus on triangle pattern rule inference? By observing KB structure, we find that transitivity existing in most of existing KG, and it is a key factor for KB completion because it can reliably infer new relations between a pair of entities. To model the KB transitivity, we need a simple but reasonable representation, triangle pattern. We use the triangle pattern to represent the transitive relation inference, and conclude interpretable relation inference rule based on triangle pattern.

Why we use relation transitivity statistics? Compare with baselines, especially TransX KB completion methods, we use a totally different training framework to solve KB sparsity problem. We focus on KB relations only and abandon entity embedding learning, we also use relation transitivity statistics, The occurrence of the triangle pattern, as the input of our learning process. As long as the relation triangle pattern related to an entity is proved by a large number of samples in the entire KG, the infrequent occurrence of this entity doesn't influence the KB completion accuracy.

Why we use embedding model? We use embedding model in this paper for learning result generalization. If a triangle pattern in the testing set has never occurred in training data, we can't determine whether it is true with traditional methods, such as rule-based model, however, the embedding model generalizes the learning result. By embedding model, we can learn embeddings of the relations consisting of the triangle pattern, which enables us to determine the probability of that the triangle pattern is tenable. Through the generalization of embedding model, as long as the relations occurred in training data, we can do a prediction for triangle patterns which has ever occurred in training data.

5.4 Meta-structure based Transitive Relation Embedding for Knowledge Graph Completion

TRE model achieves relatively accurate KG completion for infrequent entities, while we proposed Meta-structure Transitive Relation Embedding (MSTRE) for complex relation transitive inference based on Meta-structure. We focus on addressing the following issues.

- We extract local transitive relation inference structure through Meta-structure,

the relation embedding learned based on Meta-structure can improve the accuracy of KG completion, especially for infrequent entities.

- Our MSTRE model infer new relation r_o between entities based on Meta-structure, which consists of multiple triangle patterns $(a - r_p - b, b - r_q - c)$. The relation prediction result is highly interpretable, we can explicitly analyze the contribution of each triangle pattern and each Meta-structure in the prediction process.
- In MSTRE, we only learn relation embedding without learning the embeddings for entities, which makes the size of parameters small and training efficient.

5.4.1 Meta-structure

We can observe a large number of triangle patterns in KG, which can help for relation transitive inference. As shown in Table 5.4, in current version of KG, we extract 115,939 triangle patterns from FB15K, 46,327 triangle pattern existing in DBpedia dataset. And even with only 18 types of relations in total, we can still extract 1,068 triangle patterns from WN18 dataset. Meta-structure is composed of one or multiple triangle patterns between the same pair of entities, which means the amount of Meta-structure in KG is even larger.

In the social network, triangle patterns can be extracted from user graph and page link graph, the extracted triangle patterns is helpful for multiple tasks including user community detection and semantic meaning inference. The triangle pattern in KG has also been proved useful for knowledge relation inference, in this paper, we try to discover more complex transitive inference patterns between relations, we construct Meta-structure, a local structure more complex than triangle pattern, for each pair of entities.

To formulate the proposed Meta-structure, we give a formal definition in Definition. 1

Definition 1. We define each Meta-structure with a pair $(r_o, \mathcal{S})$, $\mathcal{S}$ is a set of triangle patterns $S = \{(a - r_p^\pm - b, b - r_q^\pm - c), r_p, r_q \in \mathcal{R}\}$, $\mathcal{R}$ is the set of all relations. r_p and r_q have directions, either forward or backward in one triangle pattern, r_p^+ represent forward direction $a - r_p - > b$, r_p^- represent backward direction $a < -r_p - b$, similarly, r_q^+ represent forward direction $b - r_q - > c$, r_q^- represent backward direction $b < -r_q - c$.

In this case, there are 4 types of triangle patterns can be used as components of a Meta-structure, as shown in middle part of Fig. 5.3.

5.4.2 Transitive Relation Embedding

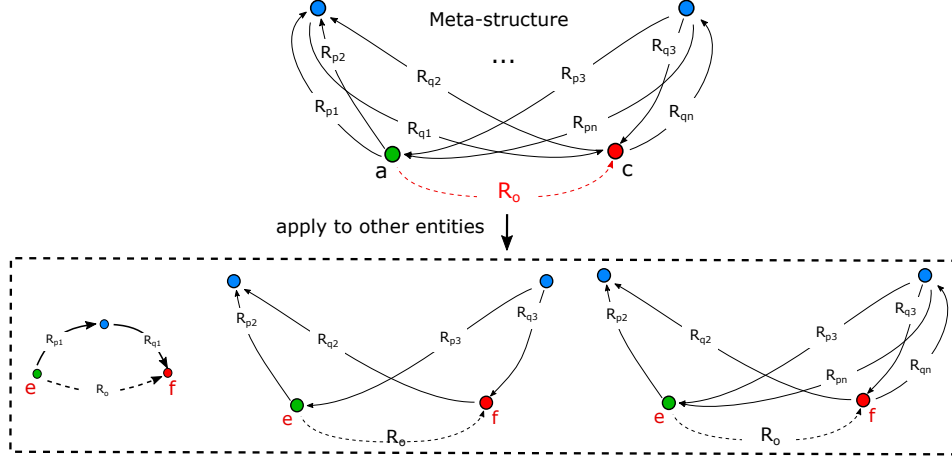


Figure 5.5: Entities a, c are connected through relations r_o , and triangle patterns consists of r_p, r_q . If these three relations, regardless of which entities are connected, are extracted from KG, then we may believe there is a pattern. The pattern is then applied to incomplete triangles to predict the missing relation between new entities e and f .

In Fig. 5.5, we show the inference idea of Meta-structure Relation Transitive Embedding model. Meta-structure is composed of one or multiple triangle patterns between the same pair of entities. To extract the Meta-structure between a pair of entities, the triangle patterns between the entity pair should be extracted, all the extracted triangle patterns share an edge because they are between the same pair of entities, then the meta-structure is composed with all the extracted triangle patterns. We can extract a Meta-structure between entity nodes a and c from the training set as the input of our model, in the output side, we need to predict missing relation between entities, such as e and f , we need to predict missing relation for not only the entity pair contain the same Meta-structure, but also all the entity pair contain Meta-structure consisting of relations appeared in training set.

Our model needs both explicit relation inference ability of Meta-structure, it also needs the ability of generalization, we want to adapt our model for missing relation prediction for a large range of entity pairs, as long as the relations involved with an entity pair is observed in training set and Meta-structure can be found between entity pair, the missing relation can be predicted. We propose an embedding based model, Meta-structure Transitive Relation Embedding (MSTRE), as inference model for Meta-structure for generalization ability, the MSTRE model learns the embedding for each relation of KG, without learning any entity embedding, the model intends to let relation

embeddings contain triangle pattern transitivity information of Meta-structure, and use triangle patterns contained in the same Meta-structure jointly during the embedding learning process.

We define the possibility equation of using one triangle pattern for predicting target relation r_o . The direction of the target relation r_o is fixed, from entity a to entity c , the relation r_p and r_q in triangle pattern have two direction, either forward or backward in each triangle pattern, we have 4 types of triangle pattern here, (r_p^+, r_q^+) , (r_p^+, r_q^-) , (r_p^-, r_q^+) and (r_p^-, r_q^-) , + for forward and - for backward. As shown in Eq. 5.7, we list possibility equations for all cases, we use one k-dimension vector to represent each relation, each relation may act as a different role in different position of the triangle pattern, we assign one matrix M_o to map relation r_o from the original relation space to role-specific space of target relation as V_{r_o} . Similarly, we need to map r_p and r_q from original relation space into their role specific space, but r_p and r_q have two directions, we assign four matrices M_p^+ , M_p^- , M_q^+ and M_q^- for mapping. With r_p and r_q together, we can have a potential triangle pattern, in this case, the mapped role specific space representation of triangle pattern is defined as sum of mapped representation of r_p and r_q , as $V_{r_p^+, r_q^+}$, $V_{r_p^+, r_q^-}$, $V_{r_p^-, r_q^+}$ and $V_{r_p^-, r_q^-}$. With the above role-specific space representation, we can define the possibility that if the triangle pattern occurs then how likely the target relation r_o also occur from entity a to entity c . We list the possibility equations for 4 types of triangle pattern as $P(r_o|r_p^+, r_q^+)$, $P(r_o|r_p^+, r_q^-)$, $P(r_o|r_p^-, r_q^+)$ and $P(r_o|r_p^-, r_q^-)$. We calculate the match score of target relation and triangle pattern $\exp(V_{r_o}^T V_{r_p^+, r_q^+})$ as numerator of fraction, then we replace target relation r_o with any relation r_k in relation list and calculator the sum of match score as denominator.

Multiple triangle patterns are integrated into one complex structure of relation inference, Meta-structure. We need to sufficiently utilize the triangle pattern based transitivity contained in one Meta-structure jointly into one possibility equation for training. Between each pair of entities (a, c) , we can extract a set of triangle patterns $TP(a, c)$, we can construct the Meta-structure between this entity pair by using all triangle patterns in set $TP(a, c)$. Then we define the $P(r_o|a, c)$ in Eq. 5.8 to represent the possibility that if the Meta-structure from a to c consisting of the set of triangle patterns $TP(a, c)$ occurred then target relation r_o also occurred from a to c .

$$\begin{aligned}
 V_{r_o} &= M_o \vec{r}_o \\
 V_{r_p^+, r_q^+} &= M_p^+ \vec{r}_p + M_q^+ \vec{r}_q \\
 V_{r_p^+, r_q^-} &= M_p^+ \vec{r}_p + M_q^- \vec{r}_q \\
 V_{r_p^-, r_q^+} &= M_p^- \vec{r}_p + M_q^+ \vec{r}_q \\
 V_{r_p^-, r_q^-} &= M_p^- \vec{r}_p + M_q^- \vec{r}_q \\
 P(r_o | r_p^+, r_q^+) &= \frac{\exp(V_{r_o}^T V_{r_p^+, r_q^+})}{\sum_k^{|\mathcal{R}|} \exp(V_{r_k}^T V_{r_p^+, r_q^+})} \\
 P(r_o | r_p^+, r_q^-) &= \frac{\exp(V_{r_o}^T V_{r_p^+, r_q^-})}{\sum_k^{|\mathcal{R}|} \exp(V_{r_k}^T V_{r_p^+, r_q^-})} \\
 P(r_o | r_p^-, r_q^+) &= \frac{\exp(V_{r_o}^T V_{r_p^-, r_q^+})}{\sum_k^{|\mathcal{R}|} \exp(V_{r_k}^T V_{r_p^-, r_q^+})} \\
 P(r_o | r_p^-, r_q^-) &= \frac{\exp(V_{r_o}^T V_{r_p^-, r_q^-})}{\sum_k^{|\mathcal{R}|} \exp(V_{r_k}^T V_{r_p^-, r_q^-})} \\
 r_o, r_p, r_q &\in \mathcal{R}
 \end{aligned}
 \tag{5.7}$$

$$P(r_o | a, c) = \prod_{r_p, r_q}^{TP(a, c)} P(r_o | r_p, r_q)
 \tag{5.8}$$

5.4.3 Training and Prediction

To train the model, we need to extract existing triplets a, r_o, c from entire current KG, and maximize the possibility $P(r_o | a, c)$ of existing triplets, because we assume the triplets existing in current KG is true and accurate. To simplify the likelihood equation, we maximize the $\log[P(r_o | a, c)]$ instead of $P(r_o | a, c)$, as shown in Eq. 5.9, we have $\log[P(r_o | a, c)]$, and transfer the cumulative product operation into sum operation.

$$\begin{aligned}
 \log[P(r_o | a, c)] &= \log \left[\prod_{r_p, r_q}^{TP(a, c)} P(r_o | r_p, r_q) \right] \\
 &= \sum_{r_p, r_q}^{TP(a, c)} \log[P(r_o | r_p, r_q)]
 \end{aligned}
 \tag{5.9}$$

We define the final likelihood equation for training in Eq. 5.10, $\mathcal{D}$ is defined as the set of all existing triplets a, r_o, c in current KG, we pick up each a, r_o, c to calculate

Algorithm 5 Training With Meta-structure Transitive Relation Embedding

Input: Knowledge Graph.

Training:

Step 1: Draw subject-predicate-object triplets set $\mathcal{D}$ from Knowledge Graph.

Step 2: Repeat

for each (subject entity a , predicate relation r_o , object entity c) $\in \mathcal{D}$ **do**

Get all 4 types of potential triangle patterns which link subject entity a and object entity c

$a - r_p^+ - > e_k - r_q^+ - > c, a - r_p^+ - > e_k < -r_q^- - c, a < -r_p^- - e_k - r_q^+ - > c, a < -r_p^- - e_k < -r_q^- - c$

$k \in |\mathcal{E}|, p \in |\mathcal{R}|, q \in |\mathcal{R}|$, $\mathcal{E}$ is set of all entities, $\mathcal{R}$ is set of all relations

Conclude Triangle Patterns list for r_o as $\mathcal{S}$

$\mathcal{S} = \{(r_{p1}^{+/-}, r_{q1}^{+/-}), (r_{p2}^{+/-}, r_{q2}^{+/-}), (r_{p3}^{+/-}, r_{q3}^{+/-}), \dots\}$

$+$ means forward, $-$ means backward, each r_p or r_q in the list can only be either forward or backward.

for each (relation r_p , relation r_q) $\in \mathcal{S}$ **do**

Calculate possibility that if r_p and r_q occur then r_o occur.

$$P(r_o|r_p, r_q) = \frac{\exp\langle V_{r_o}, V_{r_p, r_q} \rangle}{\sum_k |\mathcal{R}| \exp\langle V_{r_k}, V_{r_p, r_q} \rangle}$$

Calculate and maximize the likelihood equation

Maximize $\mathcal{L} = \sum_{\mathcal{S}}^{r_p, r_q} \log P(r_o|r_p, r_q)$

$\log[P(r_o|a, c)]$, then sum them up to get final likelihood equation. By maximizing the likelihood equation $\mathcal{L}$, we can jointly learn proper embedding for each relation in KG and then to give accurate missing relation prediction. Details of the training process are described in Alg. 5.

$$(5.10) \quad \mathcal{L} = \sum_{a, r_o, c}^{\mathcal{D}} \log[P(r_o|a, c)]$$

The prediction equation for predicting missing relation is not exactly the same as possibility equation used in the training process. In training process, given a pair of entities a and c , we can extract the Meta-structure from a to c , we define the possibility of using a Meta-structure for target relation inference $P(r_o|a, c)$, as cumulative product of the possibility of using triangle patterns in Meta-structure to infer target relation, $P(r_o|r_p, r_q)$. If we still use possibility $P(r_o|a, c)$ for target relation inference in prediction, there is a problem, $0 \leq P(r_o|r_p, r_q) \leq 1$, more triangle patterns contained in one Meta-structure may cause the possibility $P(r_o|a, c)$ smaller, which is obvious wrong, with more triangle patterns contained in one Meta-structure, the inference for target relation should be more confident. We define possibility equation $P_{pred}(r_o|a, c)$ for prediction

Algorithm 6 Prediction With Meta-structure Transitive Relation Embedding

Input: Knowledge Graph.

