Information Network Modeling and Mining

by

Jingyuan Zhang
B.E., Software Engineering, Dalian University of Technology, 2009
M.S., Computer Science, Dalian University of Technology, 2012

Thesis submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Chicago, 2017

Chicago, Illinois

Defense Committee:
Philip S. Yu, Chair and Advisor
Bing Liu
Chris Kanich
Yuheng Hu, Department of Information and Decision Sciences
Roger Jie Luo, Snapchat Inc.

This thesis is dedicated to my parents and my husband

for their love, endless support,

and encouragement.

ACKNOWLEDGMENTS

First and foremost, I would like to express my deepest gratitude to my advisor Prof. Philip S. Yu. I cannot thank him enough for his continuous support, mentoring and patience during my PhD study at UIC. I'm truly indebted to him for providng a research environment in which I could freely explore different interesting topics. Without his great help, support, and encouragement, I would certainly not be where I am today. My thanks also go to Dr. Yi Chang at Huawei Research America. It is my honor to work with him and he always gave me generous help, inspiration and mentoring on my research.

I would like to thank Prof. Bing Liu, Prof. Chris Kanich, Prof. Yuheng Hu and Dr. Roger Jie Luo for taking their valuable time to serve on my dissertation committee and for their constructive suggestions and feedbacks. I also want to thank Yahoo! Research for providing me opportunities to apply my research into practice.

I wish to extend my warmest thanks to all my colleagues and friends that I met in UIC. I am very grateful to work with them during my PhD study and there are countless fun and enjoyable moments while we discuss and learn.

Finally, I would like to thank my family for their unconditional support and love for all these years. Especially, I want to thank my beloved husband, Dr. Wenfei Zhang. His love and trust make our long-distance relationship sustain all the difficulties since 2009.

JZ

CONTRIBUTION OF AUTHORS

Chapter 1 is an introduction that outlines my dissertation research. Chapter 2 presents a published manuscript (1) for which I was the primary author. Dr. Xiangnan Kong, Dr. Luo Jie, Dr. Yi Chang, and my advisor Professor Philip S. Yu contributed to discussions with respect to the work and revising the manuscript.

Chapter 3 presents a published manuscript (2) for which I was the primary author. I started the research of this problem when I was an intern at Yahoo! Research under the supervisor of Dr. Yi Chang. Dr. Luo Jie, Dr. Altaf Rahman, Dr. Sihong Xie, Dr. Yi Chang and Professor Philip S. Yu contributed to discussions with respect to the work and revising the manuscript.

Chapter 4 presents a published manuscript (3) for which I was the primary author. This work was done when I interned at Yahoo! Research for the second time. Chun-Ta Lu, Dr. Mianwei Zhou, Dr. Sihong Xie, Dr. Yi Chang and Professor Philip S. Yu contributed to discussions with respect to the work and revising the manuscript.

Chapter 5 presents a manuscript under review (4) for which I was the primary author. Chun-Ta Lu, Bokai Cao, Dr. Yi Chang and Professor Philip S. Yu contributed to discussions with respect to the work and revising the manuscript.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

TABLE OF CONTENTS (Continued)

CHAPTER **PAGE**

LIST OF TABLES

LIST OF FIGURES

SUMMARY

We are living in an Internet world with huge amounts of information. For example, online Question-and-Answer (Q&A) websites include different kinds of informational objects, i.e., questions, answers and users. Web search logs contain queries and clicked webpages. These online infrastructures usually represent information in various forms. An urgent challenge is to discover meaningful knowledge from massive information that can empower online platforms from different perspectives.

Motivated by this trend, my research addresses the modeling and mining tasks for several online platforms, including the Q&A websites, the search engines and the social media. The major goal of my research is to propose information network based modeling and mining techniques regarding to heterogeneous data sources. The frameworks, by extracting interconnected objects, modeling them into information networks, and designing network based learning algorithms, help improve the service offered by different online platforms.

The methodology of the information network based modeling and mining is proved to be effective on several topics of knowledge discovery, including the co-ranking problem in large-scale Q&A sites (1), the learning of entity types from massive search query logs (2), and the detection of emerging relationships from news and knowledge graphs (3; 4). The future work is to explore the network base modeling and mining techniques on more online platforms and study how they can fit for new situations.

CHAPTER 1

INTRODUCTION

1.1 Thesis Outline

Online information is usually represented in various forms. Examples include the different kinds of informational objects, questions, answers and users in Q&A websites, the queries and clicked webpages in search logs. In order to learn useful knowledge from massive information, we extract these interconnected objects and model them into *information networks*, aiming to capture and integrate information as much as possible. The real world can be abstracted and represented in an information network. It focuses on the objects (nodes) and the interactions (links) between the objects. Such network modeling represents and stores essential information about the real world with great power. In addition, it also provides a useful manner to mine meaningful knowledge from the interacted objects.

In this thesis, we focus on the information network modeling and mining techniques on several real-world problems. Through modeling different online platforms into information networks, we solve real-life issues with insightful analysis and innovative methodologies for different online platforms, including the Q&A websites, the search engines and the social media.

Our work in this thesis covers four different research directions in the study of information network modeling and mining:

- **First**, we model the Q&A sites into a heterogeneous network, in which question, answers and users are connected to each other through different types of links. Based on the network modeling, we study the unsupervised co-ranking problem in Q&A sites.
- **Second**, we model search query logs into a heterogeneous network of entities (e.g., person, movie or place), contexts and clicked URLs. Based on the network modeling, we learn entity types to help improve user search experience.
- **Third**, we focus on news texts and model them into a heterogeneous network of entities and contexts. Based on the network modeling, we detect emerging relationships to help complete the current knowledge graphs.
- **Finally**, we still focus on the news texts and the knowledge graphs to detect emerging relationships. By formulating an elegant fourth-order tensor, we connect emerging relation types from news to knowledge graphs.

1.2 Network-Based Co-Ranking in Q&A Sites

(Part of the section was previously published in (1).)

Question-and-answer (Q&A) websites, such as Yahoo! Answers, Stack Overflow and Quora, have become a popular and powerful platform for Web users to share knowledge on a wide range of subjects. This has led to a rapidly growing volume of information and the consequent challenge of readily identifying high quality objects (questions, answers and users) in Q&A sites. Exploring the interdependent relationships among different types of objects can help find high quality objects in Q&A sites more accurately.

In Chapter 2, we specifically focus on the ranking problem of co-ranking questions, answers and users in a Q&A website. By studying the tightly connected relationships between Q&A objects, we can gain useful insights toward solving the co-ranking problem. However, co-ranking multiple objects in Q&A sites is a challenging task: a) With the large volumes of data in Q&A sites, it is important to design a model that can scale well; b) The large-scale Q&A data makes extracting supervised information very expensive. In order to address these issues, we propose an unsupervised **N**etwork-based **C**o-**R**anking framework (NCR) to rank multiple objects in Q&A sites. Empirical studies on real-world Yahoo! Answers datasets demonstrate the effectiveness and the efficiency of the proposed NCR method.

1.3 Learning Entity Types from Query Logs

(Part of the section was previously published in (2).)

Entities (e.g., person, movie or place) play an important role in real-world applications and learning entity types has attracted much attention in recent years. Most conventional automatic techniques use large corpora, such as news articles, to learn types of entities. However, such text corpora focus on general knowledge about entities in an objective way. Hence, it is difficult to satisfy those users with specific and personalized needs for an entity. Recent years have witnessed an explosive expansion in the mining of search query logs, which contain billions of entities. The word patterns and click-throughs in search logs are not found in text corpora, thus providing a complemental source for discovering entity types based on user behaviors.

In Chapter 3, we study the problem of learning entity types from search query logs and address the following challenges: a) queries are short texts, and information related to entities

is usually very sparse; b) large amounts of irrelevant information exists in search logs, bringing noise in detecting entity types. We first model query logs using a bipartite graph with entities and their auxiliary information, such as contextual words and clicked URLs. Then we propose a graph-based framework called ELP (**E**nsemble framework based on **L**able **P**ropagation) to simultaneously learn the types of both entities and auxiliary signals. In ELP, two separate strategies are designed to fix the problems of sparsity and noise in query logs. Extensive empirical studies are conducted on real search logs to evaluate the effectiveness of the proposed ELP framework.

1.4 Detecting Emerging Relations from News

(Part of the section was previously published in (3).)

Real-world knowledge is growing rapidly nowadays. New entities arise with time, resulting in large volumes of relations that do not exist in current knowledge graphs (KGs). These relations containing at least one new entity are called *emerging relations*. They often appear in news, and hence the latest information about new entities and relations can be learned from news timely.

In Chapter 4, we focus on the problem of discovering emerging relations from news. However, there are several challenges for this task: a) at the beginning, there is little information for emerging relations, causing problems for traditional sentence-based models; b) no negative relations exist in KGs, creating difficulties in utilizing only positive cases for emerging relation detection from news; and c) new relations emerge rapidly, making it necessary to keep KGs up to date with the latest emerging relations. In order to address these issues, we start from

a global graph perspective and propose a novel **H**eterogeneous graph **E**mbedding framework for **E**merging **R**elation detection (HEER) that learns a classifier from positive and unlabeled instances by utilizing information from both news and KGs. Extensive experiments on real-world news datasets demonstrate the effectiveness of the proposed HEER model.

1.5 Connecting Emerging Relation Types from News to Knowledge Graphs

Knowledge graphs (KGs) have been widely used to represent relationships among entities, while KGs cannot capture new relationships between entities emerging along time. Since news often provides the latest information regarding the new entities and relationships, there is an opportunity to connect emerging relationships from news timely. However, it is a challenging task due to the source heterogeneity of structured KGs and unstructured news texts.

In order to address the issue, we propose a tensor-based framework to capture the complex interactions among multiple relations, entities and text descriptions in Chapter 5. We further develop an efficient **T**ext-**A**ware **MU**lti-**RE**lational learning method (TAMURE) that can learn the embedding representations of entities and relations from both KGs and news, by jointly factorizing the interaction parameters. Moreover, the complexity of TAMURE is linear to the number of interaction parameters, making it scalable to large KGs and news texts. Extensive experiments via TensorFlow demonstrate the effectiveness of the proposed TAMURE model compared with eight state-of-the-art methods on real-world datasets.

CHAPTER 2

NETWORK-BASED CO-RANKING IN Q&A SITES

(This chapter was previously published as "NCR: A Scalable Network-Based Approach to Co-Ranking in Question-and-Answer Sites", in *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management (CIKM'14)* (1). DOI: `https://doi.org/10.1145/2661829.2661978`.)

2.1 Introduction

Over the past few years, question-and-answer (Q&A) sites, such as Yahoo! Answers, Stack Overflow and Quora, have provided a new way for Web users to share knowledge on a wide range of subjects. Individuals can conveniently address specific needs to the public and get first-hand replies. These Q&A sites have exploded in popularity. Yahoo! Answers, the first and largest Q&A site, had 7,000 new questions and 21,000 new answers posted per hour in July, 2012 (5); Stack Overflow, a Q&A site for computer programmers, had over 1.9 million registered users and more than 5.5 million questions in August, 2013. These repositories of valuable knowledge provide a gold mine for information retrieval and automatic question answering.

Since the Q&A site allows anyone to contribute knowledge on the Web, the quality of its objects (questions, answers and users) varies dramatically. On Yahoo! Answers, the answers provided by experts are detailed and useful, while others, provided by non-experts, may even contain spam and junk information. This is true for other types of objects as well. Some

questions become popular in a short period of time and they get thousands of answers. In contrast, some questions are ridiculous and fail to get any answers. Distinguishing high quality objects from low quality ones can help improve the service offered by Q&A websites. Therefore, the problem of object ranking in Q&A sites has received considerable attention in the last few years (6; 7; 8).

Most conventional approaches of object ranking in Q&A sites focus on a single type of object. For instance, the works in (9; 6) investigated how to evaluate or predict the quality of questions, and (8; 10) aim to find high quality answers. In reality, the multiple types of objects are interrelated, e.g., good questions often attract high quality answers from competent users. Exploring the interrelationships among different types of objects can help identify high quality objects in Q&A sites more effectively.

In this chapter, we specifically focus on the ranking problem of co-ranking questions, answers and users in a Q&A website. Studying the co-ranking problem has many benefits to real-world applications. With high quality objects, Q&A websites can recommend the most trendy information to users. In addition, the top objects under a certain topic or over all the topics can help improve user engagements because individuals can have a quick tour over popular information in the community. Furthermore, the co-ranking performance can be considered as an important signal for answer vertical search and web search. Integrated with such query independent signal, we can better rank the relevant questions, answers and users for a certain query. By studying the tightly connected relationships among different types of objects, we can

gain useful insights toward solving the co-ranking problem. However, this is a challenging task due to the following reasons:

- Q&A sites are getting larger with millions and billions of objects. For example, Yahoo! Answers had already hit 1 billion answers in May 2010 (11) and it had received 300 million questions in July, 2012 (5). So it is important to design a model that not only can rank each type of object well, but also can accurately scale to large websites.
- Another challenge lies in the fact that little prior knowledge about Q&A sites exists. Previous approaches (12; 13) focus on learning to rank different objects in Q&A sites under supervised or semi-supervised settings, which explicitly or implicitly assume the availability of some prior knowledge. However, with large volumes of new questions, answers and users, extracting supervised information from Q&A sites can be very expensive and time consuming.

In order to address these issues, we model a Q&A site as a heterogeneous network, and capture the interdependent relationships to infer the *popularity of questions*, *interestingness of answers* and *contributions of users* simultaneously. Take the ranking of questions for example, Yahoo! Answers simply ranks the questions according to the number of answers they have. However, we observe in Section 2.3 that the popularity of a question does not depend on how many answers it attracts but on the interestingness of those answers, which is a common heuristic similar to the PageRank (14) principle. For instance, a question ("**Is 1+1=2?**") attracting 100 boring answers ("**Yes**" or "**No**") is much less popular than a question ("**Why do cats**

act like monkeys?") attracting 10 interesting answers. We consider such interdependencies in our co-ranking model.

In summary, our contributions can be summarized as follows:

- We explore the interdependent relationships among questions, answers and users, and verify the existence of such relationships in Q&A sites. Based on these interrelationships, we rank each object under an unsupervised setting. To the best of our knowledge, this is the first work that investigates the usefulness of such relationships in Q&A sites.
- We propose an unsupervised **N**etwork-based **C**o-**R**ank-ing model (NCR) to simultaneously recognize high quality objects in Q&A sites. We design the NCR framework in a divide-and-conquer way to decompose the co-ranking problem into three separate types of sub-modules for questions, answers and users. The interdependencies are captured through iterative computations. To the best of our knowledge, this is the first unsupervised method that solves this problem in Q&A sites.
- We show how the proposed NCR model can be executed in a parallelized environment to scale up to very large Q&A sites. The divide-and-conquer framework makes the co-ranking problem parallelizable. At each iteration, ranking results for users, questions and answers are computed in parallel. Then these results are used in the next iteration to capture interdependencies among objects. The distributed algorithm helps Q&A sites rapidly identify high quality objects even when the data arrives in large volumes.
- We conduct extensive empirical studies on real Yahoo! Answers datasets to demonstrate the effectiveness of the proposed NCR method and the efficiency of the distributed version.

2.2 Problem Definition

In this section, we first introduce several related concepts and notations. Then, we will formally define the problem of co-ranking multiple types of objects in Q&A networks.

Definition 1 ***Heterogeneous Q&A Network****: A heterogeneous Q&A network is a special kind of information network. It is represented as a graph* $G = (V, E)$*.* V *is the set of nodes (objects). It has* t *types of objects* $T_1 = \{v_{11}, ..., v_{1n_1}\}, ..., T_t = \{v_{t1}, ..., v_{tn_t}\}$*.* $E \subseteq V \times V$ *is the set of links (relations) between the nodes in* V*. It involves multiple types of links. The network of Yahoo! Answers is shown in Figure 1, where each shape represents one type of object in the network, and each arrowed line represents one type of link. It involves three types of objects, i.e., users (U), questions (Q) and answers (A), and three types of links, i.e., askedBy, answeredBy and givenBy.*

In heterogeneous Q&A networks, each type of link means an unique binary relation R from node type i to node type j, where $R(v_{ip}, v_{jq})$ holds iff object v_{ip} and v_{jq} are related by relation R. R^{-1} is the inverted relation of R. It holds naturally for $R^{-1}(v_{jq}, v_{ip})$. $dom(R) = \mathcal{T}_i$ denotes the domain of relation R, $range(R) = \mathcal{T}_j$ denotes its range. For example, in Figure 1, the link type "*askedBy*" can be written as a relation R between question nodes and user nodes. $R(v_{ip}, v_{jq})$ holds iff question v_{ip} is asked by user v_{jq}. Given a Q&A network, we can define different adjacent matrices according to the different types of links as follows:

Definition 2 $\mathbf{L_{QA}}$ ***matrix****:* $E_{QA} \subseteq Q \times A \subset E$ *is the set of answeredBy links between questions and answers. The corresponding adjacent matrix can be denoted as* L_{QA}*, where* $L_{QA}(q, a) = 1$

if a question $q \in Q$ has an answer $a \in A$, and $L_{QA}(q, a) = 0$ otherwise. Q and A have the relationship of $1 : n$, i.e., a question may have n answers ($n \geq 1$) but an answer is only for a certain question.