Prediction:

Step 1: Draw subject-object pairs set as $\mathcal{T}$.

Step 2: Repeat

for each (subject entity a , object entity c) $\in \mathcal{T}$ **do**

 Get all 4 types of potential triangle patterns which link subject entity a and object entity c

$a - r_p^+ - > e_k - r_q^+ - > c, a - r_p^+ - > e_k < -r_q^- - c, a < -r_p^- - e_k - r_q^+ - > c, a < -r_p^- - e_k < -r_q^- - c$

$k \in |\mathcal{E}|, p \in |\mathcal{R}|, q \in |\mathcal{R}|$, $\mathcal{E}$ is set of all entities, $\mathcal{R}$ is set of all relations

 Conclude Triangle Patterns list for (a, c) pair as $\mathcal{S}$

 Calculate possibility that target relation r_o occur from a to c

$P_{pred}(r_o|a, c) = 1 - \prod_{r_p, r_q}^{\mathcal{S}} [1 - P(r_o|r_p, r_q)]$

process in Eq. 5.11, given an entity pair (a, c) , we extract Meta-structure as set of triangle patterns $TP(a, c)$, we assume that if all the triangle patterns in $TP(a, c)$ don't occur, then target relation r_o also doesn't occur. In this case the non-occurring possibility that r_o should be cumulative product of non-occurring possibility of each triangle patterns, $poss_{unoccur} = \prod_{r_p, r_q}^{TP(a, c)} [1 - P(r_o|r_p, r_q)]$, then naturally, the occur possibility of r_o should be $1 - poss_{unoccur}$. Details of the training process are described in Alg. 6.

$$(5.11) \quad P_{pred}(r_o|a, c) = 1 - \prod_{r_p, r_q}^{TP(a, c)} [1 - P(r_o|r_p, r_q)]$$

5.4.4 Joint Prediction Strategy

In some special cases, there exists no triangle pattern between a pair of entities, of course, we can't build valid Meta-structure in this kind of situation, without Meta-structure, our MSTRE model can't give a prediction for missing relation, which leaves blank in our final prediction result. In this section, we give a joint prediction strategy to fill in the blank of our model. with results of baseline models, TransE, TransH, TransR, TransD, RESCAL, DistMult and ComplEx. The final prediction result comparison will between baselines and MSTRE filled with baselines.

In entity link prediction and relation link prediction tasks, we target to predict possibility of a given entity-relation-entity triple, (a, r_o, c) , exists or not. More specifically, we want to find the missing relation from a to c . For each triple (a, r_o, c) , we have two entities, a and c , r_o is the target relation from a to c , we extract all the triangle patterns

from a to c to build Meta-structure. If there exists a triangle pattern between the entity pair (a, c) , we can build Meta-structure, we can compute the occur possibility that target relation based on the Meta-structure relation inference. Otherwise, if there is no triangle pattern from a to c , MSTRE model can't give the prediction, we use the results of baselines to fill in the blank of our result. Our proposed joint prediction strategy uses the same result of baselines if there is no triangle pattern from a to c , and we use the results of Meta-structure based relation inference for the entity pair with the triangle pattern and try to achieve better overall performance than baselines.

Table 5.3: Space and Time Complexity of MSTRE and baselines

	Space complexity	Time complexity
TransE	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
TransH	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
TransR	$\mathcal{O}(nd + mdk)$	$\mathcal{O}(dk)$
TransD	$\mathcal{O}(nd + mk)$	$\mathcal{O}(\max(d, k))$
RESCAL	$\mathcal{O}(nd + md^2)$	$\mathcal{O}(d^2)$
DistMult	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
ComplEx	$\mathcal{O}(nd + md)$	$\mathcal{O}(d)$
MSTRE	$\mathcal{O}(mk)$	$\mathcal{O}(k^2)$

5.4.5 Advantages of Proposed Model

Our proposed model has a few differences and advantages comparing with traditional entity embedding required models and some previous relation based models.

Outperform on KG with infrequent entities. In KG, a considerable number of entities appear infrequently, only a limited number of times or even just once or twice. For these infrequent entities, it is hard for traditional entity embedding required model to give accurate link prediction result, especially result of predicting missing relation between entity pair. Because of using transitivity information among relation only, without learning the embeddings for entities, the occurrence frequency of a single entity is irrelevant during the prediction process of proposed model MSTRE. As long as there is valid Meta-structure containing relation transitivity information between entity pair, a quite accurate prediction can be obtained to determine which relation is likely to appear between subject and object in entity pair.

Sufficiently use relation inference patterns with Meta-Structure. Based on representing the relation transitivity in the form of a special structure defined in this

paper, Meta-structure, our proposed model MSTRE can sufficiently use the relation inference patterns. Given an entity pair, to predict the relation between entities, we need to extract triangle patterns containing relation transitivity information, there may exist multiple triangle patterns between entities. If we consider each triangle pattern alone in the training process, then the interaction between triangle patterns is not considered, some target relation needs to be inferred by multiple triangle patterns jointly. In MSTRE, Meta-structure is used to jointly represent multiple triangle pattern between the same pair of entities, the interaction between triangle patterns and multiple triangle pattern joint inferences are both considered during the training process.

Interpretability. MSTRE shows good interpretability as it predicts missing relation the link between entities based on Meta-structure consisting of multiple triangle patterns. One final occur possibility of target relation can be computed based on joint Meta-structure inference, furthermore, by computing the target relation occur possibility with each single triangle pattern in Meta-structure, we can explicitly tell the contribution of each triangle pattern for final joint inference.

Embedding based model for generalization. Besides the advantage of high interpretability because of using Meta-structure for relation inference, MSTRE also takes generalization advantage from embedding model. MSTRE learns the embedding for each relation, and then apply the relation embedding for Meta-structure based relation inference. As long as the relations in test set appeared in the training set, we can do a prediction for the test set. Even although the Meta-structures in the test set haven't appeared in the training set, we can obtain the representation of Meta-structure based on relation embeddings.

Efficient parameter learning. The proposed model MSTRE learns relation embedding only, without learning any entity embedding, which largely reduces the parameter number of the model. We analyze the space and time complexity of MSTRE model and baselines, as shown in Table 5.3, comparing with traditional entity embedding required models, the training process of MSTRE with fewer parameters is obviously more efficient.

5.5 Experiment and Analysis

To test performance of Transitive Relation Embedding (TRE) model and Meta-structure Transitive Relation Embedding (MSTRE) model, we use several entity embedding required knowledge base embedding models, which are popular KB Embedding models used in previous works, as our baselines including TransE[12], TransH[90], TransR[44],

RESCAL[59], TransD[36], DistMult[95], and ComplEx[91].

Table 5.4: Dataset Size of FB15k, WN18 and DBP

	Entity count	Relation count	Triple count	Triangle pattern count	Avg. entity occurrence	Med. entity occurrence
FB15K	14,951	1,346	483,142	115,939	64.63	41
WN18	40,943	18	141,442	1,068	6.91	4
DBP	376,941	566	432,760	46,327	2.27	1

We run the TRE model and baselines on several widely used KB datasets, including FB15K and WN18. We also construct an extremely sparse dataset by extracting subset from entire DBpedia project, we call this dataset “DBP” in the experiment.

We compare the KG completion performance of MSTRE and baselines on three KG datasets, FB15K, WN18, and DBpedia, in the experiments as well, each KG dataset has its distinguishing feature, the size of datasets listed in Table 5.4.

The sizes of datasets are listed in Tab. 5.4, we also list average and median time of entities occur in training dataset in the last two columns of the table. We can see that DBP is far more sparse than the other two datasets.

FB15K dataset is a common KG dataset and widely used in many applications and experiments of previous works, rich semantic information is contained in FB15K, the number of triangle pattern extracted from FB15K is the largest in three datasets, which indicates that FB15K contains strong relation inference patterns.

WN18 is also a popular dataset for testing KG completion task, with only 18 types of relations in it, a limited number of valid triangle pattern can be extracted to construct useful Meta-structure, we still want to check the performance of MSTRE on such kind of dataset with extremely few relation types.

We also directly extract a subset of DBpedia open-source KG without manual filtering or cleaning, we call it “DBpedia” in our experiment, also, the entity number of DBpedia dataset is far more than the other two datasets, the relation number is less, and the triangle patterns extracted is also less. DBpedia dataset is extremely sparse than the other two datasets, most of the entities only appeared once in training set, the number of triangle patterns is also limited.

The two DBP datasets used in experiments of TRE and MSTRE are two different datasets, the two datasets are both extracted from the same original DBpedia subset, while the different data filtering strategies and the different training test data splitting methods are used to obtain two final proper datasets for TRE and MSTRE experiments respectively. Although the FB15K and WN18 datasets used for testing TRE and MSTRE

are same, the results of the baseline for them also have a minor difference because of the different training iteration numbers and the different parameter initial process.

5.5.1 Entity Link Prediction

Entity Link Prediction with Transitive Relation Embedding. To compare the performance of TRE model with baselines direct, we do an experiment for entity link prediction under the framework of previous works, predicting tail entity $(h, r, ?)$ and predicting head entity $(?, r, t)$. Given a pair (h, r) or (r, t) , h is head entity of triple, r is relation, t is tail entity, our task is to predict the missing part of triple, to predict t for (h, r) and to predict h for (r, t) . For tail entity prediction, $(h, r, ?)$, we fill tail entity with any entity e in KG, and rank the entities with the probability that the triple (h, r, e) is true, the triple with the higher rank is more likely to be true. Similarly, for head entity prediction, $(?, r, t)$, we rank the triples (e, r, t) and determine which triples are more likely to be true.

For triple ranking, we need to compute the score of each triple. In baselines, we compute the score of the triple with the score function of baselines, we rank the triples in ascending order, a low score indicates high rank. In the proposed model, we first detect all potential triangle pattern between pair $(h, e)/(e, t)$, then we sum up all the probability that r occurs between $(h, e)/(e, t)$ as a score of triple, we rank the triples in descending order, a high score indicates high rank. With the ranking result, we use three evaluation methods, Mean Rank (MR), Mean Reciprocal Rank (MRR) and Hit@10. The target of the prediction task is to achieve low MR, high MRR, and high Hit@10.

In Tab. 5.5, we show the entity link prediction results on FB15K dataset. In Tab. 5.6, we show the entity link prediction results on WN18 dataset. The left part of the tables contain the result of tail entity prediction, $(h, r, ?)$, the right part of tables contain the result of head entity prediction, $(?, r, t)$. By comparing the prediction result of baselines and joint prediction result of baselines and TRE, we find that joint prediction result outperforms the baselines on both two datasets. For each baseline, we compare the results for baseline and TRE+baseline, the bold font results mean the better one between baseline and TRE+baseline, the results with a star represent the best result among all 14 methods including 7 baseline methods and 7 TRE+baseline methods. As we can see, in entity link prediction, joint prediction (TRE+baseline) result outperforms the baselines on both two datasets.

Entity Link Prediction with Meta-structure based Transitive Relation Embedding. We also do entity link prediction on FB15K dataset in an experiment designed

Table 5.5: Result of FB15K Entity Prediction with TRE

	$(h, r, ?)$						$(?, r, t)$					
	Baseline			Baseline+TRE			Baseline			Baseline+TRE		
	MRR	MR	Hit@10	MRR	MR	Hit@10	MRR	MR	Hit@10	MRR	MR	Hit@10
TransE	.2371	222.22	.4355	.5444	60.17	.7974	.1786	346.65	.3536	.4789	83.51	.7184
TransH	.2317	234.33	.4222	.5442	60.54	.7965	.1733	364.00	.3428	.4784	85.57	.7176
TransR	.2428	209.27	.4422	.5451	55.74	.7986	.1822	347.35	.3627	.4794	82.67	.7201
RESCAL	.1519	523.31	.2615	.5392	114.90	.7825	.1015	806.08	.1879	.4741	163.55	.7054
TransD	.2307	244.57	.4193	.5441	62.56	.7967	.1735	375.40	.3404	.4785	86.25	.7173
DistMult	.1904	231.57	.3603	.5425	58.32	.7909	.1370	334.54	.2852	.4768	84.14	.7130
ComplEx	.2430	207.30	.4671	.5467*	52.05*	.8043*	.1871	296.30	.3899	.4811*	71.82*	.7257*

Table 5.6: Result of WN18 Entity Prediction with TRE

	$(h, r, ?)$						$(?, r, t)$					
	Baseline			Baseline+TRE			Baseline			Baseline+TRE		
	MRR	MR	Hit@10	MRR	MR	Hit@10	MRR	MR	Hit@10	MRR	MR	Hit@10
TransE	.2424	625.15	.4564	.3649	582.83	.5422	.2184	614.51	.4202	.3400	555.27	.5072
TransH	.0378	2974.43	.073	.1922	2572.43	.229	.0400	2948.18	.0716	.1933	2500.31	.2262
TransR	.2776	469.16	.5268	.3924	443.12*	.5954	.2537	482.39	.5114	.3698	441.37*	.581
RESCAL	.0408	7172.78	.0722	.1962	6167.40	.2278	.0584	6677.36	.0984	.2101	5777.30	.252
TransD	.2208	769.21	.3986	.3471	706.03	.4946	.2042	850.86	.3752	.3323	747.70	.4782
DistMult	.3226	761.08	.5866	.4333	703.62	.655	.2966	767.61	.5602	.4091	709.83	.634
ComplEx	.5627	819.72	.8012	.6351*	719.08	.8328*	.5399	839.39	.781	.6177*	730.66	.818*

for evaluating MSTRE model. For baselines, the score of filled triple (h, r, t) can be computed as score function, and for MSTRE, occur possibility, $P(r|h, t)$, of relation r based on entity pair (h, t) is computed as score, a high score indicates high rank. We use MR, MRR, and Hit@K for prediction result evaluation as same as previous works.

We pick up a part of data for entity link prediction task, the (h, r) and (r, t) pairs picked up have at least 1 but no more than 10 entities e , which can match $P(r|h, e) \geq threshold$ or $P(r|e, r) \geq threshold$. MSTRE can't entirely replace baselines in entity link prediction task, we only improve the prediction result for (h, r) and (r, t) pairs that we can ensure the relation predicted by MSTRE is very likely to be right.

Table 5.7: Result of FB15K Entity Prediction with MSTRE

	$(h, r, ?)$						$(?, r, t)$					
	Baseline			MSTRE+Baseline			Baseline			MSTRE+Baseline		
	MR	MRR	Hit@10	MR	MRR	Hit@10	MR	MRR	Hit@10	MR	MRR	Hit@10
TransE	67.0217	0.3738	0.6218	65.2774	0.3834	0.6299	119.2389	0.2987	0.5307	115.8628	0.2976	0.5377
TransH	68.6098	0.3769	0.6289	66.8542	0.3832	0.6305	119.3314	0.3010	0.5319	116.9423	0.2972	0.5371
TransR	60.5363	0.3796	0.6380	59.3535	0.3852	0.6390	109.8846	0.3057	0.5420	107.9336	0.2990	0.5442
RESCAL	67.8642	0.3754	0.6254	66.1500	0.3829	0.6294	120.7185	0.2958	0.5305	118.2567	0.2965	0.5366
TransD	283.3099	0.2745	0.4340	225.4346	0.3593	0.5446	438.9827	0.2128	0.3619	382.7724	0.2752	0.4584
DistMult	76.3043	0.3159	0.5475	69.1444	0.3738	0.5909	135.3584	0.2497	0.4681	127.9309	0.2879	0.4991
ComplEx	54.0884	0.3834	0.6849	50.5660	0.3904	0.6626	100.8783	0.3147	0.5976	97.5188	0.3052	0.5772

As shown in Table 5.7, we test MSTRE with baselines on FB15K dataset for entity link prediction task, we can see that in object entity prediction $(h, r, ?)$, the left part of the table, the joint prediction result of MSTRE and baseline, MSTRE+baseline, is better than most of the baseline result, the Hit@10 result of ComplEx is slightly better than MSTRE+baseline, but still falls in MR and MRR result. And in the right part of the table, the result of subject entity prediction $(h, r, ?)$ is listed, the MRR result is close, but the MR result of MSTRE+baseline is better than baselines, and most of the Hit@10 results of MSTRE+baseline is better than baselines.

5.5.2 Relation Link Prediction

Relation Link Prediction with Transitive Relation Embedding. We test relation link prediction, predicting $(h, ?, t)$, to show that the TRE model is capable to predict new relations between entities. Given an entity pair (h, t) , the task is to predict the relation r between h and t . For each pair (h, t) , we fill triple $(h, ?, t)$ with any relation r , the score of the triple (h, r, t) is computed as same as in entity link prediction. We rank the triples to determine which triples are more likely to be true, we also use MR, MRR, and Hit@10 for result evaluation.

Table 5.8: Result of FB15K, WN18 and DBP Relation Prediction with TRE $(h, ?, t)$

	FB15K						WN18						DBP					
	Baseline			Baseline+TRE			Baseline			Baseline+TRE			Baseline			Baseline+TRE		
	MRR	MR	Hit@10	MRR	MR	Hit@10	MRR	MR	Hit@10	MRR	MR	Hit@10	MRR	MR	Hit@10	MRR	MR	Hit@10
TransE	.5049	63.99	.7256	.6730	19.54	.8882	.5833	3.61	.9398	.6100	3.34	.9466	.0121	258.56	.0150	.1827*	192.50*	.2265
TransH	.5281	55.47	.7494	.6780*	20.37	.8691	.1568	10.68	.3764	.2372	9.52	.4666	.0124	273.02	.0169	.1824	201.92	.2263
TransR	.4169	194.95	.5998	.6550	37.72	.8638	.4582	4.33	.9628	.4976	4.04	.9618	.0119	267.27	.0159	.1821	198.93	.2260
RESICAL	.4314	11.47	.7969	.6629	8.47*	.9035*	.6417	3.52	.9046	.6486	3.43	.9136	.0112	284.79	.0144	.1820	210.73	.2254
TransD	.5115	67.04	.7304	.6743	22.19	.8856	.6891	3.52	.8404	.7060	3.23	.8716	.0120	258.92	.0165	.1824	192.57	.2270
DistMult	.0855	44.77	.2120	.5959	15.68	.7972	.7669	1.84	.9764*	.7489	2.04	.9728	.0124	275.04	.0194	.1823	204.69	.2278*
CompLex	.4442	19.89	.8467	.6591	11.04	.9010	.9493*	1.48*	.9752	.9016	1.75	.9716	.0117	270.99	.0162	.1817	202.58	.2256

Table 5.9: Result of FB15K, WN18 and DBpedia Relation Prediction with MSTRE $(h, ?, t)$

	FB15K						WN18						DBP					
	Baseline			MSTRE+Baseline			Baseline			MSTRE+Baseline			Baseline			MSTRE+Baseline		
	MR	MRR	Hit@10	MR	MRR	Hit@10	MR	MRR	Hit@10	MR	MRR	Hit@10	MR	MRR	Hit@10	MR	MRR	Hit@10
TransE	68.3522	0.5084	0.7260	16.9821	0.7526	0.9450	3.6986	0.5822	0.9620	3.2182	0.6312	0.9672	259.0124	0.1719	0.2143	252.2866	0.1901	0.2357
TransH	57.5110	0.5300	0.7520	14.6896	0.7546	0.9483	8.6422	0.1902	0.5718	7.7096	0.2843	0.6196	382.2455	0.0958	0.1137	369.9285	0.1171	0.1398
TransR	198.6166	0.4101	0.5962	33.8362	0.7326	0.9226	3.5524	0.4819	0.9738	3.2222	0.5356	0.9764	277.6007	0.1482	0.1883	268.9491	0.1683	0.2120
RESICAL	70.9024	0.4986	0.7194	16.9188	0.7506	0.9450	3.2830	0.5886	0.9836	2.8928	0.6340	0.9864	266.1769	0.1700	0.2109	259.2638	0.1890	0.2326
TransD	19.3125	0.4057	0.6489	4.9231	0.7491	0.9507	4.8782	0.5435	0.8262	4.3890	0.5813	0.8514	157.4593	0.3064	0.4182	152.2376	0.3231	0.4363
DistMult	18.2110	0.2472	0.6316	5.3232	0.7140	0.9436	1.8692	0.7626	0.9744	1.8722	0.7590	0.9772	272.9531	0.1764	0.2625	263.2182	0.1972	0.2863
CompLex	25.6853	0.3207	0.6642	7.8102	0.7173	0.9310	1.4240	0.9547	0.9772	1.5120	0.9246	0.9774	257.0301	0.1542	0.2335	248.0354	0.1743	0.2571

We can see on FB15K and DBP, the TRE+baseline methods outperform the baseline methods, however, on WN18, we observe that some result baseline methods are slightly better than TRE+baseline methods, this is because there are only 18 relations in WN18 dataset, we can extract a limited number of triangle pattern for training. As shown in Tab. 5.4, the entity size and triple size of WN18 is close to the other two datasets, but the relation size is extremely small, which causes the triangle pattern extracted is extremely less than the other two datasets. The meaning of bold font and star is as same as in entity link prediction task.

Relation Link Prediction with Meta-structure based Transitive Relation Embedding. Relation link prediction task is also tested in an experiment designed for MSTRE evaluation. For baselines, the score of filled triple (h, r, t) can be computed as score function, and for MSTRE, the filled triple score can be computed as occur possibility, $P(r|h, t)$, of relation r based on entity pair (h, t) , a high score indicates high rank. We use mean rank (MR), mean reciprocal rank (MRR) and Hit@K to evaluate the prediction result.

As shown in Table 5.9, we list the result of using 7 baselines on three datasets, and also the result that joint prediction with MSTRE and baselines (MSTRE+baseline), the task here is to prediction target relation r for a given entity pair (h, t) . We compare the Hit@10 and MRR result separately and visualize the result in histogram in Fig. 5.6 and Fig. 5.7. On DBpedia dataset, the MSTRE+baseline has a better result than baselines, and on FB15K dataset, the result of MSTRE+baseline largely outperform the baselines. On WN18, the results of MSTRE+baseline and baselines are close, some MRR result of baselines even slightly better than MSTRE+baseline, because with only 18 types of relations in WN18, number of the useful triangle pattern is limited, in this case, we think the result is predictable and acceptable.

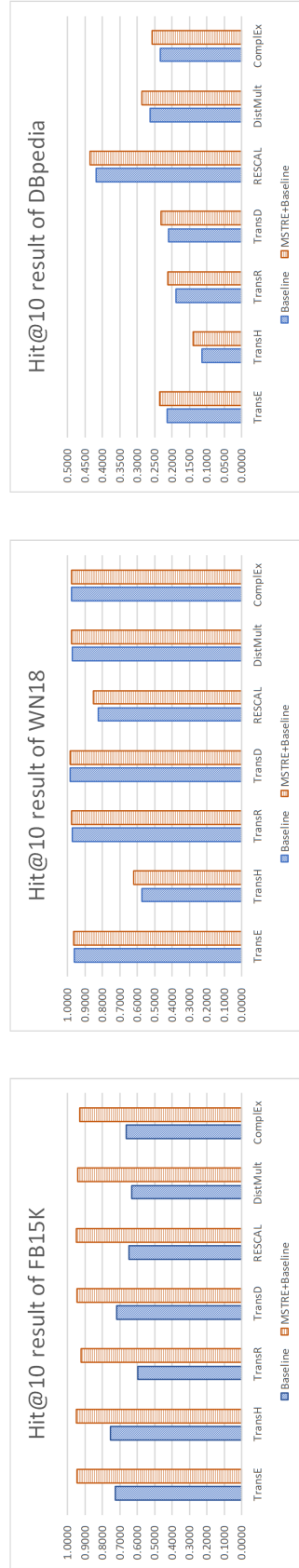


Figure 5.6: Hit@10 result comparison between baselines and MSTRE filled blank with baselines (MSTRE+baseline).

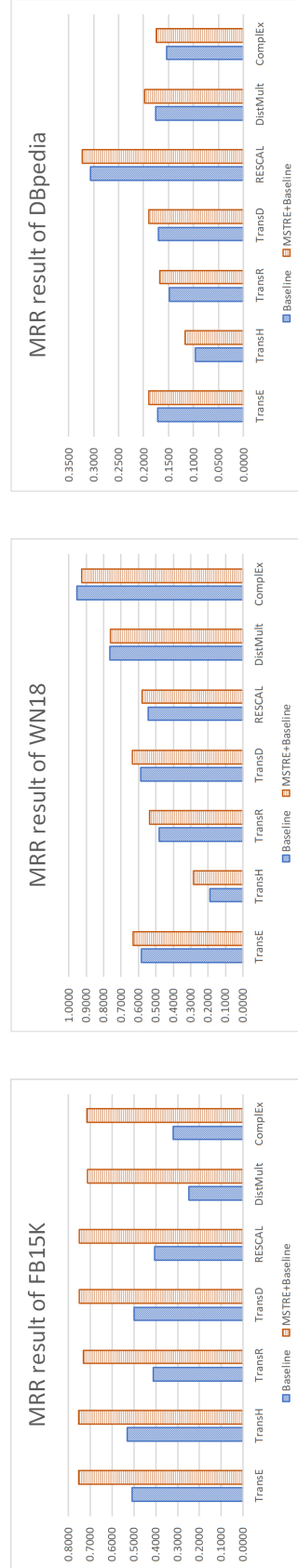


Figure 5.7: MRR result comparison between baselines and MSTRE filled blank with baselines (MSTRE+baseline).

To compare the performance of baselines and MSTRE more clearly, we filter out all the entity pairs with Meta-structure, and test MSTRE and baselines on these filtered pairs. Because the entity pairs without valid Meta-structure will use the same result as baselines, the difference between results of MSTRE and baseline actually is on filtered entity pairs. In Table 5.10, we list the MR, MRR, Hit@10 results, we can clearly see that the result of MSTRE obviously outperform the baselines, on filtered data of FB15K and DBpedia. Still predictably, because of the limited number of relation types, on WN18 dataset, the Hit@10 results of MSTRE and baseline are close, some baselines have a small advantage in MRR result.

Table 5.10: Result of FB15K, WN18 and DBpedia Relation Prediction ($h, ?, t$) for Entity Pairs with MSTRE

	FB15K			WN18			DBP		
	MR	MRR	Hit@10	MR	MRR	Hit@10	MR	MRR	Hit@10
TransE	63.8048	0.4962	0.7201	4.6340	0.4885	0.9629	233.3910	0.1695	0.2351
TransH	53.6493	0.5195	0.7471	7.3385	0.2184	0.7081	416.7223	0.0694	0.0828
TransR	198.5310	0.4031	0.5925	3.7356	0.4604	0.9785	296.5371	0.1085	0.1608
RESCAL	66.9096	0.4868	0.7121	4.0945	0.5096	0.9773	239.5311	0.1453	0.2253
TransD	19.8734	0.3782	0.6217	4.6866	0.5552	0.8433	184.0719	0.2175	0.3435
DistMult	18.0896	0.2316	0.6096	1.7428	0.8025	0.9773	332.0536	0.0835	0.1571
ComplEx	24.0143	0.3150	0.6633	1.2344	0.9612	0.9928	307.7856	0.1093	0.1657
MSTRE	2.7795	0.7862	0.9803	1.7608	0.7811	0.9940	12.8551	0.7671	0.9367

5.5.3 Accurate Prediction on Extremely Sparse KG with Transitive Relation Embedding

We do the relation link prediction on this sparse dataset DBP. The result shows that the TRE model has a clear advantage than other baselines when they deal with the sparse dataset. From Tab. 5.4, we can see that the average occurrence of each entity in DBP is far less than the other two datasets, which makes the dataset DBP extremely sparse. We test the baseline methods and TRE+baseline methods, the result shows that our proposed model has a large advantage, the baseline methods perform poorly, however, the predictions of the proposed model are still accurate.

For further testify our assumption, we also extract a sparse subset of FB15K to show the capability of the proposed method on sparse KB data through leveraging transitive relation inference. We extract a sparse subset of FB15K by extracting the triples meeting following conditions, at least one entity occurred less than 5 times in training dataset, at least one triangle pattern existing between two entities. Comparing result in Tab. 5.11

Table 5.11: Result of Sparse FB15K Relation Prediction with TRE

	MRR	MR	Hit@10
TransE	.3632	204.63	.5295
TransH	.3867	174.89	.5810
TransR	.3322	287.74	.4668
RESCAL	.3713	27.88	.6438
TransD	.3663	199.49	.2659
DistMult	.0414	78.20	.1128
ComplEx	.4488	42.01	.8088
TRE	.6429	14.84	.8723

with the FB15K result in Tab. 5.8, we can find that the results of baseline methods fall largely on at least one evaluations, however, the TRE results are barely influenced by sparsity.

5.6 Summary

In this section, firstly, we proposed TRE, a new embedding model using the transitivity of Knowledge Base relations to efficiently solve KB sparsity problem. To take advantages of Knowledge Base relation transitivity, we extract the relation triangle pattern from large-scale Knowledge Bases. We measure the reliability of Knowledge Base relation inference rule with confidences of relation triangle patterns, which are used as input to train TRE model. Then, we propose a new embedding model for relation transitivity based inference, Meta-structure Transitive Relation Embedding (MSTRE). MSTRE takes advantage of Meta-structure based relation inference, which can give more accurate relation prediction for entity pairs involving infrequent entity. By using embedding based model, MSTRE learns the embedding for each relation and complex Meta-structure can be represented by relation embeddings, as long as the relation sets of the training set and test set are consistent, the missing relation prediction can be done, the generalization ability of MSTRE is high.