Definition 3 $\mathbf{L_{AU}}$ *matrix*: *$E_{AU} \subseteq A \times U \subset E$ is the set of givenBy links between answers and users. The corresponding adjacent matrix can be represented as L_{AU}, where $L_{AU}(a, u) = 1$ if an answer $a \in A$ is posted by a user $u \in U$, and $L_{AU}(a, u) = 0$ otherwise. A and U have the relationship of $n : 1$, i.e., an answer is only from a certain user but a user may provide n answers ($n \geq 1$) to different questions.*

Definition 4 $\mathbf{L_{QU}}$ *matrix*: *$E_{QU} \subseteq Q \times U \subset E$ is the set of askedBy links between questions and users. The corresponding adjacent matrix is L_{QU}, where $L_{QU}(q, u) = 1$ if a question $q \in Q$ is asked by a user $u \in U$, and $L_{QU}(q, u) = 0$ otherwise. Q and U have the relationship of $n : 1$, i.e., a question is only asked by a certain user but a user may post n questions ($n \geq 1$) on Q&A websites.*

Given a Q&A network, our goal is to rank different types of nodes simultaneously. So we define variables that quantify the qualities of questions, answers and users more precisely:

Definition 5 *Popularity of questions*: *The popularity of a question $q \in Q \subset V$ (denoted by $\mathcal{P}(q)$) is a score of how popular the question q is. It indicates the question q's ability to attract hot debates or discussions. For ease of understanding and computations, we limit the range of $\mathcal{P}(q)$ to (0, 1).*

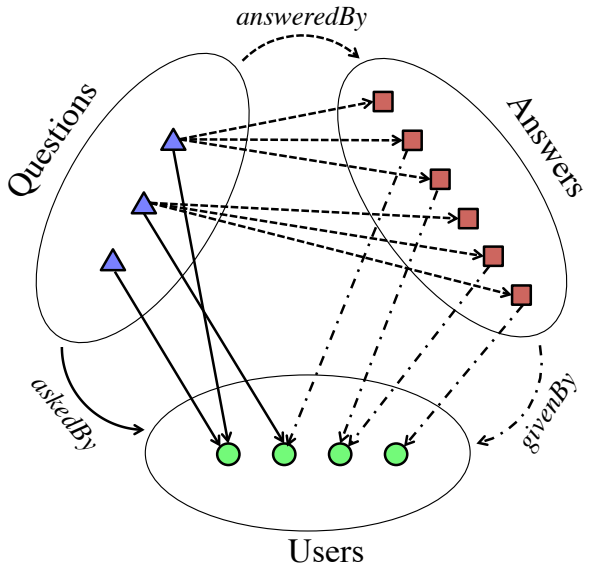


Figure 1. The heterogeneous network in Yahoo! Answers.

For example, "Why do cats act like monkeys?" is a popular question because everyone can be involved to show diverse answers from different viewpoints.

Definition 6 ***Interestingness of answers****: The interestingness of an answer $a \in A \subset V$ (denoted by $\mathcal{I}(a)$) is a score of how interesting the answer a is. The interestingness represents the answer a's strength to impress users. $\mathcal{I}(a)$ is within the range (0, 1).*

We still take the question "Why do cats act like monkeys?" as an example, an answer "They are scared little cats and hiding from the big dogs." is more interesting than another answer "Wait.....what?".

Definition 7 ***Contribution of users****: The contribution of a user $u \in U \subset V$ (denoted by $\mathcal{C}(u)$) is a score of how the user u contributes to the Q&A community. It indicates the user u's*

ability to ask popular questions and give interesting answers. $\mathcal{C}(u)$ *is also within the range (0, 1).*

For instance, a user asking 10 popular questions contributes more than one who asks 100 questions with nobody answering them. Similarly, a user providing many interesting answers also contributes a lot to the Q&A system.

Based on the node and relation types in Q&A networks, the input of the co-ranking task consists of a heterogeneous network $G = (V, E)$. V includes three types of nodes Q, A and U. E involves three types of links E_{QA}, E_{AU} and E_{QU}. Our goal is to build a general framework to simultaneously recognize high quality questions, answers and users under an unsupervised setting. In order to solve this problem, we need to address the following challenges:

1. How can one capture the interdependent relationships among a question's popularity, an answer's interestingness and a user's contribution?

2. Based upon the interdependent relationships, how can one simultaneously rank the questions, answers and users in an unsupervised way?

3. As the network is getting larger, how can one scale up the ranking algorithm to the growth of Q&A sites?

For the first challenge, we explore some observations on the real Yahoo! Answers data. Based on them, we present the interdependent relationships among the three different types of nodes in Section 2.3. For the second challenge, we introduce how to iteratively rank the three types of nodes based on the interrelationships in Section 2.4. For the third challenge, we

implement the framework in a parallelized environment and show the time efficiency in Section 2.5.

2.3 Data Analysis

A motivation for this work is that a question's popularity, an answer's interestingness, and a user's contribution could be strongly correlated with each other in Q&A websites. Before proceeding, we first introduce real-world data used in this work and investigate whether the observations support the assumption of interdependent relationships among questions, answers and users.

2.3.1 Data Description

Yahoo! Answers is a popular Q&A website where people ask and answer questions on any topic. Each question has a lifecycle. After it is posted by an asker, it stays in an "open" state with arrivals of answers. Then the question becomes "closed" without receiving any further answers at certain time (this point is decided by the asker or by an automatic timeout in the Q&A system). At the "closed" stage, either the asker or a voting procedure from other users can select a "best answer". The question becomes "resolved" after a best answer is selected. A question can also be awarded a "star" by any user at any time, marking it as an interesting question.

We collect 169,103 resolved questions posted in April, 2013. Each of these questions has at least 5 answers and the "best answer" is selected according to the votes of other users. We also randomly select 4 subcategories and take the questions, answers and users under each

TABLE I

Statistics of Yahoo! Answers datasets.

Subcategory	#questions	#answers	#users	#stars	#votes
All	169,103	1,380,082	263,512	53,035	577,710
Religion & Spirituality	15,926	187,627	22,883	7,318	72,360
Politics & Government	8,823	94,374	11,722	3,269	44,261
Baby Names	3,913	44,687	12,380	1,089	13,831
Dogs	3,291	25,076	9,703	218	11,223

subcategory as a small dataset. The data is publicly available on Yahoo! Answers and the basis statistics are shown in Table I.

2.3.2 Verifying Interdependent Relationships

In this section, we conduct a data analysis on questions, answers and users under the subcategory of Religion & Spirituality. We investigate some of the basic principles that reveal the potential interdependent relationships among the three types of nodes in the heterogeneous Q&A network.

On Yahoo! Answers, a user can mark a question as an interesting one by awarding it a "star". The number of stars can reflect the popularity of a question. The red bars in Figure 2 indicate that about 97% of the questions have at most 5 stars. A user can also vote an answer as a "best answer". The number of votes can reveal the interestingness of an answer. The blue bars in Figure 2 show that around 98% of the answers have at most 3 votes. A user can be awarded more points if s/he provides more answers. The number of points can reflect the

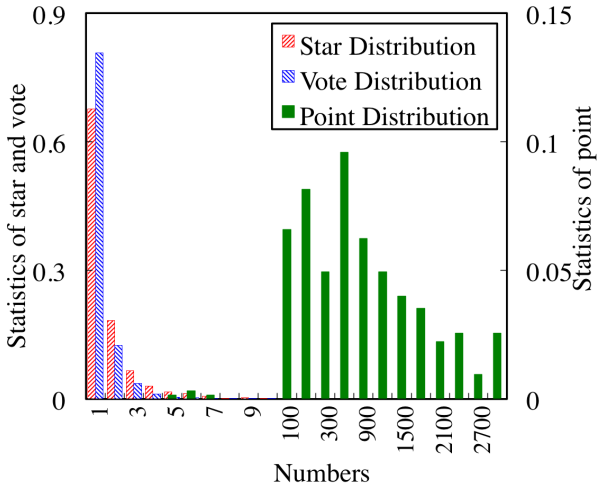
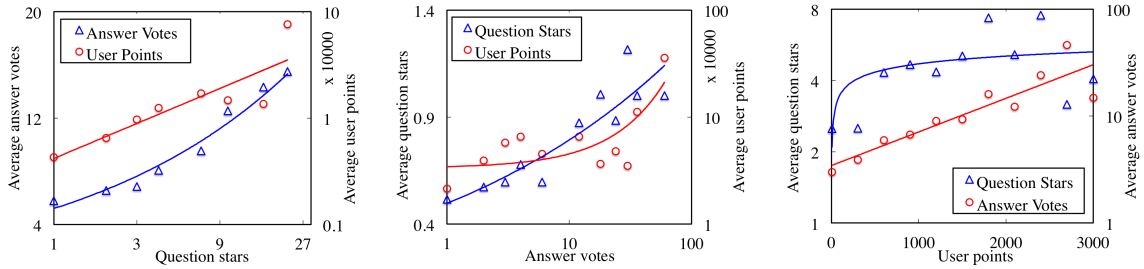


Figure 2. Data distributions on Yahoo! Answers.



(a) Influence on questions (b) Influence on answers (c) Influence on users

Figure 3. Interdependent relationships on Yahoo! Answers.

contribution of a user to some extent. We can observe that around 60% of the users get points less than 3000 from the green bars in Figure 2.

Based on these statistics, we first conduct a data analysis on questions. Figure 3 (a) reveals the influences of answers and users on questions. It can be observed that if a question has more stars, its answers will have more votes and the asker will also have more points. This

phenomenon illustrates that: (1) Popular questions often attract more interesting answers than those questions with little popularity. (2) Popular questions are likely to be asked by those high contribution users.

Then we analyze the influences of questions and users on answers. Figure 3 (b) shows the analysis. If an answer has more votes, the corresponding question usually has more stars and the user who provides the answer often has more points. This observation shows that interesting answers are usually given to those popular questions and they are often provided by those high contribution users.

We also do some analysis on the users with no more than 3000 points. Figure 3 (c) reveals the influences of questions and answers on users. If a user has more points, his/her questions often have more stars and his/her answers often have more votes. It illustrates that high contribution users would like to ask popular questions and post interesting answers.

The relationship between answers and users we observed is consistent with that in (15). Furthermore, we additionally capture the relationships between a question and its asker. Based on these observations, we propose the following principles to solve the co-ranking problem in Q&A networks:

1. We can identify the interestingness of an answer given the popularity of the corresponding question, plus the contribution of the user who posted the answer.

2. For a question, if we have the interestingness scores of its answers and the contribution score of its asker, we can infer its popularity, because a question is more popular if it attracts more interesting answers and the asker has more contribution to the system.

3. Now we go back to indicate a user's contribution. Intuitively, a user contributes more to the community if s/he asks more popular questions and gives more interesting answers. In contrast, one contributes less if one has few popular questions and interesting answers.

With these observations and principles, we next introduce how to model the qualities of different types of nodes in the network-based co-ranking framework.

2.4 Proposed Method

In this section, we propose an unsupervised **N**etwork-based **C**o-**R**anking model (**NCR**) and introduce the iterative computation algorithm for NCR. The above observations and principles serve as the base of the proposed model. NCR is designed in a divide-and-conquer way to decompose the co-ranking problem into three separate types of sub-modules (as shown in Figure 4) for questions, answers and users. The interdependencies are captured through iterative computations to help identify high quality objects in heterogeneous Q&A networks.

2.4.1 Question Popularity

Given a question q, we denote the set of answers for q as $\mathcal{S}_a(q) = \{a_i \mid \forall a_i, L_{QA}(q, a_i) = 1\}$. The influence of $\mathcal{S}_a(q)$ on the popularity of q is defined as the summation of the interestingnesses of all the answers in $\mathcal{S}_a(q)$,

$$\mathfrak{I}_a(q) = \frac{\sum_{k=1}^{|\mathcal{S}_a(q)|} \mathcal{I}(a_k)}{\mathcal{N}_{aq}} \tag{2.1}$$

where $\mid \mathcal{S}_a(q) \mid$ is the number of answers q has. We use $\mathcal{N}_{aq}$ to normalize $\mathfrak{I}_a(q)$ on the entire question set Q so their squares sum to 1: $\sum_{q \in Q} (\mathfrak{I}_a(q))^2 = 1$.

Similarly, the set of users for q can be denoted as $\mathcal{S}_u(q) = \{u_i \mid \forall u_i, L_{QU}(q, u_i) = 1\}$. Since each question can only be asked by a certain user, we can denote the only element as u_q. The influence of u_q on the popularity of q can be defined in a similar way as follows:

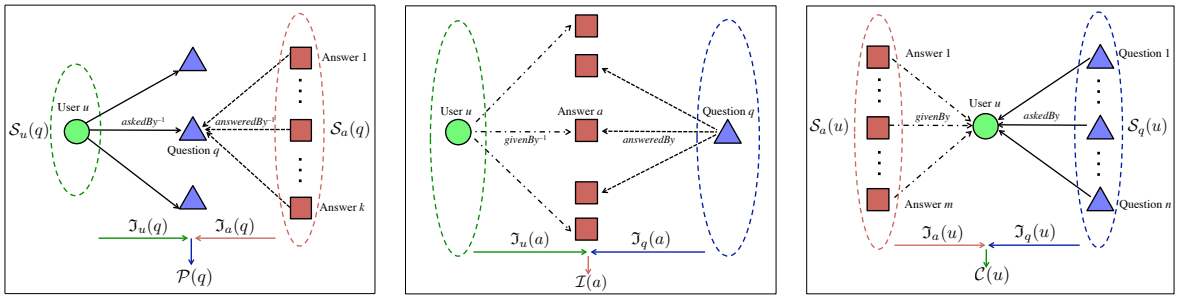
$$\mathfrak{I}_u(q) = \frac{\mathcal{C}(u_q)}{\mathcal{N}_{uq}} \tag{2.2}$$

where $\mathcal{N}_{uq} = \sum_{q \in Q}(\mathfrak{I}_u(q))^2$ is a normalization factor on the entire question set Q so their squares sum to 1.

Depending on the influences of answers and users, we can compute the popularity of q as follows:

$$\mathcal{P}(q) = \frac{\mathfrak{I}_a(q) + \mathfrak{I}_u(q)}{\mathcal{N}_q} \tag{2.3}$$

where $\mathcal{N}_q = \sum_{q \in Q}(\mathcal{P}(q))^2$ is a normalization factor on the entire question set Q so their squares sum to 1. In Equation 2.3, we add $\mathfrak{I}_a(q)$ and $\mathfrak{I}_u(q)$ together instead of multiplying them. The reason is that if a question is newly posted without any answers, its popularity can still be predicted by its asker's contribution. However, if we multiply $\mathfrak{I}_a(q)$ and $\mathfrak{I}_u(q)$ together, the popularity becomes 0, which will cause inaccurate rankings for new questions on Q&A sites. Figure 4 (a) summaries the entire process of computing the popularity score of a question q according to the influences of its answers and user. In order to calculate $\mathcal{P}(q)$, we need the interestingness values of q's answers, which we define next.



(a) Calculation of $\mathcal{P}(q)$ (b) Calculation of $\mathcal{I}(a)$ (c) Calculation of $\mathcal{C}(u)$

Figure 4. Calculation process in NCR model.

2.4.2 Answer Interestingness

To decide the interestingness of an answer, we have similar intuitions to the calculation of question popularity. Given an answer a, the answerer u belongs to U ($u \in U$) and $L_{AU}(a, u) = 1$. Similarly, the corresponding question q belongs to Q ($q \in Q$) and $L_{QA}(q, a) = 1$. We denote the influences of u and q on the interestingness of a as $\mathfrak{I}_u(a)$ and $\mathfrak{I}_q(a)$, respectively. According to Figure 3 (b), the interestingness of an answer is highly influenced by its question's popularity and its user's contribution. Figure 4 (b) represents how to obtain the interestingness of a via the influences of q and u.