The TRE and MSTRE models are tested on 3 KG datasets, along with 7 popular baselines in the field, the advantages of models are evaluated in the experiment. We evaluate TRE and MSTRE models with two tasks, entity prediction and relation prediction, the proposed model outperforms baselines. We especially test proposed models on the sparse dataset, the advantage of proposed models is greater. Because of using relation triangle pattern statistics as training data, the entity occurrence frequency is

irrelevant, the proposed model can achieve a good result on sparse data. The possibility of each given triangle pattern relation inference rule can be explicitly computed, which makes the prediction results of both TRE and MSTRE interpretable. Without computing entity embeddings, the parameter numbers of TRE and MSTRE are both far less than Traditional Knowledge Graph Embedding models, the training process is efficient.

HIERARCHICAL COLLABORATIVE EMBEDDING FOR KNOWLEDGE GRAPH AND RECOMMENDER SYSTEM

6.1 Introduction

Personalized recommendation plays an important role in today's business. By observing user behaviors, recommender systems can identify items with the potential to be liked by users. One of the most popular recommendation techniques is collaborative filtering (CF) which is based on the intuition that preference structures can be transferred across like-minded users. However, CF suffers from the cold-start data sparsity problem in which users provide a limited amount of preferences, e.g., preference data of newly registered users are quantitatively sparse, making recommendation inaccurate. Particularly, in some real recommendation scenarios, user preferences are often quantitatively sparse because of the application of nature. For example, unlike watching many movies, users typically can only study a few subjects in *Coursera*¹ or contribute to a few repositories (projects) on *GitHub*², both of which contain very high numbers of subjects or projects.

To address the sparsity problem, a common practice is to couple preferences with auxiliary information such as demographic attributes and content information, into hybrid recommender systems [2]. Researchers have proposed to extend the spare data by connecting to external knowledge graphs [15, 74, 103, 105]. This approach leverages both

¹Online course platform <https://www.coursera.org/>

²Project hosting platform <https://github.com/>

network structure and text info embedded in knowledge bases to supplement traditional CF. To be specific, a knowledge base is a data repository containing interlinked entities across different domains. Since knowledge base is often represented in a graph way, it is also called a knowledge graph (KG). The beauty of the knowledge graph is not only the textual knowledge representations, but also the linked structure of knowledge entities. Recently, the knowledge graph has emerged as a new method in recommender systems research. For example, latent features are often extracted from heterogeneous information network to represent users and items [15, 74, 103]. More recently, Zhang et al. [105] proposed the first work, Collaborative Knowledge base Embedding (CKE), to build a hybrid heterogeneous information a network containing both the recommender system and knowledge graph. CKE-based methods learn entity/relation embedding representations via TransE/TransR [12, 44] and user/item embedding representations via Matrix Factorization [68].

Although the aforementioned recommendation tasks exhibit the significant sparsity, we argue that the user choices/behaviors carry on rich semantics info which has not been fully utilized in the recommendation. For example, knowing a user’s interest in a subject or repository reveals lots of information about this user, such as preferences over *programming language, operating system, field of study, research topic*. From the knowledge graph view, such information pieces are not isolated and fragmented, instead, interrelated, forming a comprehensive view of this author. This rich semantic information can play an important role in alleviating such cold-start and sparsity problem, therefore using KG-based approaches becomes an ideal solution to this kind of tasks. However, previous studies on KG-based CF suffer from one or more of the following limitations: 1) rely on tedious feature engineering; 2) the high data sparsity; and 3) recommended items need to be one exact entity within the knowledge base.

In this chapter, we first propose a novel collaborative recommendation framework to integrate the recommender system and knowledge graph with an extensible connection between items and knowledge entities. Overall, our method constructs a multi-level network via knowledge graph to enhance sparse semantic information between users and items. Let’s take example of *GitHub* recommendation task shown in Fig. 6.1 to show how our model works. In order to reveal the latent correlations between *GitHub* repositories, the names of which are not existent in the knowledge graph, we frame the integrated system into 3-level, where users, repositories, knowledge graph entities are placed in different levels, with edges between users and repositories indicating the user has interest in the repository, edges between repositories and entities indicating that

the entity is possibly related to the repository, and edges between entities meaning there is at least one specific relationship between this pair of entities. A hierarchical structure heterogeneous network, which contains multiple types of nodes and multiple types of edges, is built for automatic collaborative learning. Particularly, to link the recommender system and knowledge graph properly, knowledge conceptual level is proposed to indirectly map item-entities, different from previous works of direct mapping. Serving as a middle level of three-level hierarchical structure model, the knowledge conceptual level can fully interconnect the whole system in a proper way, tackling the restriction that recommendation items need to be within the knowledge base.

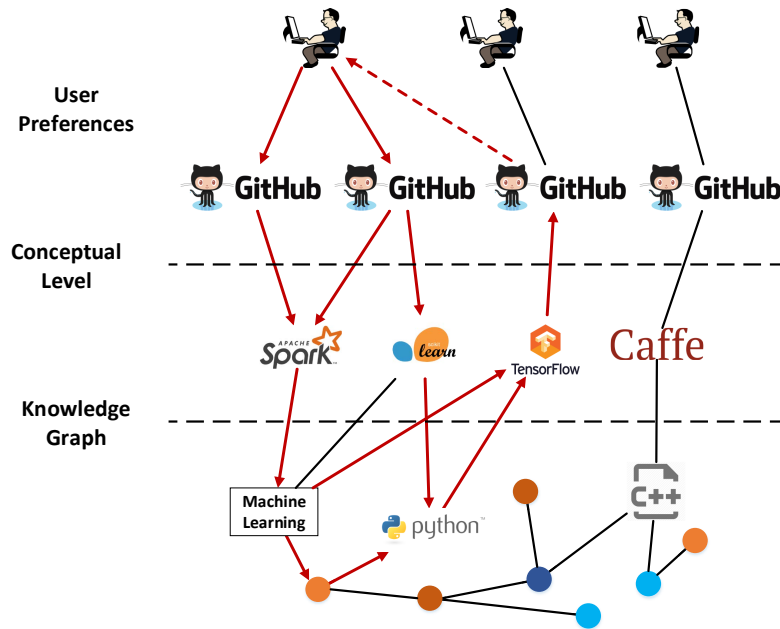


Figure 6.1: Conceptual Level of Hierarchical Collaborative Embedding

In Collaborative Embedding Learning, the representation vectors for Knowledge Graph and Recommender System can be stacked together, while the embedding representation measurement of components in the Knowledge Graph and Recommender System is not necessarily the same, user preference score is used for Recommender System, while link prediction score is used for Knowledge Graph. This imposes a limit on the recommendation performance.

To overcome that limitation, we propose the Collaborative Network Embedding (CNE) model to integrate the Knowledge Graph and Recommender System based on unified embedding representation measurement. **The CNE work introduced in this chapter**

is prepared to be submitted as a journal paper. The idea is illustrated in Fig. 6.2. To be specific, we extract the mutual entities from preferences data and KG data to form a

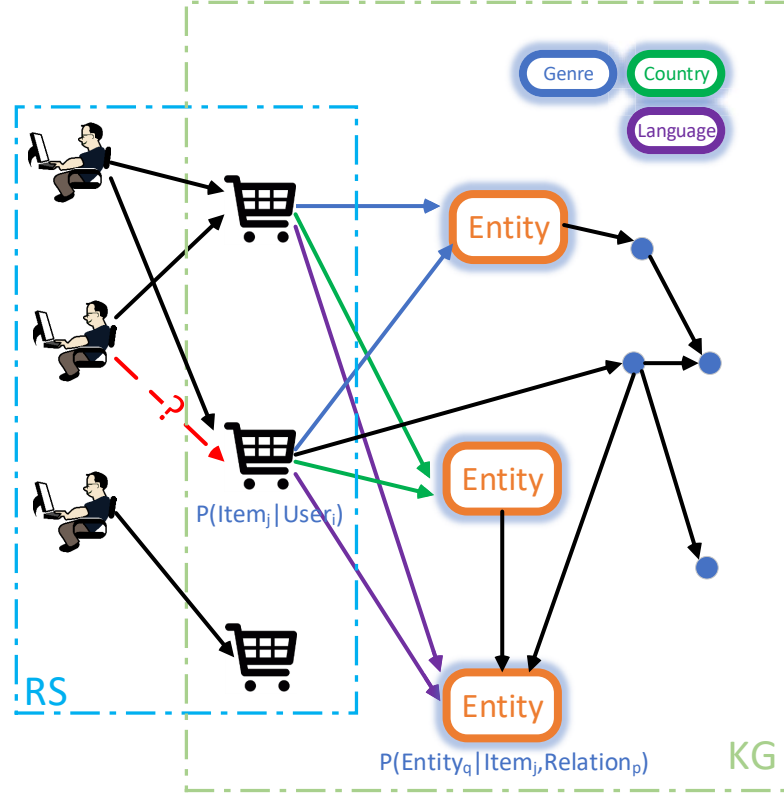


Figure 6.2: The CNE model consists of 2 parts: the CF part in the blue box and KG part in the green box. The two parts are connected via mutual elements.

joint heterogeneous network. The RS part of the joint network is a hierarchical bipartite network, and the KG part of the joint network is a multi-relational network. We can consider the recommendation task of RS as a **link prediction task on hierarchical bipartite network**, we consider the integrated system of Recommender System and Knowledge Graph as a multi-level network system, there are 3 levels, first level contains only users, second level contains only items, and the third level contains the entities related with the items. We can determine whether an item should be recommended to a user by predicting a link between the user node and the item node. Similarly, we can also consider inference of KG triple (head entity, relation, tail entity) as **link prediction task on multi-relational network**, whether the triple is true can be determined by predicting a specific type of relation link between two entity nodes. We intend to learn embedding representations for users, items, entities, and relations collaboratively, such

that both the user-item hierarchical information and the multi-relational structure information is modeled seamlessly. There are several previous methods for RS and KG achieve learning embedding for elements of RS and KG, such as BPR [68] for RS and TransE [12] for KG. Considering recommendation and KG inference as user preference prediction task, we need to compute user preference score for each new predicted link by the embedding of elements involved, BPR uses proximity degree of user and item as user preference score, the proximity degree is computed by product value of user embedding vector and item embedding vector, the higher the score is, the user more likely to select the item. In KG triple inference, TransE model adds head entity embedding and relation embedding as predicted tail entity embedding, then the distance between tail entity embedding and predicted tail entity embedding is used as link prediction score, the smaller the score is, the triple is more likely to be true. Some previous works, such as CKE [105], tried to stack these two types of model together directly, while it ignores the different meaning and scale of user preference score and link prediction score. The user preference score of BPR means proximity degree between user and item, the link prediction score of TransE means distance in a vector space, the scale of the link prediction scores of both BPR and TransE are not specifically constrained.

CNE model unifies the user preference and the link prediction score, inspired by proximity definition of LINE [80], we extend it into our proposed model, we redesign proximity probability based embedding model for both RS and KG. To compute proximity probability, embeddings are still needed to be learned for elements of RS and KG, then, for each new predicted link, we use embeddings of elements involved to compute a proximity probability which indicates how likely the link is true, for the recommendation, proximity probability indicates the probability that the item is proximate to the user's preference, for KG triple inference, proximity probability indicates the probability that tail entity is proximate to the missed answer of the KG triple containing only head entity and relation. Same measurement score, proximity probability, is used for both RS user preference prediction and KG link prediction, **same meaning, same scale**, and as a result, the recommendation accuracy is improved.

We summarize the main contributions of these two works as follows:

- A novel KG-based recommender system with knowledge conceptual level is proposed to properly encode the correlation amongst items which are non-existent knowledge graph entities. To the best of our knowledge, this is the first trial work of using knowledge graph embedding, to deal with a semantic enhancement for entities not existent in the conventional knowledge graph.

- We also proposed Collaborative Network Embedding to learn embeddings of components in Recommender System and Knowledge Graph. To overcome challenges that Recommender System user preference score and Knowledge Graph link prediction score have different meaning and scale, a proximity probability based unified score measurement is used to properly integrate Recommender System and Knowledge Graph into collaborative embedding learning process.
- We evaluate the performance of Hierarchical Collaborative Embedding model on *GitHub* recommendation task, which is extremely sparse but rich semantics in user preference. We tested the Collaborative Network Embedding model on two widely used recommendation task datasets, BookCrossing and Movielens, to show the effectiveness of the proposed Collaborative Network Embedding model.

6.2 Preliminaries

This section briefly summarizes the necessary background of *Implicit Feedback Recommendation* and *Knowledge Graph* that form the basis of this chapter.

6.2.1 Implicit Feedback Recommendation

The works introduced in this chapter try to solve the implicit feedback recommendation problem [104], i.e., analyzing interactions among users and items instead of explicit ratings. The implicit user feedback is encoded as a matrix $\mathbf{R} \in \mathbb{R}^{m \times n}$, where $R_{ij} = 1$ if user i has interacted with item j and $R_{ij} = 0$ otherwise. The user-item interactions are defined per application scenario, e.g., a *GitHub* user “stars” (follows) a repository, or a *Coursera* user “enrolled” in a subject. Generally speaking, an interaction $R_{ij} = 1$ implies the user is interested in the item, however, the meaning of $R_{ij} = 0$ is not necessarily to be not interested. In fact, the matrix $\mathbf{R}$ is often sparse and most entries will be 0, where the 0 value indicates that the user either has no interest in the item or has interest but not interacted with the item yet. The goal of implicit feedback recommendation is to identify which 0 entries in $\mathbf{R}$ have the potential to become 1.

6.2.2 Collaborative Filtering using Implicit Feedback

Collaborative filtering learns users’ interests from their feedback, either explicit or implicit. Popularized by the *Netflix prize*³, traditional methods focus on explicit feedback such as *ratings*. However, the last decade has seen a growing trend towards exploiting implicit feedback such as *clicks* and *purchases*. Implicit feedback has a major advantage of eliminating the needs of asking users explicitly. Instead, user feedback is collected silently, resulting in more user-friendly recommender systems. Hu et al. [34] and Pan et al. [61] investigated item recommendation from implicit feedback and propose to impute all missing values with zeros. More recently, Shi et al. [75] and Bayer et al. [5] extended Bayesian Personalized Ranking (BPR) [67] for optimizing parameters from implicit feedback. In this paper, we employ an optimization strategy similar to BPR, but with semantic information modeling. Standard BPR methods are also used as baselines for our experiments.

6.2.3 Knowledge Graph

The implicit feedback matrix $\mathbf{R}$ can be extremely sparse as some users may have only interacted with one or two items. Although modeling moderately sparse data has been considered by traditional CF methods, it remains a challenging problem of utilizing extreme sparse data. Fortunately, if the items contain rich semantic information, then only a few items will be able to connect the user to the knowledge graph, such that more complete user profiles can be built. To be specific, the knowledge graph is a semantic web consist of *entities* and *relations*, where entities represent anything in the world including people, things, events, etc., and relations connect entities that have interactions with each other. For example, in *GitHub* repository recommendation, entities can be software development concepts such as programming language *C++*, operating system *Linux*, development framework *TensorFlow*, etc. The entities are connected through relations such as “is programming language of”, “have dependency on”, “is operating system of”, etc. Denoting entities as nodes and relations as edges, the knowledge graph can be represented by a heterogeneous network with multiple types of nodes and multiple types of edges.

Although using knowledge graph in the recommendation is promising, it is assumed that the recommended items are entities in the knowledge graph. This assumption may hold for recommending movies or tourism destinations where the items are already

³<http://www.netflixprize.com/>

entities in the knowledge graph, but it becomes invalid for items that are non-existent in the knowledge graph, such as repositories in *GitHub*. Therefore, the link between non-existent items and knowledge graph entities must be identified together with reliability and importance measures.

For ease of reference, notations used throughout this paper are summarized as following. U_i represents vector of user i ; V_j represents vector of item j ; E_i represents vector of entity i ; B_j represents bias vector of item j ; W_k represents weigh of entity k ; I_j represents possibly related entities set of item j ; R represents factorized matrix of relation r ; r represents vector of relation r ; M_r represents subspace mapping matrix of relation r ; $p(i, j, j')$ represents preference function of triple (user i , item j , item j'); $\mathcal{X}_{i,j,j'}$ represents user preference term of training function; $\mathcal{Y}_{h,r,t,t'}$ represents knowledge graph embedding term of training function; $\mathcal{Z}$ represents regularization term of training function; f_r represents knowledge graph triple score function used in training function; f_r^{TransR} represents TransR score function; f_r^{RESCAL} represents RESCAL score function.

6.3 Hierarchical Collaborative Embedding

In this section, we propose the **H**ierarchical **C**ollaborative **E**MBEDDING model (**HCE**) to bridge Knowledge Graph to Recommender System, which jointly learns the embedding of elements, including users, items, entities, and relations.

6.3.1 Knowledge Graph Structured Embedding

With a large amount of knowledge being extracted from open source, knowledge graph was proposed to store the knowledge with graph structure. The knowledge facts are represented by triples, each triple has two entities (head entity and tail entity) and one relationship in between. Given all triples, the entities and relations can be considered as nodes and edges, respectively, resulting in a large scale of the heterogeneous knowledge graph. To capture the latent semantic information of entities and relations, several embedding based methods [12, 13, 44, 57, 77, 90] were proposed. These methods embed entities and relations into a continuous vector space, in which the latent semantic information can be reasoned automatically according to the vector space position of entities and relations.

Two state-of-the-art knowledge graph structured embedding methods are employed in this work: RESCAL [57] and TransR [44]. One important advantage of these two

methods is the capability of modeling multi-relational data where more than one relation may exist between two entities.

RESCAL uses three-way tensor to represent triples set, each element of a triple (head entity, relation, or tail entity) is represented by one dimension, and tensor factorization is used to obtain the entity and relation representations. To be specific, each entity is represented by a vector and each relationship is represented by a matrix. Y is the three-way tensor which represents all the triples, Y_k is a matrix picked up from Y , it only contains triples with relation k . E_i and E_j are the representation vectors of entity h and t , W_k is the representation matrix of relation r .

The representations of entities and relations are constructed by minimizing the following objective function:

$$(6.1) \quad \min_{E, W_k} \sum_k \|Y_k - EW_k E^T\|_F^2.$$

In a triple (h, r, t) , each entity is represented by a vector, E_h for head entity h , E_t for tail entity t , and relation r is represented by a matrix R . The RESCAL score function of a triple (h, r, t) is defined as:

$$(6.2) \quad f_r^{\text{RESCAL}}(h, t) = \|E_h R E_t\|_2^2,$$

TransR uses a different score function for triples. Given a triple (h, r, t) , the head and tail entities are represented by vectors E_h and E_t , respectively. Each relation is represented by a vector $\vec{r}$ together with a matrix M_r . TransR firstly maps entity h and t into subspace of relation r by using matrix M_r :

$$(6.3) \quad E_h^r = M_r E_h, E_t^r = M_r E_t.$$

The score function of TransR is defined as follows:

$$(6.4) \quad f_r^{\text{TransR}}(h, t) = \|E_h^r + \vec{r} - E_t^r\|_2^2.$$

In learning process, we pick up a true triple (h, r, t) and generate a false triple by replacing one entity of the triple by another entity: (h, r, t') . Then we make the score value of true triple larger than that of false triple: $f_r(h, t) > f_r(h, t')$.

6.3.2 Knowledge Conceptual Level Connection

This work focuses on recommender systems without direct connection to the knowledge graph, i.e., most recommendation items do not exist in the knowledge graph. For example,

a *GitHub* project with a customized name is not an entity in the knowledge graph. Consequently, methods such as [105] that rely on the direct mapping between items and knowledge graph entities are not applicable. However, by extracting content information from items, such as item description and user reviews, potential links between items and entities can be constructed. To bridge the recommender system and knowledge graph with item-entity links, we propose a collaborative learning model with the hierarchical structure of three levels: the recommender system level, the knowledge graph level, and the knowledge conceptual level (KCL). The KCL plays a key role in the model to connect the other two levels and enables collaborative learning.

Creating a knowledge conceptual level has two challenges. The first challenge is how to filter irrelevant linkages. The automated extraction of item content introduces lots of irrelevant information for the recommendation task. For example, in *GitHub* project recommendation, the project description may include the vocabulary of specific areas such as biology and chemistry, which are off-topic of general-purpose coding recommendation. While this information is irrelevant, it is actually linked to knowledge graph entities, thus introducing noise data.

The second challenge is how to measure the influences of knowledge graph entities on recommendation items. The entities have different influences on items, thus the links between items and entities must be weighed in order to represent an item precisely. To tackle these two challenges, the proposed Knowledge Conceptual Level implements the filtering and weighing functionalities. To be specific, a weighed link function is used to represent each item with their possibly related entities (automatically extracted from side information). The representation of an item is the weighted sum of vectors of possibly related entities plus a bias term B_j . Maximizing the weighed link function of each item is one of the targets in the collaborative learning process. The weighed link function of an item is defined as follows:

$$(6.5) \quad V_j = B_j + \sum_{k \in I_j} W_k E_k$$

where V_j is the representation of item j , E_k is the representation of entity k , W_k is the weigh parameter of entity k , and I_j is the set containing all the entities which are possibly related to the item j . If the entity is unrelated to the current recommendation task, the weighing parameter should be lowered to near zero during the learning process, if the influential degree of the entity is minor, the weighing parameter should be lowered accordingly. The filtering and weighing are both achieved by knowledge conceptual level.

6.3.3 Collaborative Learning

To integrate the recommender system with knowledge graph, the proposed collaborative learning framework learns the embedding representations of both recommender system elements (users and items) and knowledge graph elements (entities and relations).

Because of user feedback is implicit, similar to some previous works [67, 105], we use the pairwise ranking of items in our learning approach. Given user i , item j and item j' , using $F_{i,j}$ to represent the feedback of user i for item j , if $F_{i,j} = 1$ and $F_{i,j'} = 0$, then we consider user i prefer item j over item j' , we use preference function $p(i, j, j')$ to represent this pairwise preference relation, and $p(i, j, j') > 0$. More specifically, in our model, we use same-dimension vector representation for user and item, the preference function is defined as following,

$$(6.6) \quad p(i, j, j') = \ln \sigma(U_i^T V_j - U_i^T V_{j'})$$

U_i is the vector representing user i , V_j is the vector representing item j , $V_{j'}$ is the vector representing item j' , σ is sigmoid function.

Integrating knowledge graph embedding and knowledge conceptual level, the collaborative learning leverage the information from both user feedback and knowledge graph. by repeating the following procedure. Jointly, we aim to maximize the likelihood function in Eq. 6.7 and the overall learning algorithm is summarized in Alg. 7.

$$(6.7) \quad \begin{aligned} \mathcal{L} &= \sum_{(i,j,j') \in D} \mathcal{X}_{i,j,j'} + \sum_{(h,r,t,t') \in S} \mathcal{Y}_{h,r,t,t'} + \mathcal{Z} \\ \mathcal{X}_{i,j,j'} &= \ln \sigma(U_i^T V_j - U_i^T V_{j'}) \\ \mathcal{Y}_{h,r,t,t'} &= \ln \sigma(f_r(h, t) - f_r(h, t')) \\ \mathcal{Z} &= \frac{\lambda_U}{2} \|U\|_2^2 + \frac{\lambda_E}{2} \|E\|_2^2 + \frac{\lambda_R}{2} \|R\|_2^2 + \frac{\lambda_B}{2} \|B\|_2^2 + \frac{\lambda_W}{2} \|W\|_2^2 \\ V_j &= \sum_{k \in I_j} W_k E_k \end{aligned}$$

6.4 Collaborative Network Embedding with Different Measurement Scale

HCE model bridges the Knowledge Graph to Recommender System by using a collaborative embedding learning for users, items, entities, and relations. However there is still a problem left, Recommender System uses user preference score for preference prediction

Algorithm 7 *HCE* Algorithm

Input: User preferences, Knowledge graph, Item&entity links.

Training:

Step 1: Draw pairwise user-entity triple set $\mathcal{D}$.

Step 2: Repeat

for each $(u_i, v_j, v_{j'}) \in \mathcal{D}$ **do**

 Draw pairwise entity-relation quadruple set $\mathcal{S}_{j,j'}$

 Draw possibly related entities set I_j and $I_{j'}$

for each $(h, r, t, t') \in \mathcal{S}_{j,j'}$ **do**

Represent item by embedding of entities:

$$V_j = B_j + \sum_{k \in I_j} W_k * E_k$$

$$V_{j'} = B_{j'} + \sum_{k \in I_{j'}} W_k * E_k$$

Compute interaction of user and item:

$$\mathcal{X}_{i,j,j'} = \ln \sigma(U_i^T V_j - U_i^T V_{j'})$$

Compute score of entity-relation quadruple:

$$\mathcal{Y}_{h,r,t,t'} = \ln \sigma(f_r(h, t) - f_r(h, t'))$$

Compute regularization:

$$\mathcal{Z} = \|U_i\|, \|E_{\{h,t,t'\}}\|, \|R_r\|, \|B_{\{j,j'\}}\|, \|W_{\{j,j'\}}\|$$

maximize $\mathcal{X}_{i,j,j'} + \mathcal{Y}_{h,r,t,t'} + \mathcal{Z}$

Predictions:

for each $u_i \in U$ **do**

 Recommend items for user i in order

$$j_1 > j_2 > \dots > j_n \ (U_i^T V_{j_1} > U_i^T V_{j_2} > \dots > U_i^T V_{j_n}).$$

and Knowledge Graph uses link prediction score for missing link prediction, there is a gap between two scores, different meanings and different scales.

Inspired by several previous works including Large-scale Information Network Embedding model (LINE)[80] and Collaborative Knowledge base Embedding (CKE)[105], we propose our proposed model, Collaborative Network Embedding (CNE), to perform the Knowledge Graph based Recommendation task. The proposed model fully leverages collaborative filtering information of RS and graph transitivity information from KG. We connect two parts of our system, RS and KG into one collaborative network based mutual elements of RS items and KG entities, some items in RS can be directly projected into Knowledge Graph. We need to determine whether two types of edges between nodes are real, 1) whether a user node has interaction edge with an item node, 2) whether there is a relation edge from the head entity node to tail entity node. Based on BPR[68], we redesign a proximity-based embedding learning model for RS part of the system, and similarly, we redesign a proximity-based embedding model for KG part of the system based TransE [12]. We integrate two proximity based embedding models to build our CNE

model, inspired by LINE [80], the unified proximity defined is used for determination on both RS side and KG side.

6.4.1 Proximity Probability Embedding on RS Bipartite Network

Inspired by proximity definition of LINE, considering RS as a bipartite network consists of users and items, we define the proximity probability equation for bipartite network in Eq. 6.8.

$$(6.8) \quad p(v_j|u_i) = \frac{\exp(\vec{u}_i^T \vec{v}_j)}{\sum_{k \in |V|} \exp(\vec{u}_i^T \vec{v}_k)}$$

We get interaction score of user u_i with item v_j by computing interaction expectation of user u_i embedding $\vec{u}_i$ and item v_j embedding $\vec{v}_j$, then we replace item v_j with any item in item list and get interaction score of user u_i with any item v_k . We define the probability of recommending item v_j to user u_j by node proximity probability from user u_j node to item v_j node, $p(v_j|u_i)$.

6.4.2 Proximity Probability Embedding on KG Multi-relational Network

We also design a proximity probabilistic equation for KG multi-relational network, the equation is defined in Eq. 6.9 based on a popular KG embedding learning model, TransE [12].

$$(6.9) \quad \begin{aligned} f^{(t|h,r)}(h,r,t) &= (\vec{e}_h + \vec{e}_r)^T \vec{e}_t \\ p(t|h,r) &= \frac{\exp[f^{(t|h,r)}(h,r,t)]}{\sum_{t' \in E} \exp[f^{(t|h,r)}(h,r,t')]} \end{aligned}$$

As shown in Eq. 5.1, TransE model use vector space distance for reality determination of knowledge fact triple (head entity h , relation r , tail entity t). We also turn the original distance link prediction score equation of TransE into proximity probabilistic equation which can be integrated into final CNE model. We also add embedding of head entity $\vec{e}_h$ with embedding of relation $\vec{e}_r$, the assumption is that if the triple is true, sum vector of head entity embedding and relation embedding should be close to the tail

entity embedding. Different than TransE, we define the link prediction score of triple (h, r, t) , $f^{(t|h,r)}(h, r, t)$, by using interaction of sum vector and tail entity embedding. as $(\vec{e}_h + \vec{e}_r)^T \vec{e}_t$, the higher the score is, the triple is more likely to be true. First, we compute the expectation of triple (h, r, t) 's link prediction score. Then, we replace tail entity t with any entity t' in the entity list. After that, we can define the probability of how likely entity t occur on tail entity position of a triple with h as its head entity and r as its relation. Specifically, we define the proximity probability as $p(t|h, r)$, if h and r occur, the probability that t also occurs.