$\mathfrak{I}_u(a)$, $\mathfrak{I}_q(a)$ and $\mathcal{I}(a)$ can be formulated in a similar way.

$$\mathfrak{I}_u(a) = \frac{\mathcal{C}(u_a)}{\mathcal{N}_{ua}} \tag{2.4}$$

$$\mathfrak{I}_q(a) = \frac{\mathcal{P}(q_a)}{\mathcal{N}_{qa}} \tag{2.5}$$

$$\mathcal{I}(a) = \frac{\mathfrak{I}_u(a) + \mathfrak{I}_q(a)}{\mathcal{N}_a} \tag{2.6}$$

where $\mathcal{N}_{ua}$, $\mathcal{N}_{qa}$ and $\mathcal{N}_a$ are three normalization factors to ensure the scale to be within the range (0, 1).

2.4.3 User Contribution

How do we judge a user? We usually consider two things when we look at a user. The first one is the answers the user provides. If the answers are very interesting and most of them are selected as "best answers", the user tend to contribute much to the Q&A community. In contrast, if the answers are always boring (like "Yes", "No" or "What?"), we may doubt the contribution of the user. The second factor is the questions the user asks. Those users are likely to contribute more if their questions are popular with lots of interesting answers. Based on these observations, we model a user's contribution by considering both his/her answers and questions. Figure 4 (c) shows how to calculate the contribution score of a user via the influence of his/her answers and questions.

Given a user u, we denote the set of answers u gives as $\mathcal{S}_a(u) = \{a_i \mid \forall a_i, L_{AU}(a_i, u) = 1\}$. We formulate the influence of $\mathcal{S}_a(u)$ on the contribution of u as the summation of the interestingness scores of all answers in $\mathcal{S}_a(u)$,

$$\mathfrak{I}_a(u) = \frac{\sum_{m=1}^{|\mathcal{S}_a(u)|} \mathcal{I}(a_m)}{\mathcal{N}_{au}} \tag{2.7}$$

where $\mid \mathcal{S}_a(u) \mid$ is the number of answers u provides. We use $\mathcal{N}_{au}$ to normalize $\mathfrak{I}_a(u)$ on the entire user set U so their squares sum to 1: $\sum_{u \in U} (\mathfrak{I}_a(u))^2 = 1$.

Similarly, the set of questions u asks can be denoted as $\mathcal{S}_q(u) = \{q_i \mid \forall q_i, L_{QU}(q_i, u) = 1\}$, and the influence of $\mathcal{S}_q(u)$ on the contribution of u is the summation of the popularity scores of all questions in $\mathcal{S}_q(u)$,

$$\mathfrak{I}_q(u) = \frac{\sum_{n=1}^{|\mathcal{S}_q(u)|} \mathcal{P}(q_n)}{\mathcal{N}_{qu}} \tag{2.8}$$

where $\mid \mathcal{S}_q(u) \mid$ is the number of questions u asks. $\mathcal{N}_{qu} = \sum_{u \in U} (\mathfrak{I}_q(u))^2$ is used to normalize $\mathfrak{I}_a(u)$ on the entire user set U so their squares sum to 1.

Depending on the influences of answers and questions, we can compute the contribution score of u as follows:

$$\mathcal{C}(u) = \frac{\mathfrak{I}_a(u) + \mathfrak{I}_q(u)}{\mathcal{N}_u} \tag{2.9}$$

where $\mathcal{N}_u = \sum_{u \in U} (\mathcal{C}(u))^2$ is a normalization factor on the entire user set U.

2.4.4 Iterative Computation Framework

Integrating all the information of the heterogeneous Q&A network together, NCR adopts an iterative method to compute question popularity, answer interestingness and user contribution, by exploring the interdependent relationships among them. The iterative computation framework is summarized in Algorithm 1. The time complexity of the NCR algorithm is $O(t|E|)$. Here t is the number of iterations and $|E|$ is the link numbers in the Q&A network. Through our experiments, the algorithm converges after 3 rounds in most cases.

2.4.5 Parallel Computing of NCR

The proposed NCR model computes the quality of each object in Q&A sites according to the quality of its linked objects. The very design of NCR makes the entire network naturally

Algorithm 1 The NCR algorithm

Input: A heterogeneous Q&A network $G = (V, E)$, maximum # of iteration Max_It
Output: The set of *popularity* $\mathcal{P}$, *interestingness* $\mathcal{I}$ and *contribution* $\mathcal{C}$

1: //Initialization step
   Initialize question's popularity, answer's interestingness and user's contribution to 1
2: **while** NOT converged or #iteration ¡= Max_It **do**
3: Update $\mathcal{P}$ using Equation 2.1, Equation 2.2 and Equation 2.3
4: Update $\mathcal{I}$ using Equation 2.4, Equation 2.5 and Equation 2.6
5: Update $\mathcal{C}$ using Equation 2.7, Equation 2.8 and Equation 2.9
6: **end while**

splittable when NCR calculates the qualify of objects. As shown in Figure 4, the ranking results for questions, answers and users can be computed in parallel at each iteration. Therefore, by extending NCR to a distributed version, we can reduce the runtime significantly. In the experiments, we choose to implement the distributed version of NCR on Apache Hadoop Platform using Pig Latin[1]. Pig Latin is a high-level programming language. It allows the developer to specify how the algorithm is performed, while the Pig complier transforms the specifications into Map-Reduce programs. The runtime of the Map-Reduce jobs depends on how the data is split and stored on the cluster (16). In this section, we test the scalability of the distributed NCR algorithm by generating various numbers of data splits.

[1] `http://pig.apache.org/`

2.5 Experiments

In this section, we conduct extensive experiments to evaluate the proposed NCR framework. After introducing the experiment settings and the evaluation metric, we compare different ranking methods. Furthermore, we also study the efficiency of NCR on the large-scale network.

2.5.1 Experimental Setup

We test the effectiveness of the proposed NCR model on datasets showed in Table I. In order to evaluate the performance, we have to generate ranking lists for questions, answers and users respectively from real Yahoo! Answers datasets. Since (6) considers stars as one metric for question quality, we use stars as the ground truth for question popularities. Wang et al. (8) use "best answer" labels to evaluate the quality of answers. However, only one "best answer" is selected for a question and all the other answers have the same labels. It will be biased to evaluate our ranking results using such labels. Since a "best answer" is selected through a voting procedure from other users, we use vote numbers as the ground truth for answer interestingness in our experiments. For users, we generate the ground truth of a user by considering both the stars of questions s/he asked and the votes of answers s/he provided because they reflect the contribution of users in (17). With such rich "human labelings" on Yahoo! Answers, we average these values and then convert them into integers in a scale from level 1 (the lowest quality) to level 4 (the highest quality) as in (6).

We set up the evaluation criterion using normalized discounted cumulative gain (nDCG) (18). nDCG is a popular measure in information retrieval tasks, and it focuses on correct rankings of high quality nodes. We compare the result of NCR with the ground truth for

each type of node. For questions and users, we can obtain a single ranking list with the entire participating nodes and then calculate the nDCG values. However, it is different for the answers, since an answer is only useful to its corresponding question. If we rank all the answers together, it would be meaningless. So for each question, we rank its answers and calculate the nDCG value. Then we use the average nDCG value as the final evaluation of answer quality.

2.5.2 Compared Methods

In order to demonstrate the effectiveness of our NCR approach, we compare the following methods:

- PR: This method is proposed in (19) which uses PageRank algorithm (14) to rank the contribution of users. A directed post-reply network is constructed by viewing each user as a node and linking the asker to everyone who replied to his/her questions. Note a user may answer more than one question posted by another user, the frequencies one replies another are considered as weights to edges as in (19). Since it is difficult to create a meaningful homogeneous network of questions (answers), we only run this method on the ranking task of user contribution.
- HITS (A): This baseline method is also used in (19). It ranks the contribution of users according to the authority value of HITS algorithm (20). In the post-reply network of users, a good authority is a user who helps many good askers (hubs) and s/he may have high contribution by providing useful answers according to Definition 7.
- HITS (H): We compare with another baseline using HITS algorithm. We use hub values of HITS to correspond to contribution ranks of users. A good hub in the post-reply network

is a user who is helped by many good answerers (authorities). Such user may also have high contribution by asking popular questions according to Definition 7.

- HITS (UQ): This method is extended from HITS to rank both the users and questions in a heterogeneous network containing users and the questions they answered. If a user u_i answered a question q_j, we add a link from u_i to q_j. A good authority is a popular question that attracts many good users (hubs) to answer, and a good hub is a user who provides answers to many good questions (authorities). We can only generate such a heterogeneous network with two different types of nodes since users and the questions they answered are n to n relationships while any other two types of nodes are not.

- RAT: This baseline method is the **R**anking of **A**nswer **T**ime (RAT) for answers' interestingness. It is derived from (15), which shows that high quality answers often arrive earlier. RAT algorithm ranks the interestingness of answers for a question according to their posting time.

- RCS: This method is the **R**anking of **C**osine **S**imilarity (RCS) for answers' interestingness. It is based on the assumption that a high quality answer and the corresponding question often focus on similar content. So we rank each answer according to its cosine similarity with its question.

- RAD: This method is the **R**anking of **A**verage **D**iversity (RAD) for questions' quality. We assume that a popular question usually attracts many diverse answers. We define the diversity between two answers as $div(a_i, a_j) = 1 - cosine(a_i, a_j)$. A larger value means

greater diversity between two answers of a question. RAD algorithm ranks the popularity of each question according to the average diversity on all of its answer pairs.

All these baseline methods ignore the interdependent relationships in the heterogeneous Q&A network (e.g., PR, CSR, etc.) or only consider incomplete relationships (e.g., HITS(UQ)).

2.5.3 Performance Evaluation

In this subsection, we study the effectiveness of the proposed NCR method. Table II presents the comparison results for question popularity, answer interestingness and user contribution under the subcategory of Religion & Spirituality. Due to space limit, we only show the performances of question popularity and user contribution from top 10% to 50% in Table II (a) and (c). Since each question has at least 5 answers, we report the average nDCG value from top 1 to top 5 for interestingness of answers in Table II (b). It can be observed that NCR consistently outperforms other baseline methods on the rankings of all the three types of nodes.

In particular, compared with the three baseline methods PR, HITS (A) and HITS (H), all of which are focusing on the homogeneous network of users, NCR can achieve the best performance as shown in Table II (c). It illustrates that the interdependent relationships among questions, answers and users can help improve the ranking result of users. Moreover, although PR and HITS (A) can identify high contribution users with good results, NCR can still perform better than these two baseline methods with an improvement of at least 3.5%. Furthermore, the performance of NCR is significantly better than that of HITS (H) with an improvement of at most 24%.

TABLE II

Ranking performances under subcategory of Religion & Spirituality.

(a) Question popularity

Criterion	Top k%	Methods: HITS (UQ)	RAD	NCR
nDCG ↑	10	0.641	0.640	**0.715**
	20	0.691	0.705	**0.757**
	30	0.718	0.729	**0.766**
	40	0.774	0.780	**0.811**
	50	0.819	0.830	**0.855**

(b) Answer interestingness

Criterion	Top k	Methods: RAT	RCS	NCR
Average nDCG ↑	1	0.563	0.581	**0.613**
	2	0.695	0.716	**0.732**
	3	0.748	0.770	**0.781**
	4	0.785	0.806	**0.814**
	5	0.814	0.833	**0.839**

(c) User contribution

Criterion	Top k%	Methods: PR	HITS (A)	HITS (H)	HITS (UQ)	NCR
nDCG ↑	10	0.866	0.863	0.721	0.764	**0.897**
	20	0.786	0.776	0.663	0.686	**0.824**
	30	0.782	0.770	0.677	0.695	**0.828**
	40	0.801	0.794	0.722	0.733	**0.850**
	50	0.840	0.834	0.780	0.789	**0.890**

Compared with the baseline method HITS (UQ) considering the heterogeneous network with two types of nodes (questions and users), NCR can still have better performances on both questions and users as shown in Table II (a) and (c). It reveals that questions, answers and users are tightly interconnected with each other and ignoring either one type of object would weaken the co-ranking performance.

Compared with the other baseline methods RAT, RCS and RAD, which do not consider the network structure in the ranking task, NCR still achieves the best results. It shows that the interdependent relationships in the heterogeneous network are more powerful and helpful than the time information (used in RAT) and the text information (used in RCS and RAD) in detecting high quality nodes. Moreover, extracting meaningful features from text information is challenging and time-consuming. With large volumes of new questions and answers, making use of text information efficiently becomes more and more difficult.

We further show the performances of the proposed NCR model under another three subcategories in Table III, Table IV and Table V. We can observe that the performances of NCR are

TABLE III

Ranking performances under subcategory of Politics & Government.

(a) Question popularity

Criterion	Top k%	Methods: HITS (UQ)	RAD	NCR
nDCG ↑	10	0.634	0.605	**0.688**
	20	0.687	0.672	**0.731**
	30	0.732	0.720	**0.761**
	40	0.788	0.783	**0.818**
	50	0.830	0.830	**0.862**

(b) Answer interestingness

Criterion	Top k	Methods: RAT	RCS	NCR
Average nDCG ↑	1	0.569	0.577	**0.599**
	2	0.704	0.717	**0.730**
	3	0.758	0.774	**0.781**
	4	0.795	0.811	**0.816**
	5	0.824	0.838	**0.843**

(c) User contribution

Criterion	Top k%	Methods: PR	HITS (A)	HITS (H)	HITS (UQ)	NCR
nDCG ↑	10	0.892	0.900	0.738	0.800	**0.927**
	20	0.859	0.867	0.685	0.756	**0.891**
	30	0.842	0.848	0.699	0.750	**0.873**
	40	0.843	0.847	0.728	0.764	**0.870**
	50	0.880	0.881	0.783	0.809	**0.906**

TABLE IV

Ranking performances under subcategory of Baby Names.

(a) Question popularity

Criterion	Top k%	Methods: HITS (UQ)	RAD	NCR
nDCG ↑	10	0.580	0.355	**0.857**
	20	0.519	0.396	**0.828**
	30	0.520	0.476	**0.844**
	40	0.594	0.557	**0.884**
	50	0.673	0.608	**0.907**

(b) Answer interestingness

Criterion	Top k	Methods: RAT	RCS	NCR
Average nDCG ↑	1	0.548	0.581	**0.637**
	2	0.700	0.733	**0.767**
	3	0.758	0.792	**0.819**
	4	0.797	0.829	**0.851**
	5	0.827	0.855	**0.874**

(c) User contribution

Criterion	Top k%	Methods: PR	HITS (A)	HITS (H)	HITS (UQ)	NCR
nDCG ↑	10	0.766	0.762	0.675	0.676	**0.820**
	20	0.740	0.730	0.662	0.655	**0.776**
	30	0.804	0.794	0.732	0.723	**0.829**
	40	0.847	0.841	0.792	0.780	**0.872**
	50	0.877	0.873	0.837	0.827	**0.903**

the best under all these subcategories except that of user contribution under the subcategory of Dogs. Though NCR does not perform well for user contribution, it still achieves the best results for question popularity and answer interestingness. Moreover, Table IV (a) shows that NCR can achieve an improvement of at most 140% on question quality under the subcategory of Baby Names. We also present the result on the entire Yahoo! Answers data in Table VI. It reveals that NCR is robust and stable regardless of any category information.

TABLE V

Ranking performances under subcategory of Dogs.

(a) Question popularity

Criterion	Top k%	Methods: HITS (UQ)	RAD	NCR
nDCG ↑	10	0.586	0.630	**0.688**
	20	0.637	0.705	**0.728**
	30	0.709	0.767	**0.804**
	40	0.755	0.809	**0.831**
	50	0.801	0.828	**0.857**

(b) Answer interestingness

Criterion	Top k	Methods: RAT	RCS	NCR
Average nDCG ↑	1	0.573	0.607	**0.620**
	2	0.714	0.749	**0.776**
	3	0.773	0.810	**0.835**
	4	0.816	0.849	**0.873**
	5	0.852	0.879	**0.897**

(c) User contribution

Criterion	Top k%	Methods: PR	HITS (A)	HITS (H)	HITS (UQ)	NCR
nDCG ↑	10	**0.761**	0.732	0.449	0.664	0.678
	20	**0.743**	0.719	0.498	0.663	0.680
	30	**0.795**	0.777	0.581	0.727	0.755
	40	**0.842**	0.827	0.665	0.784	0.816
	50	**0.876**	0.865	0.737	0.829	0.857

TABLE VI

Ranking performances for all topic categories.

(a) Question popularity

Criterion	Top k%	Methods: HITS (UQ)	RAD	NCR
nDCG ↑	10	0.661	0.657	**0.710**
	20	0.739	0.719	**0.766**
	30	0.771	0.742	**0.788**
	40	0.835	0.805	**0.850**
	50	0.880	0.854	**0.894**

(b) Answer interestingness

Criterion	Top k	Methods: RAT	RCS	NCR
Average nDCG ↑	1	0.574	0.597	**0.618**
	2	0.723	0.750	**0.764**
	3	0.785	0.811	**0.821**
	4	0.827	0.850	**0.858**
	5	0.860	0.878	**0.885**

(c) User contribution

Criterion	Top k%	Methods: PR	HITS (A)	HITS (H)	HITS (UQ)	NCR
nDCG ↑	10	0.844	0.766	0.632	0.793	**0.856**
	20	0.807	0.758	0.616	0.774	**0.822**
	30	0.801	0.769	0.626	0.779	**0.817**
	40	0.833	0.812	0.672	0.818	**0.850**
	50	0.872	0.860	0.747	0.864	**0.892**

In summary, with the help of interdependent relationships among questions, answers and users, NCR always outperforms the baseline methods. In the next subsection, we investigate more details about the efficiency of the proposed NCR framework.

2.5.4 Scalability Performance

In this subsection, we evaluate the runtime efficiency of two versions of NCR: single-NCR runs in Python on a single node Server (Intel Xeon^{TM} Quad-Core CPUs of 2.26GHz and 36GB RAM) and parallel-NCR runs on a multi-node Hadoop cluster (Intel Xeon^{TM} 2×Quad-Core

TABLE VII

Scalability performance of NCR.

Methods	Implementation	# of nodes	Runtime (seconds)
Single-NCR	Python	1	11,969
Parallel-NCR	Pig Latin	8	1,593

CPUs of 2.50GHz and 16GB RAM) in Pig Latin. Parallel-NCR algorithm is complied into Map-Reduce jobs and executed over Hadoop in a distributed fashion. We use the entire dataset with questions from all categories and report the runtime of these two versions in Table VII. Since the ranking results are the same no matter we run NCR in local or in parallel, we only present the runtime (in seconds) in Table VII. It can be observed that with 8-node Hadoop cluster, parallel-NCR is 6.5 times faster than single-NCR.

We also test the scalability of parallel-NCR with different mappers. We set the data split size (mega byte) to be 6 different numbers from 5 to 30, with an interval of 5. The smaller the data split size is, the larger the number of mappers is. Figure 5 summarizes the performance of mapping time with different numbers of mappers. It can be seen that with more mappers, the mapping process becomes much more efficient. For example, if 70 mappers are used to run parallel-NCR, the mapping procedure costs 324 seconds. If we use more mappers such as 129 mappers, the mapping time is reduced to 252 seconds. However, the communication time will increase if we use more mappers. So in our experiment, we set the data split size as 10Mb. The total number of mappers are 129.

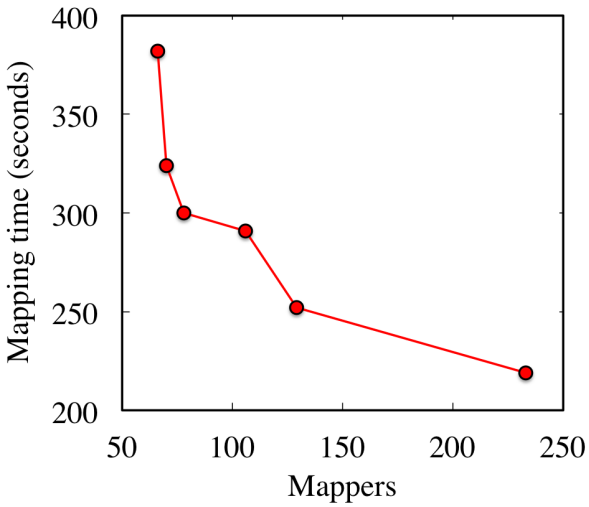


Figure 5. The mapping time with different number of mappers.

In summary, as the network gets larger, parallel-NCR can efficiently scale up the proposed framework to the growth of Q&A sites.

2.6 Related work

User communities in online Q&A websites have already been investigated from several perspectives. The first is the study of user's interests and motivations for contribution (21; 22; 23; 24) in the community. These studies model the authority, reputation and expertise of users in social networks and communities (25; 26; 27). The second is the study of user quality in Q&A sites by developing several link-based ranking algorithms (17; 28; 29; 19). For example, Zhang et al. (19) focus on the data from Java forum and construct a post-reply network, in which the nodes correspond to users and the links represent interactions between askers and answerers. Both ExpertiseRank (a PageRank-like algorithm) and HITS are applied in (19) to

identify users with high expertise. Jurczyk and Agichtein (17) also apply the HITS algorithm to user communities and they aim to discover authoritative users in topical categories. However, all these studies focus on ranking only users by extracting a homogeneous user network from Q&A sites. Our study is different from them since we model the Q&A site as a heterogeneous network and co-rank questions, answers and users in this network.

The study of content quality in Q&A websites is also related to our work. It can be categorized into two branches. The first branch investigates how to evaluate or predict the question quality (9; 6; 30). Agichtein et al. (9) analyze the essential features related to questions and propose a supervised method to identify high quality content in Q&A sites. (6) evaluates the question quality using a mutual reinforcement-based label propagation algorithm. (30) predicts the subjectivity of questions based on a co-training model. The other branch aims to find high quality answers (7; 31; 8; 32). Jeon et al. (7) extract non-textual features to predict the quality of answers. Sekai et al. (33) apply the graded-relevance metrics to evaluate the answer quality. Bian et al. (31) introduce a ranking algorithm to retrieve high quality answers according to the user interaction and the answer relevance. Wang et al. (8) model the question-answer relationships via analogical reasoning and develop an answer ranking method. Suryanto et al. (32) aim to find good answers for newly-arrived questions by considering the answerer expertise. However, all these works do not study the quality of users. In addition, some of them (9; 6; 8; 30) require substantial amounts of manual supervision.

There are also a few research studies on the link-based ranking within heterogeneous networks. Studies in (34; 35) focus on the bibliographic data with three different types of objects,

authors, publications and conferences. Zhou et al. (34) uses the heterogeneous network of authors and publications to co-rank these two types of objects while (35) ranks authors and conferences by constructing another heterogenous network. Yin et al. (36) create a heterogeneous network of facts and websites to help discover truth in multiple conflicting sources on the Web. Wang et al. (37) detect review spammers based on a heterogeneous online store review network. However, the network representations in these studies are very different from that of the Q&A site, therefore their techniques are not applicable to our work. Bian et al. (12) utilize the relationships among questions, answers and users to estimate the quality of these three types of objects. It is most related to our work. However, (12) uses a semi-supervised method while our work aims to rank the different types of objects in a totally unsupervised way. Another difference is that (12) separates the contribution of users into two groups, the contribution of askers and those of the answerers. But this separation is not realistically reasonable. User contributions should be considered as integrations of their question asking and answering behaviors. In our work, we quantify the contribution of users according to both their asking and answering activities.

CHAPTER 3

LEARNING ENTITY TYPES FROM QUERY LOGS

(This chapter was previously published as "Learning Entity Types from Query Logs via Graph-Based Modeling", in *Proceedings of the 24th ACM International Conference on Conference on Information and Knowledge Management (CIKM'15)* (2). DOI: https://doi.org/10.1145/2806416.2806498.)

3.1 Introduction

An *entity* is something that exists in itself, actually or potentially, concretely or abstractly, physically or not [1]. Entities are forming the building block for various web applications. Yelp [2] is building on top of a corpus of *l*ocal class entities (*e.g.*, the restaurant entity "The French Laundry", the Point of Interest entity "Golden Gate Bridge", etc.) associated with user reviews. IMDB [3] has a large corpus of *m*ovie and *a*ctor class entities. Modern search engines like Bing, Google and Yahoo! start building Knowledge Graph containing a large collection of diverse types of entities. When a user issues a question about an entity (*e.g.*, "net worth of Bill Gates" or "phone number of Gary Danko"), the search engine can retrieve results directly from the knowledge graph, satisfying the user's need and providing better user experience. Recent study

[1] http://en.wikipedia.org/wiki/Entity

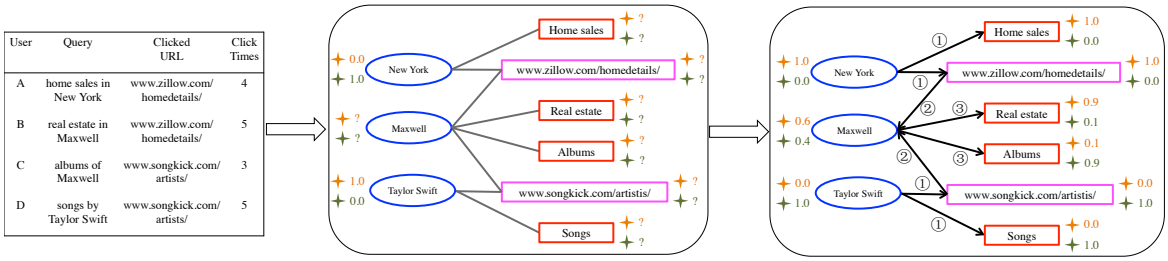
[2] http://www.yelp.com/

[3] http://www.imdb.com/

shows that around 70% of the queries contain entity information (38; 39). Hence, the coverage of entities is very important for these applications. Moreover, knowing the exact types of entities can help the application decide the best way in presenting results to users.

Various entity repositories, ranging from the more general collaborative knowledge bases such as Wikipedia and Freebase to the domain-specific corpora such as IMDB and Yelp, are widely used to extract entity information and aggregate the information into a comprehensive knowledge graph. However, there are several problems with this approach: (a) *c*overage: it is one of the key metric in measuring the quality of knowledge. Knowledge bases like Wikipedia and Freebase primary focus on popular entities from a few limited types, while other domain-specific corpora are more expensive to obtain. Plus, little information exists in knowledge bases for many less popular entities or newly generated entities, such as a new music title. It is difficult to identify and extract such entities in time; (b) *a*mbiguity: multiple types of entity are often associated with the same string collected from the same or different sources. For example, the token "Chicago" is not only a city entity, but also a movie entity or a rock band entity. How to separate them apart in case little is known about the types of the entities, and how to rank these entities according to the *p*opularity and/or user intent, are both quite challenging; (c) *d*iscrepancy: errors may exist due to user-generated contents via crowdsourcing, thus information extracted from these sources may be noisy and inconsistent.

A lot of research work in the literature tries to overcome the above challenges from different perspectives. The existing knowledge bases could only cover a fraction of the whole entity space. In order to expand the size of knowledge bases, many automatic techniques have been



(a) Search query logs (b) Graph construction (c) Learning steps

Figure 6. An intuitive example of learning entity types from search query logs.

proposed to discover entities and their types from different large corpora, such as news articles and web pages (40; 41; 42). In addition, disambiguating entities from news reports is also studied in (43; 44). Other sources such as search query logs can also be leveraged to extract and disambiguate entities. Since the search engine has become the main information source for most people to look for information, search query logs can be a nice complementary source for extracting new entity information as well as learning entity popularity and disambiguating entities. A few state-of-the-art approaches are proposed to classify and disambiguate entities in query logs (45; 46; 47). For instance, intent-based Model (IM) (46) predicts entity type distributions by jointly modeling user intent and entity types via probabilistic inference in a graphical manner. Fast Entity Linker (FEL) is proposed in (47) to disambiguate entities by linking queries to entities in a knowledge base. However, these methods do not fully explore the importance of auxiliary signals in query logs, i.e. the structural language patterns (contextual word patterns) in queries and the clicked domains from relevant web URL results. For example, given the query "menu of Purple Pig" and a user's clicked domain URL "yelp.com", both the

pattern "menu of" and the clicked URL help predicting "Purple Pig" as a local restaurant entity. Therefore, knowing the types of these important signals can help mining entity types from query logs more effectively.

In this chapter, we model search query logs into a bipartite graph to encode relations between entities and important signals. Two kinds of nodes, entities and their auxiliary information, are contained in the constructed bipartite graph shown in Figure 6 (b). The four-pointed orange and blue stars mean the person and place types, respectively. The number next to a star shows the probability of a node belonging to a type. "1.0" means the type is already known and "?" means that we need to learn the type from search query logs. With such a bipartite graph, we can take advantage of the encoded relations (1) to learn entity types. Moreover, the type information can also be assigned to auxiliary nodes, thereby helping disambiguating entities via user-generated texts (*e.g.*, contextual words) and user feedbacks (*e.g.*, clicked URLs). In this chapter, we apply a graph-based **L**abel **P**ropagation (LP) method to simultaneously learn types of both entities and auxiliary signals. Figure 6 (c) shows the steps of LP in an intuitive way. Each black arrowed line shows the propagation direction and the circled number on each line represents the order of the propagation process. Given a small number of prior-known entities, the types of these entities are first propagated to the connected auxiliary nodes, and then the types are propagated back from auxiliary nodes to unknown entities. Despite the simple idea, mining entity types from the built graph is still a challenging task due to the following reasons:

- Queries are short texts, and information related to entities is usually very sparse. It is non-trivial to explore the hidden connections among entities and auxiliary information in search logs.
- Large amounts of irrelevant information exists in search logs, bringing noise in detecting entity types. It is difficult to discover and remove such noisy information from search logs.

In order to address these two issues, we propose an **E**nsemble framework based on **L**abel **P**ropagation (ELP) to simultaneously learn types of both entities and auxiliary signals. Specifically, we design two separate strategies to fix the problems of sparsity and noise in query logs, respectively.

In summary, our contributions are as follows:

- We represent query logs as a bipartite graph about entities and their auxiliary signals. We leverage such interconnected relationships between entities and their auxiliary signals to learn both entity types and auxiliary node types together.
- We propose an **E**nsemble framework based on **L**abel **P**ropagation (ELP) and design two separate strategies in ELP to effectively learn node types from search query logs.
- We conduct extensive empirical studies on search logs from a real-world search engine to demonstrate the effectiveness of the proposed ELP framework. In addition, some case studies show that ELP can learn important word patterns for different types of entities, as well as disambiguating entities via the connected auxiliary informaiton.

3.2 Background

In this section, we first introduce several related concepts and notations. Then, we will formally define the problem of learning node types from a bipartite entity-auxiliary graph extracted from query logs.

Definition 1. **A Bipartite Entity-Auxiliary (EA) Graph**: A bipartite entity-auxiliary (EA) graph is represented as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. $\mathcal{V}$ is the set of nodes (objects), including two types of objects, i.e., entities $E = \{e_1, ..., e_M\}$ and auxiliary signals $A = \{a_1, ..., a_N\}$. $\mathcal{E} \subseteq E \times A$ is the set of links (relations) between the nodes in V, which involves the *associatedWith* link between entities and auxiliary signals. Let $\mathcal{W}$ denote an $M \times N$ weight matrix, in which element w_{ij} equals the frequency associating e_i and a_j.

Figure 6 (b) shows an example of a bipartite EA graph extracted from the search query logs in Figure 6 (a). Three entities are connected with six auxiliary signals, including four contextual words and two clicked URLs.

In EA graph, each entity has at least one type (label) in reality. We assume there are K labels for entities ($K \geq 2$) and represent entity types as $\mathcal{Y} \in \mathbb{R}^{M \times K}$. $y_{ik} \in \mathcal{Y}$ is a non-negative real number indicating the probability that entity e_i belongs to label k. In practice, a small set of entities (seed entities) in the graph may be manually labeled with their types. We denote the labeled entity set as E_L. In Figure 6 (b), both "New York" and "Taylor Swift" are considered as seed entities. We use $\mathcal{Y}^0$ to denote an instantiation of $\mathcal{Y}$ that is consistent with the seed labels. Given an entity $e_i \in E_L$ with n labels ($n \geq 1$), we set y^0_{ik} as $1.0/n$ if e_i has label k, otherwise 0. Given the entity $e_i \in E \backslash E_L$ without labels, we have $y^0_{ik} = 0$ for any label k.

From the existing search logs D, we can extract a bipartite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and get $\mathcal{Y}^0$ according to some seed entities E_L. Our goal is to learn entity types $\mathcal{Y}$ from $\mathcal{G}$. Since the auxiliary nodes can also carry labels with them to indicate the important interconnections between entities and the auxiliary signals, another goal is to assign labels to those auxiliary nodes in $\mathcal{G}$. We use $\mathcal{Z} \in \mathbb{R}^{N \times K}$ as labels of auxiliary nodes, where the element z_{jk} is a non-negative real number indicating the probability that a_j relates to label k. Thus our ultimate goal becomes to estimate $\mathcal{Y}$ and $\mathcal{Z}$ given $\mathcal{G}$ and E_L. In order to solve this problem, we apply a graph-based **L**abel **P**ropagation (LP) method to leverage these important auxiliary signals via their connections with the target entities as shown in Figure 6 (c).