6.4.3 Collaborative Network Embedding

With two proximity probabilistic equations defined in Eq. 6.8 and Eq. 6.9, we give our final likelihood equation for training based on KL-divergence as shown in Eq. 6.10.

$$(6.10) \quad \begin{aligned} \mathcal{L} = & \sum_{i,j \in \mathcal{D}} \mathcal{W}_{(v_j|u_i)} \log p(v_j|u_i) \\ & + \alpha \sum_{h,r,t \in \mathcal{S}} \mathcal{W}_{(t|h,r)} \log p(t|h, r) \end{aligned}$$

In Eq. 6.10, $\mathcal{W}_{(v_j|u_i)}$ and $\mathcal{W}_{(t|h,r)}$ are empirical distribution of KL-divergence, $p(v_j|u_i)$ and $p(t|h, r)$ are conditional distribution of KL-divergence. $\mathcal{D}$ is the set of all user-item pairs, $\mathcal{S}$ is the set of all KG triples, α is the parameter to adjust importance of KG auxiliary information in model.

$$(6.11) \quad \begin{aligned} \mathcal{W}_{(v_j|u_i)} &= \frac{\text{frequency}(u_i, v_j)}{\sum_{v_k}^V \text{frequency}(u_i, v_k)} \text{influence}(u_i) \\ \mathcal{W}_{(t|h,r)} &= \frac{\text{frequency}(h, r, t)}{\sum_{t'}^E \text{frequency}(h, r, t')} \text{influence}(h, r) \end{aligned}$$

We have given definition of $p(v_j|u_i)$ and $p(t|h, r)$, and we define empirical distribution in Eq. 6.11. In our scenarios, we define empirical distribution based on two factors, 1) the observed statistical probability of user-item pair and KG triple, 2) influence of user and (head entity, relation) pair to our model. In left part of Eq. 6.11, we use frequency of pair (u_i, v_j) and frequency of all pairs containing u_i to compute observed statistical probability of user-item pair (u_i, v_j) , we use frequency of triple (h, r, t) and frequency of all triples containing h and r to compute observed statistical probability of KG triple (h, r, t) . In the right part, we define the influence of user u_i as the frequency of u_i in our

network, similarly, we define the influence of pair (h, r) as the frequency of (h, r) in our network. Eventually, the $\mathcal{W}_{(v_j|u_i)}$ and $\mathcal{W}_{(t|h,r)}$ used in our model equals to one, but in other scenarios, $\mathcal{W}_{(v_j|u_i)}$ and $\mathcal{W}_{(t|h,r)}$ may not always equals to 1 based on different definition of observed statistical probability and influence to our model.

6.4.4 Advantages of Proposed Model

Fully leveraging information from both RS and KG. The CNE model proposed in this work not only uses user-item interaction information from Recommender System, but also leverages semantic relation information between entities (including items of RS) from Knowledge Graph to supplement semantic enhancement to RS for a more accurate recommendation. We merge RS network and KG network into one heterogeneous collaborative network, then we learn the embeddings for RS and KG elements, users, items, entities, and relations, to predict new links between users and items, furthermore, recommend items for the user.

Adapting heterogeneous structure of the collaborative network. We propose CNE model to adopt different structures of RS and KG. In the collaborative network of RS and KG, RS part of the network consists of pair links, (user, item), while KG part of the network consists of triple links, (head entity, relation, tail entity), the relation edge between the head entity node and tail entity node has multiple types. Due to the difference between structures of RS and KG, CNE model applying different proximity equations for each part of the network, and finally constrains two part of proximity equations into a joint likelihood equation for collaborative embedding learning.

Unified proximity probability measurement for both sides of the collaborative network. Applying different proximity equations for each part of the collaborative network causes new challenges, unmatching between user preference score and link prediction score, different score meanings, different score scales. In this case, instead of stacking RS embedding model and KG embedding model directly, we redesign proximity probabilistic equations for both RS and KG parts of the network, and more importantly, the redesigned equations both use proximity probability unifies the measurement of RS user preference score and KG link prediction score.

Why do we use KG as auxiliary information of Recommendation task? KG is a type of large-scale semantic web, which can provide strong semantic enhancement for multiple types of application including RS. And KG is already proved to be helpful for recommendation accuracy improvement in previous works[105].

Why do we remain heterogeneous structure of collaborative network instead of turning it into a homogeneous network? Turning a collaborative network of RS and KG into a homogeneous network (considering multiple types of relationships as one type of relation) may simplify the complexity of both graph representation and embedding model computation, some previous network embedding model[80] can be applied directly. However, important semantic information contained in KG is wasted, multi-relational information between KG entities can largely help RS to improve recommendation result through considering semantical human common knowledge information. We remain heterogeneous structure of the collaborative network, and we design a more proper collaborative network embedding model to overcome the challenges caused by heterogeneity.

Why do we use embedding based model for collaborative network link prediction? We use embedding based model because of high generalization capability of representation learning. By learning the embedding for each element of our collaborative network, given a pair of user and item, we can determine whether we should recommend this item to this user based on their embedding proximity, we don't need to do determination by searching all the explicit rules concluded from training data.

Why do we need to unify new link prediction scores with proximity probability? Some previous works, such as [105], directly stacks the popular embedding models for RS and KG together and learns a collaborative embedding to show KG can help to improve recommendation accuracy. However, ignoring different link prediction scores of RS and KG limits its effectiveness. We use unified link prediction scores, proximity probability, for both RS and KG in CNE, which makes the integration of RS and KG part in likelihood equation more reasonable, the scores of both sides are probabilities. The final recommendation result is also improved as shown in the experiments section.

6.5 Experiment and Analysis

To see the effectiveness of our proposed models, in this section, we introduce the datasets, the baselines and the results of comparative experiments.

6.5.1 Dataset

Dataset for Hierarchical Collaborative Embedding model. To demonstrate the effectiveness of HCE method, we collected *GitHub* dataset and conduct experiments on

it. The *GitHub* dataset is chosen for several reasons. Firstly, the user feedback is implicit which is more realistic in real-world recommendation. Besides, the *GitHub* dataset is quantitatively sparse but semantically dense, the dataset consists of 3,798 users, 2,477 items and 22,096 interactions. Defining the density ratio as $iteration_num/(user_num * item_num)$, the ratio of *GitHub* dataset is 0.0026. In contrast, the popular MovieLens-1M the dataset has a density ratio of 0.0119 even if only 5-star ratings are considered. Though the *GitHub* dataset is quantitatively sparse, it is semantically dense, the repositories are highly related to each other based on their semantic information including some simple entity-based interactions, such as some repositories use programming language “C++” or some repositories use toolkit “TensorFlow”. We also leverage some complex interactions from the knowledge graph, for example, a repository uses toolkit “TensorFlow” which is implemented by “C++” which is the programming language of another repository. We do recommendation on *GitHub* dataset not only based on historical cooccurrence of items but also based on semantic information enhanced by a knowledge graph. The other reason we use *GitHub* dataset is that the items (repositories) can’t be directly mapped to entities of knowledge graph because of its highly customized item name. Although directly mapping is used in some previous works, it fails in recommendation tasks where item names are customized.

Table 6.1: Dataset Size of Movielens and Book-crossing

	Movielens	Book-crossing
User	5,999	46,342
Item	3,133	98,221
Implicit Rating	226,202	226,469
Item Entity	1,069	4,195
Entity Involved	64,528	559,748
Relation Involved	291	409
Triple Involved	72,848	614,685

Dataset for Collaborative Network Embedding model. And to test the performance of the proposed CNE model, we use two real-world datasets, Movielens⁴ and Book-Crossing⁵. Film and book recommendations are two types of popular recommendation task, besides that, these two types of recommendation can also be improved by adding semantic auxiliary enhancement such as KG. Films and books have rich semantic

⁴<https://grouplens.org/datasets/movielens/>

⁵<http://www2.informatik.uni-freiburg.de/cziegler/BX/>

auxiliary information, and the user preference for films and books largely depends on this kind of semantic auxiliary information.

Movielens is a large-scale web-based recommendation dataset, the Movielens dataset we use is a subset of whole Movielens data. As shown in Tab. 6.1, it contains 5,999 users, 3,133 film items and 226,202 interactions between users and items. 1,069 film items can be recognized as entities in DBpedia⁶ KG we use. By adding KG auxiliary information in, we can have 64,528 entities into a collaborative network of RS and KG, we can have 291 types of relations, and 72,848 factual triples from KG are involved from KG to improve recommendation accuracy.

Book-Crossing is another popular recommendation dataset which contains the preference of users on books. Similarly to Movielens dataset, we also count statistics of Book-Crossing in Tab. 6.1, 46,342 users, 98,221 book items and 226,469 user-item interactions are used. 4,195 book items can be recognized as entities in KG. In the collaborative network of RS and KG, 558,748 entities, 409 types of relationships and 614,685 factual triples are involved.

The original data of Movielens and Book-Crossing datasets are collected for explicit recommendation task. The rating scores of Movielens are from 1-5, the rating scores of Book-Crossing are from 1-10. We turn these two datasets into implicit recommendation task datasets by extracting the rating score equals to 5 from Movielens and extracting the rating score large than 8 from Book-Crossing. The extract ratings are labeled as 1 implicit selection from user to an item, the rest interaction between users and items are unknown (the user don't like the item or the user hasn't seen the item).

In most of the application scenarios, implicit recommendation task is more common, the explicit user rating is hard to be collected, however implicit rating can be collected through several automatic ways, historical user buying records, historical user clicking records, etc.

6.5.2 Baselines

We choose following methods as baselines to compare with HCE model, **BPRMF** (Bayesian Personalized Ranking based Matrix Factorization), **BPRMF+TransE** and **FM** (Factorization Machines).

The baseline models used for evaluating CNE performance include Factorization Machine (FM), Large-scale Information Network Embedding (LINE), Bayesian Personalized

⁶<https://wiki.dbpedia.org/>

Ranking (BPR), Collaborative Knowledge base Embedding (CKE) and a modified version of the CNE model excluding information of KG.

BPRMF ignores the knowledge graph information, it only focuses on historical user feedback, the results are learned by using pairwise item ranking based matrix factorization.

BPRMF+TransE uses almost the same setting as our proposed models (RESCAL-based HCE and TransR-based HCE), while it only considers part of knowledge graph information. By using TransE knowledge graph embedding method, it ignores the multi-relational data.

FM [65, 66] is another popular solution for integrating side information into recommendation tasks. While it is limited by only considering the entities as items' features and ignoring the semantic structural relation between entities.

LINE [80] model considers the collaborative network of RS and KG as a homogeneous network, it ignores the three factors in KG based RS scenarios, 1) hierarchical structure between users and items in RS, 2) multi-relational link between KG entities, 3) heterogeneous structure of RS and KG.

CKE [105] model is implemented and tested, essentially, CKE model stack BPR model and TransE model into one joint embedding model and learn the parameters collaboratively. However, it still ignores the different link prediction measurement of two parts of the collaborative network.

NE is a modified version of CNE model in baseline list, in this modified version, we exclude the information of KG and only do recommendation based on hierarchical network information of RS. We call this modified version as NE (Network Embedding) in our experiments.

6.5.3 Comparison

Results Comparison for Hierarchical Collaborative Embedding model. To measure both the precision and recall of recommendation results, we use MAP@k (mean average precision) [25] and Recall@k [24] in our experiments. Due to utilizing two knowledge graph embedding methods, RESCAL and TransR, in proposed Hierarchical Collaborative Embedding (HCE) model, we compare RESCAL-based HCE and TransR-based HCE with baselines (BPRMF, FM, BPRMF+TransE) respectively.

CHAPTER 6. HIERARCHICAL COLLABORATIVE EMBEDDING FOR KNOWLEDGE GRAPH AND RECOMMENDER SYSTEM

Table 6.2: Mean MAP and Recall results of Hierarchical Collaborative Embedding on Github Dataset

Algorithm	MAP@k					Recall@k				
	MAP@10	MAP@20	MAP@30	MAP@40	MAP@50	Recall@10	Recall@20	Recall@30	Recall@40	Recall@50
BPRMF	.0172 ± .0002	.0143 ± .0001	.0127 ± .0002	.0115 ± .0002	.0105 ± .0002	.0873 ± .0016	.1416 ± .0015	.1863 ± .0055	.2225 ± .0058	.2533 ± .0060
FM	.0184 ± .0001	.0154 ± .0002	.0133 ± .0001	.0118 ± .0001	.0108 ± .0001	.0888 ± .0007	.1510 ± .0018	.1968 ± .0009	.2382 ± .0022	.2697 ± .0037
BPRMF+TransE	.0191 ± .0003	.0162 ± .0000	.0140 ± .0001	.0124 ± .0001	.0113 ± .0000	.0966 ± .0028	.1615 ± .0048	.2055 ± .0038	.2408 ± .0041	.2715 ± .0032
HCE_RESCAL	.0197 ± .0004	.0161 ± .0004	.0141 ± .0001	.0127 ± .0001	.0116 ± .0001	.0998 ± .0039	.1577 ± .0060	.2057 ± .0041	.2451 ± .0029	.2766 ± .0037
HCE_TransR	.0199 ± .0001	.0163 ± .0002	.0142 ± .0002	.0128 ± .0001	.0116 ± .0002	.1025 ± .0018	.1649 ± .0014	.2125 ± .0038	.2511 ± .0049	.2804 ± .0039

Each experiment is repeated five times with different random seeds and we report the MAP and Recall values by varying the position k in Tab. 6.2. The results can be summarized as follows: 1) Results of the FM model are better than BPRMF, because BPRMF totally ignores knowledge graph information, knowledge graph information is useful to improve the recommendation results. 2) The improvement of FM is limited, less than BPRMF+TransE model, because the FM model doesn't consider the relation structure of knowledge graph, integrating knowledge graph structured embedding in our proposed model by using knowledge conceptual level effectively elevates MAP@k and Recall@k scores. 3) Although the BPRMF+TransE model is effective, it is still outperformed by both RESCAL-based HCE model and TransE-based HCE model, because the latter two models consider the multi-relational data of knowledge graph. The effectiveness of the proposed HCE framework is presented. Knowledge Conceptual Level serves as the core component of HCE framework appropriately.

Results Comparison for Collaborative Network Embedding model. And we also try to find out the effectiveness difference between CNE and baselines. We use precision@K and Recall@K to measure the effectiveness of models for recommendation task. As shown in Tab. 6.3 and Tab. 6.4, we test CNE and baselines on Movielens and Book-Crossing dataset. For FM and LINE, ignoring KG graph transitivity, the heterogeneous structure of the collaborative network, hierarchical structure of RS and multi-relational links of KG lead to unsatisfactory results.