3.3 Proposed Method

In this section, we propose an **E**nsemble framework based on **LP** (ELP) to simultaneously learn types of both entities and auxiliary signals from query logs. Before proceeding, we first introduce how to build the entity-auxiliary graph from real-world search logs.

3.3.1 Graph Construction

Given search logs, we first have to extract entities from queries. Several methods are applied to find entities in this chapter. First, we use a part-of-speech tagger (48) to extract contiguous words of proper nouns, common nouns and capitalized words (45; 49) to form noun phrases. Second, we match the extracted noun phrases according to a dictionary of entities built from knowledge bases, such as Wikipedia, Freebase and Yelp. We do not use the type information in those knowledge bases. We assume that the types of entities are unknown in the experiments. These methods help us detect entities in high precision. Besides, we can use a more complex

model in (50) to identify the entity and the background part (i.e., contextual words). In the example of the search logs in Figure 6 (a), we extract "New York", "Maxwell" and "Taylor Swift" as entities. Thus, "home sales", "real estate", "albums" and "songs" are considered as contextual words. In our experiments, we use both the uni-gram and binary-grams of contexts as auxiliary nodes. The stop-word nodes are removed from our graph.

In search logs, clicked URLs are also very important for learning entity types. Since each clicked URL may have several levels of domain names to point to a certain webpage, there will be too many redundant nodes of clicked URLs in the constructed graph. Therefore, we group a set of URLs into a single auxiliary node if they have exactly the same top- and second-level domain names. In Figure 6 (a), we only show the first two domain names for the clicked URLs. We use the frequencies of entities and auxiliary nodes appearing together in the query logs as weights of corresponding edges.

Such a bipartite graph helps encode relations between entities and important auxiliary signals from search query logs. We can take advantage of the encoded relations to discover entity types by applying the graph-based LP method. However, directly applying LP may not be satisfying due to the following issues in query logs:

1. Queries are short texts, and information related to entities is usually very sparse. LP may not propagate labels adequately. Therefore, it is necessary to explore the hidden connections in EA graph.

2. Large amounts of irrelevant information exists in search logs, bringing noise in detecting entity types. LP may propagate errors out and enlarge the error information due to the noise. Hence, it is imperative to discover and remove such noisy information from the EA graph.

In the following, we first focus our attention on how to apply LP on the built graph to learn types of both entities and auxiliary signals simultaneously. Then we introduce two separate strategies LPA and LPD to address the problem of sparsity and noise in the EA graph respectively. After that, we describe the proposed ELP framework that takes advantage of the LPA and LPD strategies.

3.3.2 Methodology

3.3.2.1 The LP Method

The problem of learning with labeled and unlabeled data from graphs has been investigated in (51; 52; 53; 54; 55). The objective and algorithm of the LP method are heavily influenced by the works of (51; 53). Given the collection of search log data D, we can extract an entity-auxiliary graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a weight matrix $\mathcal{W}$ as introduced in Section 3.2. With a small set of seed entities E_L, we can initialize $\mathcal{Y}^0$. Our goal is to automatically estimate $\mathcal{Y}$ for entities and $\mathcal{Z}$ for auxiliary nodes according to $\mathcal{W}$ and $\mathcal{Y}^0$. We define a normalized frequency matrix as follows:

$$\mathcal{N} = \mathcal{D}^{-1/2}\mathcal{W}, \tag{3.1}$$

where $\mathcal{D}$ is a diagonal matrix and each element $d_{ii} \in \mathcal{D}$ is the sum of all the elements in the ith row (or column) of $\mathcal{W}\mathcal{W}^\top$. Intuitively, d_{ii} can be interpreted as the volume of all length-of-two

paths that start at e_i. The reason we use such a normalization is to guarantee the convergence of LP as shown in (53).

With the above definitions and notations, LP iteratively updates $\mathcal{Y}$ and $\mathcal{Z}$. For the t-th iteration, it first propagates the types of entities to the connected auxiliary nodes:

$$\mathcal{Z}^t = \mathcal{N}^\top \mathcal{Y}^{t-1}. \tag{3.2}$$

Then it propagates the types back to entities from the auxiliary nodes as follows:

$$\mathcal{Y}^t = \alpha \mathcal{N} \mathcal{Z}^t + (1-\alpha)\mathcal{Y}^0, \tag{3.3}$$

where α is a parameter to trade off the label consistency between the intrinsic graph structure and the seed entities. It has been shown in (53) that the sequence of $\mathcal{Y}^t$ asymptotically converges to:

$$\mathcal{Y}^* = (1-\alpha)(\mathbf{1} - \alpha \mathcal{D}^{-1/2} \mathcal{W} \mathcal{W}^\top \mathcal{D}^{-1/2})^{-1} \mathcal{Y}^0. \tag{3.4}$$

The time complexity of LP is $O(T|\mathcal{E}|)$, where T is the iteration number and $|\mathcal{E}|$ is the number of connections in the EA graph. Through our experiments, the algorithm converges after no more than 20 rounds in most cases. The LP method is summarized in Algorithm 2.

Once $\mathcal{Y}$ and $\mathcal{Z}$ are obtained, we normalize their elements to get the posterior probabilities $p(k|e_i)$ for $i = 1, ..., M$ and $p(k|a_j)$ for $j = 1, ..., N$ as follows:

Algorithm 2 The LP algorithm

Input: Search query log D, a set of seed entities E_L and a trade-off parameter α
Output: Label matrices $\mathcal{Y}$ and $\mathcal{Z}$

1: *//graph construction step*
   Build an entity-auxiliary graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ from D
2: *//initialization step*
   Initialize $\mathcal{Y}$ as $\mathcal{Y}^0$ according to E_L
3: Compute the weight matrix $\mathcal{W}$ from $\mathcal{G}$
4: Compute $\mathcal{N}$ from $\mathcal{W}$ according to Equation 3.1
5: *//iterative computation step*
6: **while** NOT converged **do**
7: *//propagation step from entities to auxiliary nodes*
   Compute $\mathcal{Z}^t$ according to Equation 3.2
8: *//propagation step from auxiliary nodes to entities* Compute $\mathcal{Y}^t$ according to Equation 3.3
9: **end while**
10: Normalize $\mathcal{Y}$ and $\mathcal{Z}$ according to Equation 3.5

$$
\begin{aligned}
p(k|e_i) &= y_{ik} / \sum_{l=1}^{k} y_{il}, \\
p(k|a_j) &= z_{jk} / \sum_{l=1}^{k} z_{jl}.
\end{aligned}
\tag{3.5}
$$

3.3.2.2 The Proposed LPA Strategy

Directly applying LP may not be satisfying because the connections extracted from query logs are very sparse and LP cannot propagate labels adequately. Therefore, we propose a strategy LPA (**L**abel **P**ropagation after **A**dding more connections) to explore the hidden con-

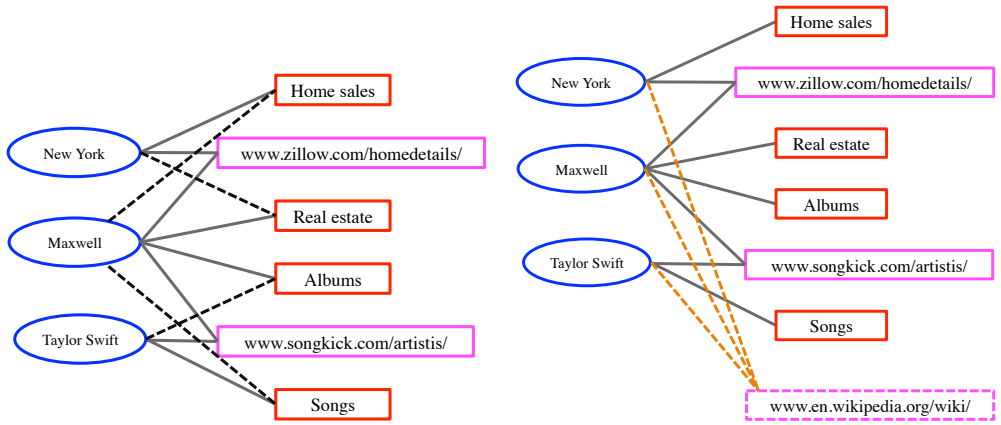
nections in the EA graph. We take advantage of the word2vec tool [1] to connect entities with more contextual words and help the LP model propagate labels more effectively.

The intuition behind LPA is that, if one entity e connects with one auxiliary node a_1 and a_1 has a high similarity with another auxiliary node a_2, we should connect e with a_2 to expand the connections in the bipartite EA graph. Hence, we need to measure the similarities among auxiliary nodes first. We focus on contextual words and measure their similarities according to the semantic meanings by the word2vec tool. The word2vec tool computes vector representations of words by implementing continuous bag-of-words and skip-gram architectures in an efficient manner (56; 57; 58). By calculating the distance between two vector representations, we can obtain the similarity value for two words. Hence, given the auxiliary node set $A = \{a_j\}_{j=1}^{N}$, we could get a similarity matrix $\mathcal{S} \in \mathbb{R}^{N \times N}$, where each element $s_{ij} \in \mathcal{S}$ denotes the similarity value between a_i and a_j. The above exploration of connections can be formulated as follows:

$$\mathcal{W}_{\mathcal{A}} = \mathcal{W} \times \mathcal{S}, \tag{3.6}$$

where $\mathcal{W}_{\mathcal{A}}$ is the updated weight matrix according to LPA. Intuitively, $w_{ij} \in \mathcal{W}_{\mathcal{A}}$ can be interpreted as the weight aggregation of all length-of-two paths from e_i to a_j via every $a_{j'} \in A$. Given the search logs in Figure 6 (a), we show an intuitive example of the updated graph according to LPA in Figure 7 (a). The dashed black edges in Figure 7 (a) represent the hidden connections explored by LPA. We assume that the contextual words "home sales" and "real

[1] https://code.google.com/p/word2vec/



(a) The LPA strategy (b) The LPD strategy

Figure 7. Two seperate intuitive examples for LPA and LPD.

estate" are very similar so we connect "New York" with "real estate" in Figure 7 (a). With such a denser graph, we could run the LP algorithm to propagate labels more effectively. We denote the node types learned from LPA as $\mathcal{Y}_{\mathcal{A}}$ and $\mathcal{Z}_{\mathcal{A}}$ for entities and auxiliary nodes, respectively.

3.3.2.3 The Proposed LPD Strategy

Another issue of directly applying LP is that noise may exist in the built EA graph so that LP may propagate errors out and enlarge the error information. Therefore, we propose a strategy LPD (**L**abel **P**ropagation after **D**eleting noisy nodes) to discover and get rid of noisy information in the EA graph. The basic idea of LPD is to discover some multi-type auxiliary nodes and delete them with their corresponding connections in the constructed EA graph. Here multi-type nodes mean contextual words or clicked URLs that cover several types of entities. For example, "picture" relates to several entity types, such as media, location, and person. Hence, it is not informative to take such contextual word into consideration. We

apply a similar approach in (59) to get rid of some multi-type auxiliary nodes and update the graph accordingly. Specifically, we start by calculating the similarity (e.g., cosine similarity) between two entities according to the bag-of-word representations of their auxiliary nodes. Low similarity pairs are more likely to represent entities with different types. Hence, auxiliary nodes involved with such entities are not likely to be very specific. So we can consider low similarity pairs as voters and let the auxiliary nodes be the candidates. Each pair votes for its auxiliary nodes they share. The more votes an auxiliary node gets, the higher probability of multi-type it is. We can then apply a threshold to get rid of some auxiliary nodes and update the EA graph accordingly. Figure 7 (b) gives an intuitive example of the updated graph according to LPD. In this figure, the dashed box of the auxiliary node means that the node is multi-type, and the dashed orange edges represent the connections we should get rid of according to LPD. We assume the clicked URL "www.en.wikipedia.org/wiki/" is a multi-type node and delete it with its corresponding edges from the graph. Then we can run LP on such a cleaner graph so that the error information can be propagated out as little as possible. We denote the node types learned from LPD as $\mathcal{Y}_{\mathcal{D}}$ and $\mathcal{Z}_{\mathcal{D}}$ for entities and auxiliary nodes, respectively.

3.3.2.4 The Proposed ELP Framework

Given the proposed LPA and LPD strategies, we can simply combine them together to derive another two strategies, LPAD and LPDA. LPAD updates the graph by first adding more connections and then deleting noisy nodes. The node types learned from LPAD are denoted as $\mathcal{Y}_{\mathcal{AD}}$ and $\mathcal{Z}_{\mathcal{AD}}$ for entities and auxiliary nodes, respectively. LPDA updates the graph in an opposite way, i.e., first deleting noisy nodes and then adding connections based on the

remaining nodes. We denote the node types learned from LPDA as $\mathcal{Y}_{\mathcal{DA}}$ and $\mathcal{Z}_{\mathcal{DA}}$ for entities and auxiliary nodes, respectively.

Since each strategy has its advantage, we propose an **E**nsemble framework based on **LP** (ELP) to combine them together and maximize the margin (60). We run each strategy separately and select the best one as the final result for each node as follows:

$$\begin{aligned} \mathcal{Y} &= max\{\mathcal{Y}_{\mathcal{A}}, \mathcal{Y}_{\mathcal{D}}, \mathcal{Y}_{\mathcal{AD}}, \mathcal{Y}_{\mathcal{DA}}\}, \\ \mathcal{Z} &= max\{\mathcal{Z}_{\mathcal{A}}, \mathcal{Z}_{\mathcal{D}}, \mathcal{Z}_{\mathcal{AD}}, \mathcal{Z}_{\mathcal{DA}}\}. \end{aligned} \tag{3.7}$$

In practice, we can also use the weighted results of the four strategies as the final solution. Since it would bring several weight parameters for these strategies, we calculate the results of ELP according to Equation 3.7 for simplicity in the experiments.

3.4 Experiments

In this section, we conduct extensive experiments to evaluate the proposed ELP framework. After introducing the datasets and the experimental settings, we compare different baseline methods.

3.4.1 Data Processing

We collect a large set of click-through data (denoted as a system set) over a continuous period of time from a real-world search engine. Then a small number of click-through data are sampled from the system set and denoted as a gold set. We manually labeled entities from queries of the gold set with correct types. The labeled data are only used for seed selections

TABLE VIII

Statistics of the collected query data.

Dataset	#Queries	#Clicked URLs
Gold	16,903	2,369,618
System	217,223,831	1,556,499,551

and performance evaluations in the experiments. The basis statistics of these two datasets are shown in Table VIII.

In the experiments, we focus on 3 target types of classes, namely Local, Media and Person. By following the extraction rules in Section 3.3.1, we get entities and related auxiliary signals belonging to these 3 target types. In order to build a compact and reliable graph, we apply a threshold to get rid of some infrequent nodes. For example, we set the threshold as 1 for the gold dataset and filter out those nodes appearing only 1 time. The basic statistics of nodes and links in the entity-auxiliary graphs are represented in Table IX. "EC Links" means the entity-context links and "EU Links" means the entity-URL links. The distributions of the 3 target labels for entities are shown in Table X.

3.4.2 Compared Methods

In order to show that the LP model fits the constructed graph very well, we compare LP with several traditional classification methods. Given the bipartite EA graph, we consider the connected auxiliary nodes as features for each entity and the frequencies (the edge weights) as feature values. We focus on the following methods:

TABLE IX

Statistics of the entity-auxiliary graphs.

		#Auxiliary nodes		#Links	
Dataset	#Entities	#Contexts	#URLs	#EC Links	#EU Links
Gold	934	1,445	10,059	3,323	36,475
System	10,722	39,279	24,107	514,489	221,668

TABLE X

Distributions of labels for entities.

	Labels		
Dataset	Local	Media	Person
Gold	36.6%	36.4%	27.0%
System	33.5%	33.7%	32.8%

- SVM: **S**upport **V**ector **M**achine (SVM) is a popular classification method. Since non-linear RBF kernel is widely used in SVM models, we apply SVM (RBF) on the constructed bipartite EA graph.
- KNN: We compare with the **K**-**N**earest **N**eighbors method (KNN) to show the effectiveness of the LP method. We denote the KNN method using n neighbors as KNN-n.
- DT: The **D**ecision **T**ree method (DT) is applied on features extracted from the EA graph.
- NB: We apply the **N**aive **B**ayers (NB) on features of entities extracted from the built EA graph.

- LP: The original **L**abel **P**ropagation method (LP) is applied on the EA graph without the feature extraction.

In addition, in order to show the effectiveness of the proposed ELP framework, we compare with different variations of the LP model. Since both contextual words and clicked URLs can be considered as auxiliary information for entities in search query logs, we can construct three different bipartite graphs. They are the **E**ntity-**C**ontext (EC) graph, the **E**ntity-clicked **U**RL (EU) graph and the **E**ntity-**Auxiliary** (EA) graph. The EA graph considers both the contexts and clicked URLs as the auxiliary information in search query logs, so it contains more information than the EC and EU graphs. We can apply our proposed strategies on these different graphs and we summarize them as follows:

- LP: There are three versions for LP. They are LP (C), LP (U) and LP (A). LP (C) focuses on the EC graph. Similarly, LP (U) applies on the EU graph and LP (A) runs on the EA graph.
- LPA: We derive two baselines from LPA. The first one is LPA (C), which applies the LPA strategy on the bipartite EC graph. The other one is LPA (A) on the EA graph. The effectiveness of using more auxiliary information from search query logs can be demonstrated by comparing LPA (C) with LPA (A). Since the LPA strategy focuses on expanding the hidden connections between entities and contextual words, we cannot apply it on the EU graph.
- LPD: Three baselines can be generated from LPD. They are LPD (C), LPD (U) and LPD (A) on the EC, EU and EA graphs respectively. In particular, LPD (C) means that

we first get rid of the top k multi-type contextual words and then apply the LP method on the updated EC graph. LPD (U) is derived in a similar way. For LPD (A), we first find the top k multi-type contextual words from the EC graph and the top k multi-type clicked URLs from the EU graph. After that we delete these contextual words and clicked URLs from the EA graph and apply LP on the updated graph. In this way, we guarantee that the most ambiguous auxiliary nodes (both contextual words and clicked URLs) are removed from the EA graph.

- LPAD: There are two baselines based on LPAD. They are LPAD (C) and LPAD (A). Specifically, LPAD (C) contains three steps: (1) expand connections and update the EC graph; (2) delete multi-type nodes in the denser EC graph; (3) run LP on the latest EC graph. LPAD (A) executes in a similar way.
- LPDA: We generate LPDA (C) and LPDA (A) from LPDA. LPDA (C) updates the EC graph by first deleting multi-type contextual words and then expanding hidden connections between entities and the remaining contextual words. LPDA (A) updates the EA graph in a similar way.
- ELP: We also derive two versions for ELP. They are ELP (C) and ELP (A). ELP (C) combines LPA (C), LPD (C), LPAD (C) and LPDA (C) in an ensemble way while ELP (A) ensembles LPA (A), LPD (A), LPAD (A) and LPDA (A) together to achieve a better performance.

For a fair comparison, we use the same parameter settings for the baselines related to the LP method. Specifically, we test with different α values for LP and find that $\alpha \in (0.5, 0.9)$

yields similar good results. So we set the parameter α to be 0.75 as in (53). In order to get the similarities among contextual words, we use a pre-trained vectors[1] on about 100 billion words and phrases from various news articles. For the number of auxiliary nodes that should be deleted, we set it to be 10 in the experiments. In addition, we use SVM (RBF) with optimized parameters and other traditional classifiers with default parameters in our experiments. For each node, we can get a list of non-negative real numbers from LP indicating the posterior probabilities that the node relates to a label. We clamp these probabilities to 0/1 values for simplicity.

In order to evaluate the results, we focus on the labeled data and use accuracy and weighted average of the F1 score of each class (abbreviated as "weighted-F1") as the performance measures for entities. Weighted-F1 means that we calculate the F1 score for each label and find their average value weighted by the number of true instances for each label. This metric takes the label imbalance into consideration. For an entity with multi-labels, if the learned label matches with one of its multiple labels, we consider it as a correct prediction. Since we do not have ground truth for the auxiliary information, we will not present the quantitative analysis on the auxiliary information. We only show some qualitative analysis in Section 3.4.4. In the experiments, we randomly select a certain portion (e.g., 10%) of the entities as seeds for 10 times and report the average performances for models related to LP. We use the same seed entities as the training data for the traditional classification models.

---

[1]freebase-vectors-skipgram1000-en.bin.gz. It can be downloaded from `https://code.google.com/p/word2vec/`.

TABLE XI

Average performances on the EC graph of the gold dataset.

Metric	Method	Percentages of seed entities					
		1%	10%	20%	30%	40%	50%
Accuracy ↑	SVM (RBF)	0.3720 (3)	0.3876 (5)	0.3853 (5)	0.3763 (5)	0.4139 (5)	0.4206 (5)
	KNN-1	0.3610 (5)	0.4617 (4)	0.5100 (4)	0.5406 (4)	0.5591 (4)	0.5779 (4)
	DT	0.3688 (4)	0.4995 (3)	0.5568 (3)	0.5818 (3)	0.5989 (3)	0.6036 (3)
	NB	0.3752 (2)	0.5713 (2)	0.6106 (2)	0.6317 (2)	0.6271 (2)	0.6318 (2)
	LP	**0.4544** (1)	**0.6959** (1)	**0.7204** (1)	**0.7313** (1)	**0.7350** (1)	**0.7386** (1)
Weighted-F1 ↑	SVM (RBF)	0.2278 (3)	0.2445 (5)	0.2353 (5)	0.2206 (5)	0.2858 (5)	0.2918 (5)
	KNN-1	0.2248 (4)	0.4190 (4)	0.4987 (4)	0.5304 (4)	0.5522 (4)	0.5755 (4)
	DT	0.2231 (5)	0.4608 (3)	0.5404 (3)	0.5702 (3)	0.5903 (3)	0.5977 (3)
	NB	0.2789 (2)	0.5643 (2)	0.6066 (2)	0.6295 (2)	0.6233 (2)	0.6293 (2)
	LP	**0.3904** (1)	**0.6910** (1)	**0.7188** (1)	**0.7308** (1)	**0.7346** (1)	**0.7385** (1)

3.4.3 Performance Evaluation

In this subsection, we show the performances of the proposed ELP framework. We first demonstrate that how the LP method takes advantage of the constructed bipartite EA graph compared with some traditional classification models. Due to space limit, we only show the performances on the EC graph of the gold dataset in Table XI. The results are reported as "average performance + (rank)". "↑" indicates that the larger the value the better the performance. Similar performances can be obtained for other graphs.

It can be observed from Table XI that LP consistently outperforms other classification methods on accuracy and weighted-F1 scores for different amounts of seed entities (training data). It illustrates that the constructed graph helps the LP method propagate the label

information out very well. Since all the other classifiers ignore the graph structure, important information may be missing and the performances are not so well compared with the LP method that takes advantage of the graph structure. In addition, when the amount of seed entities increases, the performances become better for almost all classifiers except the SVM method with the RBF kernel. It seems that more training data does not help SVM (RBF) very much. However, in reality, more seed entities means more annotations and human labelings. With large volumes of new queries, extracting such supervised information from search query logs can be very expensive and time consuming. In Table XI, LP can only achieve 45% of accuracy when 1% of data are selected as seeds. The performance should be improved if we explore the hidden connections and get rid of multi-type nodes in the constructed graph as in the proposed ELP framework. So in the following, we focus on the gold dataset with 1% of seed entities to show the effectiveness of ELP.

The results of different methods based on LP are shown in Table XII. It can be observed that ELP (A) outperforms other baseline methods on both accuracy and weighted-F1 and ELP (C) can also achieve a very good performance. ELP (A) outperforms ELP (C) with an improvement of 21% on the accuracy. It shows that more auxiliary nodes help ELP achieve a better performance on learning entity types from search query logs.

In particular, due to the noisy information in search query logs, directly applying the LP method on the constructed EA graph may reduce the performance as shown in Table XII. However, the results can be improved if we better build the graph as introduced in LPA and LPD. We can observe that the performance of LPA (LPD) on the EA graph are better than

those on the EC and EU graphs. It demonstrates that using more high-quality auxiliary nodes can provide more important information and facilitate the process of LPA or LPD. Moreover, compared with the original LP method, both LPA and LPD can improve the performances for all the constructed bipartite graphs (i.e., EC, EU and EA). For example, LPA (A) significantly outperforms LP (A) with improvements of 45% and 104% on accuracy and weighted-F1, respectively. Furthermore, LPA seems more powerful than LPD on both the EC and EA graphs. It shows that exploring hidden connections among the sparse graph plays a more important role in learning entity types from search query logs.

Though LPA and LPD perform better than LP, our proposed ELP framework achieves better results than the LPA and LPD strategies. Specifically, ELP (A) outperforms LPA (A) with an improvement of 22% on the weighted-F1 score. In addition, ELP also performs better than LPAD and LPDA with an average improvement of 18% on the weighted-F1 score as shown in Table XII. It implies that ELP can maximize the effectiveness of combining different strategies together. Simply combining LPA and LPD together (e.g., LPAD and LPDA) may not make full use of the operations of adding more connections and deleting the noisy nodes.

In summary, with the help of exploring hidden connections and getting rid of noisy nodes, the proposed ELP framework can achieve an accuracy of 68% on the EA graph with only 1% of entities as seeds. From Table XI, we can see that the LP method needs around 10% of seed entities to get the same accuracy score. Therefore, ELP can help significantly reduce the cost of human labeling in learning entity types from search query logs.

TABLE XII

Average performances on the gold dataset.

Graph	Method	Metric	
		Accuracy ↑	Weighted-F1 ↑
EC	LP (C)	0.45 (8)	0.39 (8)
	LPA (C)	0.53 (5)	0.49 (5)
	LPD (C)	0.50 (7)	0.45 (6)
	LPAD (C)	0.53 (5)	0.49 (5)
	LPDA (C)	0.54 (4)	0.50 (4)
	ELP (C)	0.56 (3)	0.56 (2)
EU	LP (U)	0.39 (10)	0.26 (10)
	LPD (U)	0.52 (6)	0.44 (7)
EA	LP (A)	0.40 (9)	0.27 (9)
	LPA (A)	0.58 (2)	0.55 (3)
	LPD (A)	0.53 (5)	0.45 (6)
	LPAD (A)	0.58 (2)	0.55 (3)
	LPDA (A)	0.58 (2)	0.55 (3)
	ELP (A)	**0.68** (1)	**0.67** (1)

We further show the effectiveness of the proposed ELP framework on the larger system set. Only 0.1% of seed entities are used in the experiments to test the power of ELP. Since we only labeled entities in the gold set and these entities are included in the system set, we calculate the accuracy and weighted-F1 scores on the labeled entities in the system set. The performances are presented in Table XIII. We can get similar observations for the system set.

TABLE XIII

Average performances on the system dataset.

Graph	Method	Metric	
		Accuracy ↑	Weighted-F1 ↑
EC	LP (C)	0.44 (6)	0.32 (8)
	LPA (C)	0.45 (5)	0.37 (5)
	LPD (C)	0.50 (4)	0.41 (4)
	LPAD (C)	0.45 (5)	0.37 (5)
	LPDA (C)	0.45 (5)	0.37 (5)
	ELP (C)	0.57 (2)	0.56 (2)
EU	LP (U)	0.40 (9)	0.27 (10)
	LPD (U)	0.43 (7)	0.34 (7)
EA	LP (A)	0.42 (8)	0.30 (9)
	LPA (A)	0.55 (3)	0.51 (3)
	LPD (A)	0.44 (6)	0.36 (6)
	LPAD (A)	0.55 (3)	0.51 (3)
	LPDA (A)	0.55 (3)	0.51 (3)
	ELP (A)	**0.59** (1)	**0.58** (1)

3.4.4 Case Study

In this subsection, we present several case studies to show the effectiveness of the proposed ELP framework. We first show the most popular auxiliary nodes with their labels learned from ELP and explain how such auxiliary information can help detect new entities from search query logs. Then we give some examples of the hidden connections we explored in LPA. After that, we list several multi-type auxiliary nodes discovered in LPD. At the end, we will analyze the potential to disambiguate multi-label entities in the proposed ELP framework.

TABLE XIV

The most popular auxiliary information on the gold dataset.

Label	Top k	Auxiliary Information	
		Contextual Word	Clicked URL
Local	1	high school	hamptoninn3.hilton.com/en/
	2	Sale in	www.homes.com/Real_Estate
	3	IL	www.wunderground.com/weather-forcast/
	4	Orlando	www.accuweather.com/en/
	5	Coupons	www.city-data.com/city/
Media	1	Watch	www.tv.com/shows/
	2	Cast	tv.yahoo.com/shows/
	3	Season	tv.msn.com/tv/
	4	Songs	tv.yahoo.com/news/
	5	Episode	yidio.com/show/
Person	1	Biography	www.theguardian.com/film/
	2	Site	marquee.blogs.cnn.com/2014/
	3	Naked	images.fanpop.com/images/
	4	Nude	movies.msn.com/movies/
	5	Divorce	www.tmz.com/2014/

3.4.4.1 Popular Auxiliary Information

Given the gold dataset, we first randomly select 1% of entities as seeds and run ELP on the EA graph. Then we apply a threshold (e.g., larger than 10) to select the most popular contextual words and clicked URLs separately. After that, we group the auxiliary nodes according to their learned types and rank nodes in each group by the learned probability value in a decreasing order. Due to space limit, we only show the top 5 related auxiliary information learned from ELP (A) for the gold dataset in Table XIV. We can observe that people care about the education, real estate and weather very much when they search about local entities.

3.4.4.2 New Entity Discovery

With the learned types of auxiliary nodes in Table XIV, we can discover new entities easily. For example, if a new TV series is released, we can detect it as a new entity when people search

with the word "episode". In our experiments, we consider those entities appearing few times (≤ 2) as new entities and ELP can learn their types correctly. For instance, "Pogo" (an online game) appears only twice and ELP detects it as a media entity because its connected contextual words are "app" and "ipad". However, "Pogo" refers to a musical artist and a comic strip in Wikipedia. Therefore, the ELP method helps us discover "Pogo" as a new media entity, and we can add such information to the current knowledge graph.

3.4.4.3 Hidden Connections

Now we analyze how the hidden connections we explored help LPA fully propagate labels. We focus on those entities with few connected contextual words and give some examples of the entities with their hidden connections we discovered from the gold dataset as shown in Table XV. It can be observed that the hidden connections provide complementary and discriminative information for learning entity types. For example, given the entity "Big Brother" and its connected contextual word "CBS", the word "showtime" help predict "Big Brother" as an entity of media with more confidence. In addition, the hidden connections can help learn entity types more correctly. Take the entity "George Clooney" as an instance. The connected contexts "movie" and "new" are a bit ambiguous and "George Clooney" may be considered as an entity of media because "movie" is more related to media. However, if we connect "George Clooney" with "wedding", we can easily learn that "George Clooney" is an entity of person. Therefore, exploring the hidden connections in query logs can help learn entity types more accurately.

TABLE XV

Some entities and their connected contextual words from query logs.

		Contextual Word	
Label	Entity	Existing word	Hidden word
Media	Big Brother	CBS	Showtime
Person	George Clooney	Movie, New	Wedding
Media	Haunted	Taylor Swift	Music, Video
Person	Sarah McLachlan	Song	Single
Local	Starbucks	free, coffee	open, free shipping, zip code

3.4.4.4 Multi-type Auxiliary Information

Here we analyze the effectiveness of discovering and removing the multi-type auxiliary information from LPD. We list the 10 most ambiguous auxiliary nodes discovered from the gold dataset in Table XVI. It can be observed that these auxiliary nodes are related to different types of entities and getting rid of them can help propagate labels more accurately in LP. For example, the contextual word "Online" can refer to the online information of a place, a movie and a person. In addition, the clicked URL "en.wikipedia.org/wiki/" is related to a navigational website that contains diverse information. It is difficult to detect the type of an entity if such URLs are connected with the target entity. Therefore, we identify the ambiguous auxiliary nodes and remove them from our constructed graph.

3.4.4.5 Entity Disambiguation

According to the proposed ELP framework, we can get a list of non-negative real numbers indicating the probabilities that an entity relates to a type. We clamp these probabilities to 0/1 values for simplicity in the performance evaluation. However, in reality, a lot of entities

TABLE XVI

The multi-type auxiliary information discovered on the gold dataset.

	Auxiliary Information	
Top k	Contextual Word	Clicked URL
1	Online	en.wikipedia.org/wiki/
2	Lyrics	geo.yahoo.com/t/
3	News	video.search.yahoo.com/video/
4	Free	video.search.yahoo.com/search/
5	Hotels	images.search.yahoo.com/search/
6	Movie	search.yahoo.com/
7	Newspaper	news.search.yahoo.com/
8	Weather	www.imdb.com/title/
9	Map	www.youtube.com/
10	Jobs	news.yahoo.com/photos/

have more than one type. In this subsection, we analyze the potential of ELP to disambiguate multi-label entities.

We first analyze those entities without ambiguity. From our experimental results, ELP gives high probability values to the types of these entities. For example, given the entity "Gold Digger", ELP learns a probability value of 0.94 for the media type. Similarly, "'Walt Disney World" has a probability value of 0.80 for the local type.