Table 6.3: Precision and Recall results of Collaborative Network Embedding on Movielens Dataset

	Precision					Recall				
	@10	@20	@30	@40	@50	@10	@20	@30	@40	@50
FM	0.01205	0.01217	0.01238	0.01243	0.01250	0.01480	0.02988	0.04561	0.06102	0.07673
LINE	0.00408	0.00355	0.00333	0.00314	0.00316	0.00501	0.00870	0.01227	0.01544	0.01938
BPR	0.07169	0.05961	0.05243	0.04733	0.04362	0.08802	0.14637	0.19312	0.23244	0.26777
CNE(BPR+TransE)	0.08001	0.06523	0.05676	0.05101	0.04655	0.09823	0.16017	0.20907	0.25054	0.28578
NE(without KG)	0.10464	0.08337	0.07050	0.06236	0.05620	0.12847	0.20471	0.25966	0.30624	0.34503
CNE	0.10864	0.08504	0.07197	0.06332	0.05741	0.13339	0.20881	0.26509	0.31096	0.35240

Table 6.4: Precision and Recall results of Collaborative Network Embedding on BookCrossing Dataset

	Precision					Recall				
	@10	@20	@30	@40	@50	@10	@20	@30	@40	@50
FM	0.000010	0.000005	0.000003	0.000010	0.000012	0.000038	0.000038	0.000038	0.000151	0.000226
LINE	0.000124	0.000171	0.000179	0.000194	0.000192	0.000452	0.001243	0.001958	0.002825	0.003502
BPR	0.003051	0.002632	0.002375	0.002177	0.002006	0.011110	0.019169	0.025948	0.031710	0.036531
CKE(BPR+TransE)	0.002978	0.002580	0.002265	0.002172	0.002081	0.010846	0.018793	0.024743	0.031635	0.037886
NE(without KG)	0.003847	0.003258	0.002916	0.002663	0.002472	0.014010	0.023726	0.031861	0.038790	0.045004
CNE	0.004002	0.003371	0.003027	0.002751	0.002519	0.014575	0.024555	0.033066	0.040071	0.045871

BPR and CKE models achieve quite an accurate recommendation precision and recall, with the help of KG, CKE performs better on Movielens dataset. However, we also find that the result of CKE is not as good as the result of BPR on Book-Crossing, which also proves the importance of measurement unification. With ununified link prediction measurements, the meaning of embedding learning on RS and KG are unmatched, which causes that the KG auxiliary information may confuse the decision of RS in some specific scenarios, such as recommendation on Book-Crossing dataset.

CNE model fully leverages the information of RS and KG into the collaborative network, remaining all useful information including graph transitivity, RS hierarchical structure, the multi-relational link of KG and heterogeneous network structure of the collaborative network. It also unifies the link prediction measurement of both RS and KG as proximity probability. As a result of the above consideration, CNE outperforms the baselines on both Movielens and Book-Crossing datasets.

And the result of CNE is better than NE on both two datasets, we can see that KG is a useful auxiliary information for recommendation task.

6.6 Summary

In this chapter, we first proposed Hierarchical Collaborative Embedding (HCE) framework, which integrates the recommender system with Knowledge Graph into a three-level model. The information of knowledge graph is leveraged to improve the results of quantitatively sparse but semantically dense recommendation scenarios. The experiment of HCE was conducted on real-world *GitHub* dataset showing that semantic information from knowledge graph has been properly captured, resulting in improved recommendation performance. To the best of our knowledge, this is the first attempt of using knowledge graph embedding to perform semantic enhancement for items that do not exist in the knowledge graph, by using the proposed Knowledge Conceptual Level.

Then, to properly integrate CF based Recommendation model and Knowledge Graph Embedding model with different measurement scale into a heterogeneous network with unified possibility measurement, we propose the Collaborative Network Embedding (CNE) model. The multi-relational heterogeneous structure information of the joint network consists of RS and KG are fully leveraged for recommendation result improvement. The embedding learned for both RS and KG part of the collaborative network are used to compute unified new link prediction measurement, proximity probability. We test the CNE model on two widely used recommendation datasets, Movielens and Book-Crossing, and the experiment results show the the effectiveness of CNE model.

CONCLUSION AND FUTURE WORK

With the rapid growth of Knowledge Graphs and Knowledge Graph based application systems, researches focusing on Knowledge Graph analysis is playing a more and more important role because of several reasons, 1) most of currently existing Knowledge Graphs require completion, 2) abundant knowledge inference information contained in the multi-relational network structure of Knowledge Graphs, and 3) Knowledge Graph can improve the result of external application scenarios as a rich semantic enhancement.

Some foundations have been established by previous works in Knowledge Graph analysis area. However, Knowledge Graph analysis still require more effective solutions for the following problems, 1) modeling multi-relational network structure Knowledge Graph, 2) integrating conceptual information into knowledge reasoning, 3) relation-based knowledge reasoning through Knowledge Graph sub-structure, and 4) properly collaborative learning for Knowledge Graph and Recommender System. By studying the network embedding learning in Knowledge Graph, this thesis proposed several embedding based approaches for Knowledge Graph completion, reasoning, and application. In this chapter, the research results and the main contributions of this thesis are summarized, some possible future research directions are also presented.

7.1 Contributions

To solve the problems in the Knowledge Graph analysis area more effectively, some solutions are proposed in this thesis, the conclusions and main contributions of the works

in this thesis are listed.

- **A bipartite graph embedding based Knowledge Graph completion method** is proposed, each knowledge fact triple contained in Knowledge Graph is represented in the form of the bipartite graph structure. As a result of using bipartite graph representation, the proposed model can achieve more reasonable and more accurate link prediction to find missing links in original Knowledge Graph. The experiment is designed to compare the proposed model with several traditional Knowledge Graph embedding methods, the result shows the effectiveness of bipartite graph embedding based Knowledge Graph completion method. **Representing knowledge fact triple by bipartite graph embedding based model, the multi-relational network structure Knowledge Graph can be properly modeled.**
- **An embedding based cross completion method for factual Knowledge Graph and conceptual taxonomy** is proposed. Conceptual taxonomy is leveraged as additive information for Knowledge Graph completion task, linking Knowledge Graph entities with matching instance entities in conceptual taxonomy, a joint embedding based method is proposed to collaboratively learn the embedding representation of components in both factual Knowledge Graph and conceptual taxonomy, entities, relations, and entity types. Based on joint embedding, the additive semantic information transfer from conceptual taxonomy into Knowledge Graph embedding, the accuracy of joint embedding outperforms the other baselines without using conceptual taxonomy. **With the joint embedding solution, the cross completion can be achieved on factual Knowledge Graph and conceptual taxonomy, the conceptual information is integrated into Knowledge Graph completion properly.**
- **Two sub-structure based Knowledge Graph transitive relation embedding methods** are proposed to reveal the structure of human knowledge reasoning through representing the reasoning process with sub-structure containing relation transitivity. To build modelable methods, two relation embedding based models are proposed to represent knowledge reasoning based on relation embedding only, no entity embedding is needed to be learned in these two models. Triangle pattern based transitive relation embedding is firstly proposed to analyze the triangle structure relation transitivity, and the learned embedding is used for link prediction on Knowledge Graph. **The triangle pattern transitive relation embedding is**

a type of modelable sub-structure based method, which can be used for knowledge reasoning.

The second proposed method, Meta-structured based transitive relation embedding, shares a similar basic idea, but meta-structure consisting of multiple triangles are used to analyze relation transitivity in the more complex structure. The experiment result shows that integrating transitive relation embedding with traditional Knowledge Graph embedding can achieve more accurate link prediction. **Meta-structure based transitive relation embedding composes complex meta-structure with multiple triangle patterns, it enables the model to achieve embedding based complex knowledge reasoning.**

- **Two hierarchical collaborative embedding methods for Knowledge Graph and recommender system** are proposed, the target is to improve the accuracy of recommender system by using the Knowledge Graph as a semantic enhancement. One solution to achieve the target is collaborative embedding for components of Knowledge Graph and recommender system, entities, relations, items, and users. The first proposed method, Hierarchical Collaborative Embedding (HCE), links Knowledge Graph and recommender system with a flexible conceptual level, then a framework is designed for collaborative embedding learning based on the conceptual level. **Hierarchical collaborative embedding model gives a latent semantic information collaborative learning solution for the system integrated with Knowledge Graph and recommender system through a flexible conceptual level.**

The second proposed method, Collaborative Network Embedding (CNE), focuses on unifying the originally different measurements of Knowledge Graph and recommender system, link prediction score is used in Knowledge Graph and user preference score is used in a recommender system, with different meaning and different scale, directly stacking embedding of two systems together ignores the different measurement issue. CNE model attempts to unify the measurement by using proximity probability as score measurement of embedding based models. The experiment result proves that the accuracy of the recommender system can be improved by using the Knowledge Graph as semantic enhancement properly. **Collaborative network embedding model uses proximity probability to unify the measurements of Knowledge Graph and recommender system for collaborative embedding learning.**

7.2 Possible Future Work

Although the solutions proposed in this thesis addressed the aforementioned research problems, there still some problems needed to be researched in the future.

- Complex transitive relation embedding based knowledge graph reasoning base on sub-structures composition. Although some sub-structure based transitive relation embedding methods are proposed in this thesis, the structure of relation transitivity used in reasoning analysis is relatively simple, such as triangle patterns and meta-structure. In the large scale Knowledge Graph, the reasoning can be related to complex relation transitivity structure, one obvious solution is to compose a complex structure by multiple simple sub-structures, designing new transitive relation embedding methods based on sub-structures composition is one future research direction.
- Applying sub-structure based transitive relation embedding into collaborative learning of Knowledge Graph and recommender system. Currently, the sub-structure based transitive relation embedding is only used for Knowledge Graph self-completion, while sub-structure analysis can also be applied on a heterogeneous network consisting of Knowledge Graph and recommender system. Modeling sub-structure in a heterogeneous network consisting of Knowledge Graph and recommender system with proper transitive relation embedding methods is also a new direction for future research in this area.

BIBLIOGRAPHY

- [1] D. H. ACKLEY, G. E. HINTON, AND T. J. SEJNOWSKI, *A learning algorithm for boltzmann machines*, Cognitive science, 9 (1985), pp. 147–169.
- [2] G. ADOMAVICIUS AND A. TUZHILIN, *Context-aware recommender systems*, in Recommender systems handbook, Springer, 2011, pp. 217–253.
- [3] S. ARORA, Y. LIANG, AND T. MA, *A simple but tough-to-beat baseline for sentence embeddings*, (2016).
- [4] S. AUER, C. BIZER, G. KOBILAROV, J. LEHMANN, R. CYGANIAK, AND Z. IVES, *Dbpedia: A nucleus for a web of open data*, in The semantic web, Springer, 2007, pp. 722–735.
- [5] I. BAYER, X. HE, B. KANAGAL, AND S. RENDLE, *A generic coordinate descent framework for learning from implicit feedback*, in Proceedings of the 26th International Conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2017, pp. 1341–1350.
- [6] Y. BENGIO, A. COURVILLE, AND P. VINCENT, *Representation learning: A review and new perspectives*, IEEE Transactions on Pattern Analysis & Machine Intelligence, 35 (2013), pp. 1798–1828.
- [7] C. BIZER, T. HEATH, AND T. BERNERS-LEE, *Linked data: The story so far*, in Semantic services, interoperability and web applications: emerging concepts, IGI Global, 2011, pp. 205–227.
- [8] K. BOLLACKER, C. EVANS, P. PARITOSH, T. STURGE, AND J. TAYLOR, *Freebase: a collaboratively created graph database for structuring human knowledge*, in Proceedings of the 2008 ACM SIGMOD international conference on Management of data, AcM, 2008, pp. 1247–1250.

- [9] A. BORDES, S. CHOPRA, AND J. WESTON, *Question answering with subgraph embeddings*, (2014).
- [10] A. BORDES, X. GLOROT, J. WESTON, AND Y. BENGIO, *Joint learning of words and meaning representations for open-text semantic parsing.*, in AISTATS, vol. 351, 2012, pp. 423–424.
- [11] —, *A semantic matching energy function for learning with multi-relational data*, Machine Learning, 94 (2014), pp. 233–259.
- [12] A. BORDES, N. USUNIER, A. GARCIA-DURAN, J. WESTON, AND O. YAKHNENKO, *Translating embeddings for modeling multi-relational data*, in Advances in Neural Information Processing Systems, 2013, pp. 2787–2795.
- [13] A. BORDES, J. WESTON, R. COLLOBERT, AND Y. BENGIO, *Learning structured embeddings of knowledge bases*, in Conference on Artificial Intelligence, no. EPFL-CONF-192344, 2011.
- [14] A. BORDES, J. WESTON, AND N. USUNIER, *Open question answering with weakly supervised embedding models*, in Joint European conference on machine learning and knowledge discovery in databases, Springer, 2014, pp. 165–180.
- [15] R. BURKE, F. VAHEDIAN, AND B. MOBASHER, *Hybrid recommendation in heterogeneous networks*, in International Conference on User Modeling, Adaptation, and Personalization, Springer, 2014, pp. 49–60.
- [16] S. CAO, W. LU, AND Q. XU, *Deep neural networks for learning graph representations*, in Thirtieth AAAI Conference on Artificial Intelligence, 2016.
- [17] A. CARLSON, J. BETTERIDGE, R. C. WANG, E. R. HRUSCHKA JR, AND T. M. MITCHELL, *Coupled semi-supervised learning for information extraction*, in Proceedings of the third ACM international conference on Web search and data mining, ACM, 2010, pp. 101–110.
- [18] D. CARTWRIGHT AND F. HARARY, *Structural balance: a generalization of heider’s theory.*, Psychological review, 63 (1956), p. 277.
- [19] K.-W. CHANG, W.-T. YIH, B. YANG, AND C. MEEK, *Typed tensor decomposition of knowledge bases for relation extraction*, in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1568–1579.

-
- [20] S. CHANG, W. HAN, J. TANG, G.-J. QI, C. C. AGGARWAL, AND T. S. HUANG, *Heterogeneous network embedding via deep architectures*, in Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 119–128.
- [21] T. CHEN AND Y. SUN, *Task-guided and path-augmented heterogeneous network embedding for author identification*, in Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, ACM, 2017, pp. 295–304.
- [22] F. CORCOGLIONITI, M. DRAGONI, M. ROSPOCHER, AND A. P. APROSIO, *Knowledge extraction for information retrieval*, in European Semantic Web Conference, Springer, 2016, pp. 317–333.
- [23] P. CUI, X. WANG, J. PEI, AND W. ZHU, *A survey on network embedding*, IEEE Transactions on Knowledge and Data Engineering, 31 (2018), pp. 833–852.
- [24] J. DAVIS AND M. GOADRIC, *The relationship between precision-recall and roc curves*, in International Conference on Machine Learning, 2006, pp. 233–240.
- [25] S. DIELEMAN AND B. SCHRAUWEN, *Deep content-based music recommendation*, in International Conference on Neural Information Processing Systems, 2013, pp. 2643–2651.
- [26] X. DONG, E. GABRILOVICH, G. HEITZ, W. HORN, N. LAO, K. MURPHY, T. STROHMANN, S. SUN, AND W. ZHANG, *Knowledge vault: A web-scale approach to probabilistic knowledge fusion*, in Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2014, pp. 601–610.
- [27] Y. DONG, N. V. CHAWLA, AND A. SWAMI, *metapath2vec: Scalable representation learning for heterogeneous networks*, in Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2017, pp. 135–144.
- [28] C. FELLBAUM, *Wordnet and wordnets*, (2005).
- [29] A. GROVER AND J. LESKOVEC, *node2vec: Scalable feature learning for networks*, in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2016, pp. 855–864.