We also focus on those multi-type entities to see whether it is easy to disambiguate them in our experiments. We take the entity "Maxwell" as an example. ELP learns it as a local entity with a probability of 0.48 and a person entity with a probability of 0.37. Table XVII shows some connected auxiliary nodes for "Maxwell". We group them according to their labels learned from ELP and rank them by their probability values in a decreasing order. It can be observed that the auxiliary nodes for "Maxwell" as a local entity and a person entity are very

TABLE XVII

Some auxiliary nodes for a multi-label entity "Maxwell".

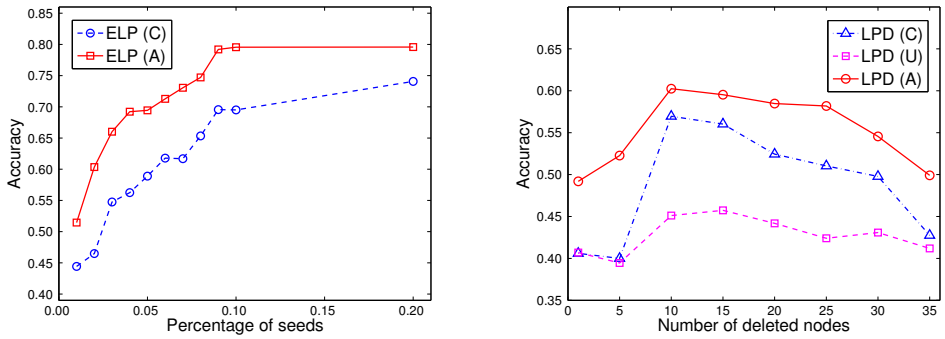
Label	Top k	Auxiliary Information	
		Contextual Word	Clicked URL
Local	1	Real estate	www.zillow.com/homedetails/
	2	Weather forecast	www.realtor.com/realestateandhomes/
	3	Map of	www.healthgrades.com/physician/
	4	Sale in	www.homes.com/Real_Estate/
	5	Homes for sale	www.wunderground.com/weather-forecast/
Person	1	New album	www.allmusic.com/artist/
	2	Discography	www.jango.com/music/
	3	Songs by	www.oldies.com/artist-songs/
	4	Albums	www.songkick.com/artists/
	5	Full album	www.mtv.com/artists/

different. Hence, ELP provides the potential for us to further split the entity node "Maxwell" into two nodes to better build the entity-auxiliary graph from search query logs.

3.4.5 Sensitivity Analysis

In this subsection, we assess the benefit of ELP with different amounts of seed entities. We focus on the EC and EA graphs extracted from the gold dataset and fix other parameters. Figure 8 (a) shows the classification accuracies. We find that the performances become better when we increase the seed entities. Moreover, the results stabilize when we use more than 10% of seed entities.

We also demonstrate the effect of the numbers of deleted nodes in Figure 8 (b). Here we fix 1% of seed entities and other parameters, but vary the numbers of deleted nodes. It can be observed that the best accuracy is achieved when we remove 10 multi-type auxiliary nodes. The noise may still exist if we get rid of too few nodes and the performance cannot be improved



(a) Effect of seed entites (b) Effect of deleted nodes

Figure 8. Parameter analysis on the gold dataset.

dramatically. However, information may be imcomplete if we delete too many nodes as shown in Figure 8 (b). So in our experiments, we set the number of deleted nodes as 10.

3.5 Related work

Entity extraction and classification have rapidly developed over the past few years (48; 61; 46). Several methods are proposed to extract entities from web documents (40; 42) and the disambiguation of entities from news articles is studied in (43; 44). In recent years, the extraction and classification of entities over query logs receive a lot of attention (62; 46; 45; 48; 49) and disambiguating entities in queries is also investigated in (47). However, all these existing methods do not fully explore the importance of the auxiliary information related to entities. Our study is different since we encode entities and the important auxiliary information together into a bipartite graph and learn the types of both entities and auxiliary nodes simultaneously.

The graph-based label propagation method is also very popular in information retrieval tasks (53; 54). Li et al. use click graphs, a bipartite-graph representation of click-through data

from search query logs, to improve query intent classifiers (53). Spam webpages are detected using the link structure of the click-through bipartite graph in (54). Given a few seed pages and websites, it propagates spam scores between queries and URLs. In our work, we focus on learning entity types from search query logs, which differs from the task in (53; 54). In addition, these models do not consider the sparsity and noise issues in search query logs. Our work proposes two separate strategies to explore hidden connections in the constructed bipartite graph and detect noisy information in query logs.

Besides the label propagation, many other learning methods, including Markov random walks (63), learning with local and global consistency (51; 64) and manifold regularization (65), are based on graphs. Furthermore, Chang et al. (66) proposed an unsupervised embedding scheme on graphs with heterogeneous components. Their method systematically captures network similarity between pairwise nodes by a deep learning framework. Though the optimization objectives are different in these models, the underlying assumption is shared among all the models, i.e., the conditional distributions of two samples will be similar if they are close in the intrinsic geometry of an input space.

The entity-oriented analysis of query data is also related to our work (67; 68). For example, class attributes are extracted from search query logs for entities in (69; 70) as a complement source for existing knowledge bases. Based on the attributes, synonymous query intent templates are identified in (71). In addition, entity-related search actions, annotations, and recommendation systems are studied in (39; 72; 73), respectively. However, our work is different from them since we study the problem of detecting entity types from search query logs.

CHAPTER 4

DETECTING EMERGING RELATIONS FROM NEWS

(This chapter was previously published as "HEER: Heterogeneous Graph Embedding for Emerging Relation Detection from News", in *Proceedings of 2016 IEEE International Conference on Big Data (BigData'16)* (3). DOI: https://doi.org/10.1109/BigData.2016.7840673.)

4.1 Introduction

A *relation* is about the connection between two entities. Entities can be persons, organizations, locations, etc., and examples of relations can be person-affiliation and organization-location. Recognizing relations between entities is required in a lot of real-world applications in information extraction, natural language understanding and information retrieval. Hence, extracting relations from unstructured texts, such as newswire, blogs, and so on have received considerable attention in the last few years (74; 75; 76; 77).

Conventional approaches of relation extraction from texts focus on a local view, where each sentence mentioning two entities is considered for feature learning. For example, the works in (74; 75; 76) extracted a lot of sentence-level features including lexical part-of-speech tags of words, syntactical dependency tree paths, etc.. In order to achieve good performance, such local view based methods require large amounts of sentences to extract useful sentence-level features. For those relations with few sentences, these methods would be problematic.

Some other works attempt to leverage knowledge graphs (KGs), such as Freebase[1] and DBpedia[2], to provide useful supervisions for extracting relations from texts. For instance, (75) uses relation instances in Freebase instead of annotated texts as their source of supervision. However, these methods are handicapped due to the limited coverage of existing KGs (78). As shown in (79), 71% of the roughly 3 million people do not have place of birth in Freebase, 94% do not have parents, and 99% do not have ethnicity. Therefore, a lot of research work tries to fill in the missing relations to mitigate the problem of knowledge sparsity (80; 81; 82; 83; 84; 85; 86). For example, Path Ranking Algorithm (PRA) (87; 88) performs link prediction in KGs via a random walk inference technique; embedded representations of entities and relations in KGs are learned to infer missing relations in (89; 90; 91).

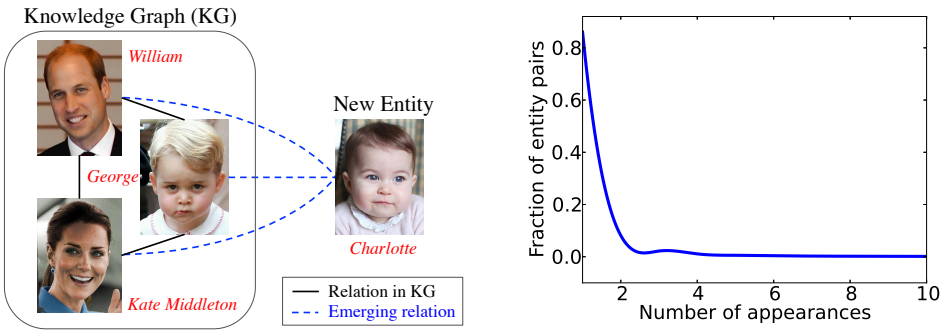
Nowadays, real-world knowledge is growing rapidly. New entities arise with time (92), resulting in large volumes of relations that do not exist in current KGs. We call such relations *emerging relations*[3]. Emerging relations often appear in news, and hence the latest information about new entities and relations can be learned from news timely. For example, when a new baby (e.g., Charlotte in Figure 9(a)) is born in the Royal Family, no information about this baby exists in KGs. However, there are lots of news talking about this new baby and her family.

---

[1] `https://www.freebase.com/`

[2] `http://wiki.dbpedia.org/`

[3] The relations with both entities in KGs are out of scope of this chapter since they can be inferred via the existing KG completion methods.



(a) Real-world examples (b) Appearance distributions

Figure 9. Examples and distributions of emerging relations.

Therefore, the relation between the new baby and her parent is an emerging relation and it can be detected from news.

In this chapter, we study the problem of discovering emerging relations from news. Detecting such relations has many benefits to real-world applications. Emerging relations can help expand current KGs and keep them up to date. In addition, emerging relations can also help news related tasks, such as news retrieval and ranking, event detection, etc.. However, detecting emerging relations is a challenging task due to the following reasons:

- Sentence-level features for emerging relations are usually rare. A data analysis conducted on 6 million online news headlines from Yahoo!, as shown in Figure 9(b), reveals that 86% of emerging entity pairs appear in only one sentence. Simply relying on sentence-level features extracted from few sentences could lead to sub-optimal results for emerging relation detection.

- Due to the lack of negative relations in KGs, previous methods (87; 83; 75) often apply different strategies to extract negative relations. However, the negative relations could be false negative (75) in reality, which may introduce noise and cause degraded performance for emerging relation detection.
- With massive amounts of news arriving every second, new relations emerge rapidly. It is necessary to keep KGs up to date with the latest emerging relations.

In order to address these issues, we start from a global graph perspective instead of the traditional local sentence perspective and propose a novel **H**eterogeneous graph **E**mbedding framework for **E**merging **R**elation detection (HEER). Figure 10 shows the simplified procedure of HEER with an example. The entities are in red. The co-occurrence links in the heterogeneous textual graph are in green and the relations in KG are in black. To capture the global corpus information in news, HEER constructs a heterogeneous textual graph from news. Two kinds of nodes – entities and contextual words – are involved in the graph and the link between two nodes represents their co-occurrence statistics from the whole news corpus. By jointly learning from the heterogeneous textual graph and the knowledge graph, HEER can embed words and entities into a low dimensional space. These graph-based embeddings not only preserve the semantic relatedness of entities and contextual words, but also provide a powerful prediction for the detection of emerging relations. To deal with the lack of negative relations in reality, HEER further predicts the emerging relations via a positive and unlabeled learning (PU) classifier (93) on the embeddings of entities.

In summary, our contributions are as follows:

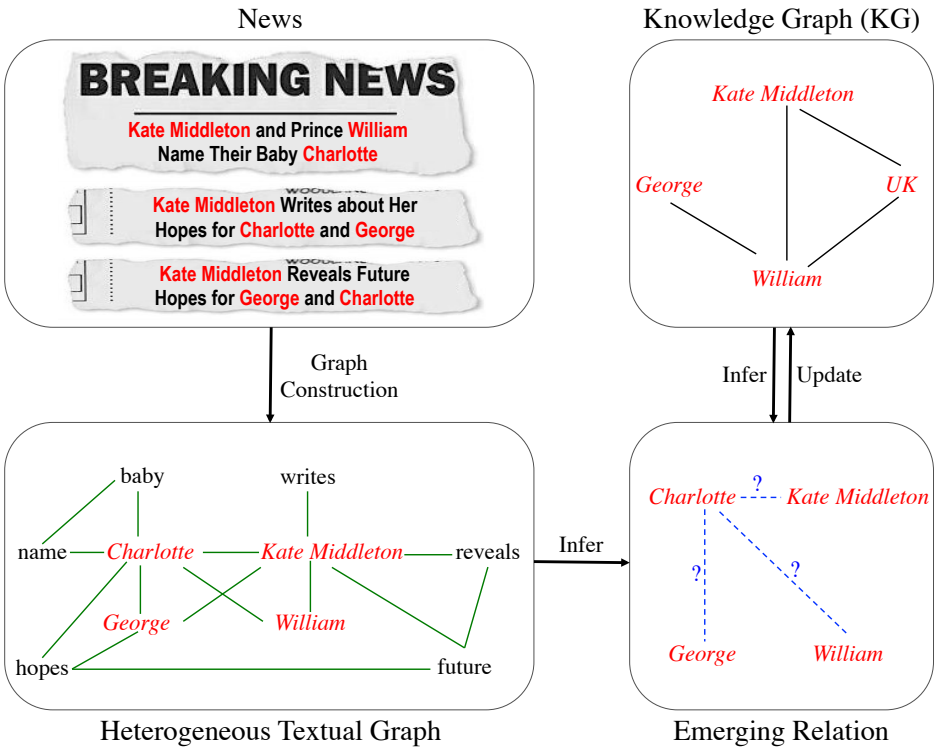


Figure 10. Detecting emerging relations from the news and the KG.

- We define *emerging relations* and propose a HEER framework to detect emerging relations by utilizing information from both news and KGs.
- We learn a classifier based on positive and unlabeled instances in the proposed HEER method by taking advantage of existing relations in KGs.
- We further implement HEER in an incremental manner to timely update KGs with the latest detected emerging relations.
- We conduct extensive empirical studies on real-world news data to demonstrate the effectiveness of the proposed HEER method.

4.2 Preliminary

In this chapter, we study the problem of discovering emerging relations from news. Before proceeding, we introduce the related concepts and notations.

Definition 8 ***Entity and Relation:*** *An entity e_i can represent a person, an organization, or a location, etc.. We use $y \in \{0, 1\}$ to denote the binary relation label for an entity pair (e_i, e_j). If two entities e_i and e_j have a relation in reality, such as a person-affiliation and an organization-location relation, $y = 1$. Otherwise, $y = 0$.*

Definition 9 ***Knowledge Graph (KG):*** *A knowledge graph is denoted as an undirected graph $\mathcal{G}_{kg} = (E_{kg}, \mathcal{E}_{kg})$, which keeps the known relations between entity pairs. For an entity pair (e_i, e_j), we use a KG label z to show whether the entity pair can match with a relation in the given KG or not, i.e., $z = 1$ if $(e_i, e_j) \in \mathcal{E}_{kg}$. Otherwise, $z = 0$.*

Because of the limited coverage of the existing KG (79), an entity pair with a KG label of $z = 0$ does not mean there is no relation for this pair (i.e., $y = 0$). Since the exact relation labels for entity pairs without KG labels are unknown to us, we call them *unlabeled relations*.

Nowadays, due to the rapid growth of real-world knowledge, large volumes of *emerging relations* are arising with time. An emerging relation is defined as follows:

Definition 10 ***Emerging Relation:*** *An emerging relation between an entity pair (e_i, e_j) exists, if its relation label $y = 1$ and it contains at least one entity that is not included in the given KG (i.e., $e_i \notin E_{kg}$ or $e_j \notin E_{kg}$).*

For example, in Figure 9(a), (*Charlotte, Kate Middleton*) is an emerging relation since *Charlotte* is a new entity and she is a child of *Kate Middleton*. Similarly, (*Charlotte, William*) and (*Charlotte, George*) are also examples of emerging relations. For an emerging relation, its KG label z always equals to 0 because at least one entity is not included in the KG.

Our goal is to learn the relation labels for those emerging entity pairs when the news just start talking about them. With rare sentences about emerging entity pairs, the traditional sentence-based methods could lead to sub-optimal results for emerging relation detection. Compared to traditional local sentence based relation detection, we construct a heterogeneous textual graph from news to capture the global corpus information in news.

Definition 11 ***Heterogeneous Textual Graph:*** *A heterogeneous textual graph is represented as an undirected graph* $\mathcal{G}_{news} = (\mathcal{V}_{news}, \mathcal{E}_{news})$. $\mathcal{V}_{news}$ *is the set of nodes (objects), including two types of objects, i.e., entities* $E_{news} = \{e_1, ..., e_M\}$ *and contextual words* $C_{news} = \{c_1, ..., c_N\}$. $\mathcal{E}_{news} \subseteq \mathcal{V}_{news} \times \mathcal{V}_{news}$ *is the set of links (edges) between the nodes in* $\mathcal{V}_{news}$*, which involves the links of entity-entity, entity-word, and word-word co-occurrences.*

An example of the heterogeneous textual graph is shown in Figure 10. Each link in the graph represents the co-occurrence of two nodes in news sentences and its weight equals to the frequencies of co-occurrences of these two nodes. For instance, the link between *Charlotte* and *baby* shows that these two nodes co-appear in some news sentence and the weight of this link is 1 since these two nodes co-appear in the first news only.