- [30] G. E. HINTON AND R. R. SALAKHUTDINOV, *Reducing the dimensionality of data with neural networks*, Science, 313 (2006), pp. 504–507.
- [31] G. E. HINTON AND T. J. SEJNOWSKI, *Learning and relearning in boltzmann machines*, Parallel distributed processing: Explorations in the microstructure of cognition, 1 (1986), pp. 282–317.
- [32] B. HIXON, P. CLARK, AND H. HAJISHIRZI, *Learning knowledge graphs for question answering through conversational dialog*, in Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2015, pp. 851–861.
- [33] R. HU, C. C. AGGARWAL, S. MA, AND J. HUAI, *An embedding approach to anomaly detection*, in 2016 IEEE 32nd International Conference on Data Engineering (ICDE), IEEE, 2016, pp. 385–396.
- [34] Y. HU, Y. KOREN, AND C. VOLINSKY, *Collaborative filtering for implicit feedback datasets*, in Data Mining, 2008. ICDM’08. Eighth IEEE International Conference on, IEEE, 2008, pp. 263–272.
- [35] H. HUANG, J. TANG, S. WU, L. LIU, ET AL., *Mining triadic closure patterns in social networks*, in Proceedings of the 23rd international conference on World wide web, ACM, 2014, pp. 499–504.
- [36] G. JI, S. HE, L. XU, K. LIU, AND J. ZHAO, *Knowledge graph embedding via dynamic mapping matrix*, in Meeting of the Association for Computational Linguistics and the International Joint Conference on Natural Language Processing, 2015, pp. 687–696.
- [37] R. KINDERMANN AND J. L. SNELL, *Markov Random Fields and Their Applications*, American Mathematical Society, 1980.
- [38] H. LEE, A. BATTLE, R. RAINA, AND A. Y. NG, *Efficient sparse coding algorithms*, in Advances in neural information processing systems, 2006, pp. 801–808.
- [39] J. LEHMANN, *Dbpedia: A large-scale, multilingual knowledge base extracted from wikipedia*, Semantic Web, 6 (2015), pp. 167–195.
- [40] D. B. LENAT, *Cyc: A large-scale investment in knowledge infrastructure*, Communications of the ACM, 38 (1995), pp. 33–38.

-
- [41] C. LI, J. MA, X. GUO, AND Q. MEI, *Deepcas: An end-to-end predictor of information cascades*, in Proceedings of the 26th international conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2017, pp. 577–586.
 - [42] J. LIANG, H. WANG, H. WANG, Y. ZHANG, AND W. WANG, *Probase+: Inferring missing links in conceptual taxonomies*, IEEE Transactions on Knowledge & Data Engineering, 29 (2017), pp. 1281–1295.
 - [43] Y. LIN, Z. LIU, H. LUAN, M. SUN, S. RAO, AND S. LIU, *Modeling relation paths for representation learning of knowledge bases*, in Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 2015, pp. 705–714.
 - [44] Y. LIN, Z. LIU, X. ZHU, X. ZHU, AND X. ZHU, *Learning entity and relation embeddings for knowledge graph completion*, in Twenty-Ninth AAAI Conference on Artificial Intelligence, 2015, pp. 2181–2187.
 - [45] H. LIU, Y. WU, AND Y. YANG, *Analogical inference for multi-relational embeddings*, in International Conference on Machine Learning, 2017, pp. 2168–2178.
 - [46] D. LUKOVNIKOV, A. FISCHER, J. LEHMANN, AND S. AUER, *Neural network-based question answering over knowledge graphs on word and character level*, in Proceedings of the 26th international conference on World Wide Web, International World Wide Web Conferences Steering Committee, 2017, pp. 1211–1220.
 - [47] F. MAHDISOLTANI, J. BIEGA, AND F. M. SUCHANEK, *Yago3: A knowledge base from multilingual wikipedias*, 2013.
 - [48] T. MAN, H. SHEN, S. LIU, X. JIN, AND X. CHENG, *Predict anchor links across social networks via an embedding approach.*, in IJCAI, vol. 16, 2016, pp. 1823–1829.
 - [49] P. N. MENDES, M. JAKOB, AND C. BIZER, *Dbpedia spotlight: shedding light on the web of documents*, in International Conference on Semantic Systems, 2011.
 - [50] T. MIKOLOV, K. CHEN, G. CORRADO, AND J. DEAN, *Efficient estimation of word representations in vector space*, arXiv preprint arXiv:1301.3781, (2013).

- [51] T. MIKOLOV, I. SUTSKEVER, K. CHEN, G. S. CORRADO, AND J. DEAN, *Distributed representations of words and phrases and their compositionality*, in Advances in neural information processing systems, 2013, pp. 3111–3119.
- [52] E. MILLER, *An introduction to the resource description framework*, Bulletin of the American Society for Information Science and Technology, 25 (1998), pp. 15–19.
- [53] G. A. MILLER, *Wordnet: a lexical database for english*, Communications of the ACM, 38 (1995), pp. 39–41.
- [54] B. MIN, R. GRISHMAN, L. WAN, C. WANG, AND D. GONDEK, *Distant supervision for relation extraction with an incomplete knowledge base*, in Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2013, pp. 777–782.
- [55] N. NATARAJAN AND I. S. DHILLON, *Inductive matrix completion for predicting gene–disease associations*, Bioinformatics, 30 (2014), pp. i60–i68.
- [56] A. Y. NG, *Feature selection, l_1 vs. l_2 regularization, and rotational invariance*, in Proceedings of the twenty-first international conference on Machine learning, ACM, 2004, p. 78.
- [57] M. NICKEL, *Tensor factorization for relational learning*, PhD thesis, lmu, 2013.
- [58] M. NICKEL, L. ROSASCO, AND T. POGGIO, *Holographic embeddings of knowledge graphs*, in Thirtieth Aai conference on artificial intelligence, 2016.
- [59] M. NICKEL, V. TRESP, AND H. P. KRIEGEL, *A three-way model for collective learning on multi-relational data*, in International Conference on International Conference on Machine Learning, 2011, pp. 809–816.
- [60] M. OU, P. CUI, J. PEI, Z. ZHANG, AND W. ZHU, *Asymmetric transitivity preserving graph embedding*, in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2016, pp. 1105–1114.
- [61] R. PAN, Y. ZHOU, B. CAO, N. N. LIU, R. LUKOSE, M. SCHOLZ, AND Q. YANG, *One-class collaborative filtering*, in Data Mining, 2008. ICDM’08. Eighth IEEE International Conference on, IEEE, 2008, pp. 502–511.

-
- [62] H. PAULHEIM, *Knowledge graph refinement: A survey of approaches and evaluation methods*, Semantic web, 8 (2017), pp. 489–508.
- [63] J. PEARL, *Probabilistic reasoning in intelligent systems: networks of plausible inference*, Morgan Kaufmann, 2014.
- [64] B. PEROZZI, R. AL-RFOU, AND S. SKIENA, *Deepwalk: Online learning of social representations*, in Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2014, pp. 701–710.
- [65] S. RENDLE, *Factorization machines*, in Data Mining (ICDM), 2010 IEEE 10th International Conference on, IEEE, 2010, pp. 995–1000.
- [66] —, *Factorization machines with libfm*, ACM Transactions on Intelligent Systems and Technology (TIST), 3 (2012), p. 57.
- [67] S. RENDLE, C. FREUDENTHALER, Z. GANTNER, AND L. SCHMIDT-THIEME, *Bpr: Bayesian personalized ranking from implicit feedback*, in Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence, AUAI Press, 2009, pp. 452–461.
- [68] —, *Bpr: Bayesian personalized ranking from implicit feedback*, (2012), pp. 452–461.
- [69] L. F. RIBEIRO, P. H. SAVERESE, AND D. R. FIGUEIREDO, *struc2vec: Learning node representations from structural identity*, in Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2017, pp. 385–394.
- [70] S. RIEDEL, L. YAO, A. MCCALLUM, AND B. M. MARLIN, *Relation extraction with matrix factorization and universal schemas*, in Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2013, pp. 74–84.
- [71] S. J. RUSSELL AND P. NORVIG, *Artificial intelligence: a modern approach*, Malaysia; Pearson Education Limited,, 2016.
- [72] R. SALAKHUTDINOV, A. MNIH, AND G. HINTON, *Restricted boltzmann machines for collaborative filtering*, in Proceedings of the 24th international conference on Machine learning, ACM, 2007, pp. 791–798.

- [73] M. SCHMACHTENBERG, C. BIZER, AND H. PAULHEIM, *Adoption of the linked data best practices in different topical domains*, in International Semantic Web Conference, Springer, 2014, pp. 245–260.
- [74] C. SHI, Z. ZHANG, P. LUO, P. S. YU, Y. YUE, AND B. WU, *Semantic path based personalized recommendation on weighted heterogeneous information networks*, in Proceedings of the 24th ACM International on Conference on Information and Knowledge Management, ACM, 2015, pp. 453–462.
- [75] Y. SHI, A. KARATZOGLOU, L. BALTRUNAS, M. LARSON, N. OLIVER, AND A. HANJALIC, *Climf: learning to maximize reciprocal rank with collaborative less-is-more filtering*, in Proceedings of the sixth ACM conference on Recommender systems, ACM, 2012, pp. 139–146.
- [76] A. SINGHAL, *Introducing the knowledge graph: things, not strings*, Official google blog, 5 (2012).
- [77] R. SOCHER, D. CHEN, C. D. MANNING, AND A. NG, *Reasoning with neural tensor networks for knowledge base completion*, in Advances in neural information processing systems, 2013, pp. 926–934.
- [78] F. M. SUCHANEK, G. KASNECI, AND G. WEIKUM, *Yago: A large ontology from wikipedia and wordnet*, Web Semantics: Science, Services and Agents on the World Wide Web, 6 (2008), pp. 203–217.
- [79] J. TANG, M. QU, AND Q. MEI, *Pte: Predictive text embedding through large-scale heterogeneous text networks*, in Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 1165–1174.
- [80] J. TANG, M. QU, M. WANG, M. ZHANG, J. YAN, AND Q. MEI, *Line:large-scale information network embedding*, 2 (2015), pp. 1067–1077.
- [81] C. E. TSOURAKAKIS, *Fast counting of triangles in large real networks without counting: Algorithms and laws*, in Eighth IEEE International Conference on Data Mining, 2008, pp. 608–617.
- [82] C. TU, W. ZHANG, Z. LIU, M. SUN, ET AL., *Max-margin deepwalk: Discriminative learning of network representation.*, in IJCAI, vol. 2016, 2016, pp. 3889–3895.

-
- [83] D. VRANDEČIĆ AND M. KRÖTZSCH, *Wikidata: a free collaborative knowledge base*, (2014).
- [84] D. WANG, P. CUI, AND W. ZHU, *Structural deep network embedding*, in Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2016, pp. 1225–1234.
- [85] H. WANG, H. WANG, J. R. WEN, AND Y. XIAO, *An inference approach to basic level of categorization*, in ACM International on Conference on Information and Knowledge Management, 2015, pp. 653–662.
- [86] H. WANG, F. ZHANG, J. WANG, M. ZHAO, W. LI, X. XIE, AND M. GUO, *Ripplenet: Propagating user preferences on the knowledge graph for recommender systems*, in Proceedings of the 27th ACM International Conference on Information and Knowledge Management, ACM, 2018, pp. 417–426.
- [87] S. WANG, J. TANG, C. AGGARWAL, Y. CHANG, AND H. LIU, *Signed network embedding in social media*, in Proceedings of the 2017 SIAM international conference on data mining, SIAM, 2017, pp. 327–335.
- [88] X. WANG, P. CUI, J. WANG, J. PEI, W. ZHU, AND S. YANG, *Community preserving network embedding*, in Thirty-First AAAI Conference on Artificial Intelligence, 2017.
- [89] X. WANG, D. WANG, C. XU, X. HE, Y. CAO, AND T.-S. CHUA, *Explainable reasoning over knowledge graphs for recommendation*, in 33rd AAAI Conference on Artificial Intelligence (AAAI 2019), 2019.
- [90] Z. WANG, J. ZHANG, J. FENG, AND Z. CHEN, *Knowledge graph embedding by translating on hyperplanes*, in Twenty-Eighth AAAI Conference on Artificial Intelligence, 2014, pp. 1112–1119.
- [91] J. WELBL, S. RIEDEL, AND G. BOUCHARD, *Complex embeddings for simple link prediction*, in International Conference on Machine Learning, 2016, pp. 2071–2080.
- [92] J. WESTON, A. BORDES, O. YAKHNENKO, AND N. USUNIER, *Connecting language and knowledge bases with embedding models for relation extraction*, (2013).
- [93] W. WU, H. LI, H. WANG, AND K. Q. ZHU, *Probase: a probabilistic taxonomy for text understanding*, 2012, pp. 481–492.

- [94] C. XIONG, R. POWER, AND J. CALLAN, *Explicit semantic ranking for academic search via knowledge graph embedding*, in Proceedings of the 26th international conference on world wide web, International World Wide Web Conferences Steering Committee, 2017, pp. 1271–1279.
- [95] B. YANG, W. YIH, X. HE, J. GAO, AND L. DENG, *Embedding entities and relations for learning and inference in knowledge bases*, (2014).
- [96] C. YANG, Z. LIU, D. ZHAO, M. SUN, AND E. CHANG, *Network representation learning with rich text information*, in Twenty-Fourth International Joint Conference on Artificial Intelligence, 2015.
- [97] X. YANG, Y.-N. CHEN, D. HAKKANI-TÜR, P. CROOK, X. LI, J. GAO, AND L. DENG, *End-to-end joint learning of natural language understanding and dialogue manager*, in 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), IEEE, 2017, pp. 5690–5694.
- [98] X. YAO AND B. VAN DURME, *Information extraction over structured data: Question answering with freebase*, in Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, 2014, pp. 956–966.
- [99] S. YEUNG, O. RUSSAKOVSKY, G. MORI, AND L. FEI-FEI, *End-to-end learning of action detection from frame glimpses in videos*, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 2678–2687.
- [100] S. W.-T. YIH, M.-W. CHANG, X. HE, AND J. GAO, *Semantic parsing via staged query graph generation: Question answering with knowledge base*, (2015).
- [101] R. YING, R. HE, K. CHEN, P. EKSOMBATCHAI, W. L. HAMILTON, AND J. LESKOVEC, *Graph convolutional neural networks for web-scale recommender systems*, in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ACM, 2018, pp. 974–983.
- [102] H.-G. YOON, H.-J. SONG, S.-B. PARK, AND S.-Y. PARK, *A translation-based knowledge graph embedding preserving logical property of relations*, in Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2016, pp. 907–916.

- [103] X. YU, X. REN, Y. SUN, Q. GU, B. STURT, U. KHANDELWAL, B. NORICK, AND J. HAN, *Personalized entity recommendation: A heterogeneous information network approach*, in Proceedings of the 7th ACM international conference on Web search and data mining, ACM, 2014, pp. 283–292.
- [104] X. YU, X. REN, Y. SUN, B. STURT, U. KHANDELWAL, Q. GU, B. NORICK, AND J. HAN, *Recommendation in heterogeneous information networks with implicit user feedback*, in ACM Conference on Recommender Systems, 2013, pp. 347–350.
- [105] F. ZHANG, N. J. YUAN, D. LIAN, X. XIE, AND W.-Y. MA, *Collaborative knowledge base embedding for recommender systems*, in Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, ACM, 2016, pp. 353–362.