Such a heterogeneous textual graph helps encode the global corpus information in news. Besides, the existing KG provides helpful guidance for learning relations between entity pairs.

We can utilize the heterogeneous textual graph and the current KG together for detecting emerging relations. However, it is challenging since entities associated with emerging relations are missing in current KGs. In addition, no negative relations exist in KGs, creating difficulties in utilizing only positive and unlabeled instances.

In order to address these challenges, we propose a novel **H**eterogeneous graph **E**mbedding framework for **E**merging **R**elation detection (HEER).

4.3 Proposed Method

In this section, we first introduce how HEER constructs a heterogeneous textual graph from news. Then we describe how HEER can jointly learn from the heterogeneous textual graph and the existing KG to embed every entity and contextual word into a low dimensional space. After that, we present the learning classifier with only positive and unlabeled relations. Furthermore, we discuss how to implement HEER in an incremental manner to timely update the KG with the latest emerging relations.

4.3.1 Constructing a Heterogeneous Textual Graph from News

Given a large collection of news $\mathcal{D}$, the proposed HEER method first extracts entities and contextual words to build the heterogeneous textual graph. In this chapter, entities in news are annotated with the Stanford Named Entity Recognizer (NER) tool[1]. We mainly focus on 3 types of entities, namely person, location and organization, and consider the public available DBpedia dataset as the given knowledge graph. The entities that cannot be exactly matched to

[1] `http://nlp.stanford.edu/software/CRF-NER.shtml`

DBpedia are viewed as new entities. Excluding entities, the remaining uni-gram words in news are considered as contextual words and we remove stop words beforehand. Each entity and each uni-gram contextual word are nodes in the constructed heterogeneous textual graph. In order to extract the co-occurrence links in the graph, nodes within every 5-word sliding window in a news sentence are considered to be co-occurring with each other as in (57). We use the frequencies of nodes co-appearing in news sentences as weights of corresponding links.

4.3.2 Joint Embedding of the News and the KG

Given the constructed heterogeneous textual graph and the KG, we aim to learn a low dimensional space for every entity and contextual word. The learned embeddings should not only fit the relation information in the KG, but also reflect the text descriptions of emerging relations in news. In order to achieve this, we should jointly embed the heterogeneous textual graph and the KG. In the following, we first explain how to learn graph embeddings from a single graph, and then present how to jointly embed multiple graphs.

According to the types of links, the heterogeneous textual graph $\mathcal{G}_{news}$ can be split into three sub-graphs: the homogeneous entity-entity sub-graph $\mathcal{G}_{ee} = (E_{news}, \mathcal{E}_{ee})$, the bipartite entity-word sub-graph $\mathcal{G}_{ec} = (\mathcal{V}_{news}, \mathcal{E}_{ec})$ and the homogeneous word-word sub-graph $\mathcal{G}_{cc} = (C_{news}, \mathcal{E}_{cc})$. These three sub-graphs together capture the global corpus information in news. Given the bipartite entity-word sub-graph $\mathcal{G}_{ec} = (\mathcal{V}_{news}, \mathcal{E}_{ec})$, for instance, we aim to embed each entity $e_i \in E_{news}$ and each word $c_j \in C_{news}$ into low-dimensional vectors $\mathbf{s}_i \in \mathbb{R}^d$ and $\mathbf{t}_j \in \mathbb{R}^d$. Here d is the dimension of embedding vectors and $d \ll |\mathcal{V}_{news}|$.

In order to learn the embeddings, for each co-occurrence link $(e_i, c_j) \in \mathcal{E}_{ec}$, we first define the conditional probability of e_i given c_j as

$$P(e_i|c_j) = \frac{e^{\mathbf{s}_i^\top \mathbf{t}_j}}{\sum_{k=1}^{|E_{news}|} e^{\mathbf{s}_k^\top \mathbf{t}_j}}. \tag{4.1}$$

For each word c_j, this probability actually calculates a conditional distribution $P(\cdot|c_j)$ over all the entities in E_{news}. In the low-dimensional space, we intend to preserve the *second-order proximity*, which means two nodes are similar to each other if they have similar neighbors (94), by making $P(\cdot|c_j)$ be close to its empirical distribution $\hat{P}(\cdot|c_j)$. Here we define $\hat{P}(\cdot|c_j) = \frac{w_{j\cdot}}{o_j}$, where $w_{j\cdot}$ is the weight of the edge $(e_\cdot, c_j)$ and o_j is the sum of weights for edges connected to c_j, i.e., $o_j = \sum_{e_k \in N(c_j)} w_{jk}$, where $N(c_j)$ is the set of entity neighbors of c_j.

By minimizing the Kullback-Leibler (KL) divergence between two distributions $\hat{P}(\cdot|c_j)$ and $P(\cdot|c_j)$, we obtain the objective function for embedding the bipartite entity-word sub-graph $\mathcal{G}_{ec}$ as follows:

$$J_{ec} = -\sum\nolimits_{(e_i,c_j)\in\mathcal{E}_{ec}} w_{ij} \log P(e_i|c_j). \tag{4.2}$$

However, it is time-consuming to directly optimize Equation 4.2 since it requires to sum over the entire set of links when calculating the conditional probability $P(\cdot|c_j)$. In order to address this issue, we adopt the techniques of negative sampling (57), where for each edge selected with a probability proportional to its weight, multiple negative links (edges) are sampled from some noisy distribution. For the detailed optimization process, readers can refer to (57).

Since a homogeneous graph can be easily converted to a bipartite graph, we can derive similar objective functions for embedding the entity-entity sub-graph $\mathcal{G}_{ee}$ and the word-word sub-graph $\mathcal{G}_{cc}$ as follows:

$$J_{ee} = -\sum\nolimits_{(e_i,e_j)\in\mathcal{E}_{ee}} w_{ij} \log P(e_i|e_j), \tag{4.3}$$

$$J_{cc} = -\sum\nolimits_{(c_i,c_j)\in\mathcal{E}_{cc}} w_{ij} \log P(c_i|c_j). \tag{4.4}$$

With the objectives Equation 4.2, Equation 4.3 and Equation 4.4, we can learn vector representations of the heterogeneous textual graph:

$$J_{news} = J_{ec} + J_{ee} + J_{cc}. \tag{4.5}$$

Besides the heterogeneous textual graph, the current KG contains a large amount of positive relations between entities, providing helpful guidance for learning relations between entities. Since the KG is a homogeneous entity-entity graph $\mathcal{G}_{kg}$ about real-world relations, we can learn vector representations of the KG in a similar way:

$$J_{kg} = -\sum\nolimits_{(e_i,e_j)\in\mathcal{E}_{kg}} w_{ij} \log P(e_i|e_j), \tag{4.6}$$

where $\mathcal{E}_{kg}$ is the set of positive relations in the KG and we set the weight w_{ij} for each relation as 1. In order to learn from both the news and the KG, we combine them together as the final objective of the proposed HEER model:

$$J = \theta J_{kg} + (1 - \theta) J_{news}. \tag{4.7}$$

Here $\theta \in [0, 1]$ is a guiding parameter that trades off between news and the KG. Specifically, $\theta = 0$ (or 1) indicates that only the news (or the KG) is utilized in learning embeddings. In addition, a higher θ indicates that the KG plays a more important role in the process of embedding.

Since the links in different sub-graphs have different meanings in reality, we sample links from each sub-graph independently to optimize Equation 4.7. The detailed process of the graph embedding is summarized in Algorithm 3.

4.3.3 Detecting Emerging Relations with Positive Cases Only

After the graph embedding procedure, each entity and word can be represented with a d-dimensional vector. We use such global graph-based embeddings of entities as features for emerging relation detection. Due to the lack of negative relations in the KG, it is challenging to detect emerging relations with positive instances only. In the following, we present a positive and unlabeled (PU) learning classifier to address this issue.

Given an instance of an entity pair (e_i, e_j), we can represent the feature of the instance as $\mathbf{x} = h(\mathbf{s}_i, \mathbf{s}_j)$, where h is a function of the entity embedding vectors $\mathbf{s}_i$ and $\mathbf{s}_j$. Different

Algorithm 3 Joint embedding of the news and the KG

Input: The heterogeneous textual graph $\mathcal{G}_{news} = \mathcal{G}_{ec} \cup \mathcal{G}_{ee} \cup \mathcal{G}_{cc}$, the KG $\mathcal{G}_{kg}$, the guiding parameter θ, the number of negative samples k, and the number of embedding iterations T.
Output: Entity embeddings $\mathcal{S}$ and word embeddings $\mathcal{T}$.

```
Initialize entity embeddings S randomly
Initialize contextual word embeddings T randomly
while iter ≤ T do
    Generate a random number γ ∈ [0, 1]
    if γ ≤ θ then
        EMBEDDING UPDATE(S, S, G_kg, k)
    else
        EMBEDDING UPDATE(S, T, G_ec, k)
        EMBEDDING UPDATE(S, S, G_ee, k)
        EMBEDDING UPDATE(T, T, G_cc, k)
    end if
end while

function EMBEDDING UPDATE(S, T, G, k)
    Sample an edge from G and draw k negative edges
    Update node embeddings S and T
end function
```

formulations of h can be derived to represent the pair features, such as the concatenation, the average, etc.. In this chapter, we simply take $h(\mathbf{s}_i, \mathbf{s}_j) = \frac{1}{2}(\mathbf{s}_i + \mathbf{s}_j)$. Let $\mathbf{X} = \{\mathbf{x} : (e_i, e_j) \in \mathcal{E}_{ee}\}$ denote the feature representations of all the entity pairs co-occurring in the news. We consider $\mathbf{X}$ as the input feature matrix of the PU classifier for emerging relation detection.

As mentioned in Section 4.2, each entity pair has a KG label z showing whether an entity pair can match with a relation in KG (i.e., $z = 1$) or not (i.e., $z = 0$). For the positive entity pairs $\mathcal{P}$ with KG labels of $z = 1$, we denote their feature matrix as $\mathbf{X}(\mathcal{P})$. The feature matrix of the remaining unlabeled entity pairs $\mathcal{U}$ with KG labels of $z = 0$ are denoted as $\mathbf{X}(\mathcal{U})$. We further denote the emerging entity pairs as $\mathcal{L}$ and their feature matrix as $\mathbf{X}(\mathcal{L})$. Here $\mathcal{L}$ is a

subset of $\mathcal{U}$. We use $\mathbf{y}$ and $\mathbf{z}$ to denote the relation labels and KG labels for all the entity pairs in $\mathcal{P} \cup \mathcal{U}$. Our ultimate goal is to predict the relation labels $\mathbf{y}$ for emerging entity pairs $\mathcal{L}$ by learning from $\mathcal{P}$ and $\mathcal{U}$.

For entity pairs in $\mathcal{U}$, the relation labels $\mathbf{y}$ are unknown but the KG labels $\mathbf{z}$ are known. Hence, we propose to train a PU classifier f on $\mathcal{P} \cup \mathcal{U}$ to learn the relation labels $\mathbf{y}$ by inferring from the KG labels $\mathbf{z}$. We adopt the idea from (93) to adjust a classifier g on KG labels $\mathbf{z}$ to a classifier f on relation labels $\mathbf{y}$ with a constant factor.

We first train a standard classifier g on KG labels $\mathbf{z}$ that yields a function $g(\mathbf{x}) = P(z = 1|\mathbf{x})$ for each instance of an entity pair in $\mathcal{P} \cup \mathcal{U}$. Here $P(z = 1|\mathbf{x})$ is the probability that an instance with feature $\mathbf{x}$ has a positive KG label, i.e., $z = 1$. By assuming that entity pairs in $\mathcal{P}$ are chosen completely randomly from all real relations in $\mathcal{P} \cup \mathcal{U}$, we can show that $P(z = 1|\mathbf{x}, y = 1) = P(z = 1|y = 1)$. This is the classic "selected at random" assumption in (93) and it is proved that

$$P(y = 1|\mathbf{x}) = \frac{P(z = 1|\mathbf{x})}{P(z = 1|y = 1)}. \tag{4.8}$$

Equation 4.8 shows that we can predict the probability $P(y = 1|\mathbf{x})$ of the relation label by estimating $P(z = 1|y = 1)$, which is the probability that an entity pair with relation label $y = 1$ exists in the KG, i.e., $z = 1$. Here $P(z = 1|y = 1)$ can be effectively estimated by using the classifier g and a set of entity pairs S randomly sampled from $\mathcal{P} \cup \mathcal{U}$. Let S_p be the subset of

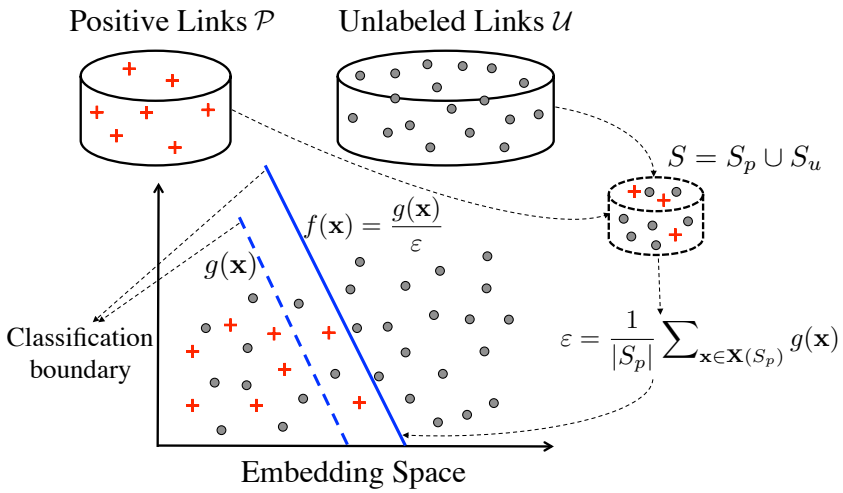


Figure 11. The PU learning classifier in HEER.

entity pairs in S with positive KG labels (i.e., $z = 1$) and $\mathbf{X}(S_p)$ is the corresponding feature set. We can obtain the following formula:

$$P(z = 1|y = 1) \sim \varepsilon = \frac{1}{|S_p|} \sum_{\mathbf{x} \in \mathbf{X}(S_p)} g(\mathbf{x}), \tag{4.9}$$

where the estimator ε is the average value of $g(\mathbf{x})$ for $\mathbf{x}$ in $\mathbf{X}(S_p)$. Since ε is based on a certain number of data instances, it has a low variance and is preferable in practice (93). With ε and the classifier g on KG labels $\mathbf{z}$, we can adjust to a classifier f on relation labels $\mathbf{y}$ as follows:

$$f(\mathbf{x}) = P(y = 1|\mathbf{x}) = \frac{g(\mathbf{x})}{\varepsilon}. \tag{4.10}$$

Algorithm 4 The HEER algorithm

Input: A set of news texts $\mathcal{D}$ and the KG $\mathcal{G}_{kg}$.
Output: A PU learning classifier f for emerging entity pairs $\mathcal{L}$.
//constructing graphs
Extract entities and contextual words from $\mathcal{D}$
Construct the heterogeneous textual graph $\mathcal{G}_{news}$
//joint embedding
Learn embeddings according to Algorithm 3
//detecting emerging relations
Learn the feature set $\mathbf{X}$ after embedding
Get positive and unlabeled relations $\mathcal{P}$ and $\mathcal{U}$
Train a classifier g on KG labels $\mathbf{z}$
Estimate ε from g
Learn f using g and ε on relation labels $\mathbf{y}$
Predict relations labels for emerging entity pairs $\mathcal{L}$

Figure 11 shows the learning process of the PU classifier on positive relations $\mathcal{P}$ and unlabeled relations $\mathcal{U}$. Since the classifier g considers all unlabeled relations as negative, an entity pair with a relation label $y = 1$ may be wrongly predicted as negative if it has a KG label $z = 0$. By adjusting g to f with an estimator ε, we can learn the correct relation label for an entity pair with a KG label of $z = 0$. After training, we can predict the relation labels for emerging entity pairs $\mathcal{L}$. By integrating the graph construction, the graph embedding, and the positive-unlabeled learning together, the proposed HEER framework can detect emerging relations from news effectively. We summarize the whole process of HEER in Algorithm 4.

4.3.4 Incremental Update of the KG

As news arrives rapidly with huge amounts of emerging relations, it is essential to timely update the KG with the latest emerging relations. In this section, we show how to implement HEER in an incremental manner for KG updates.

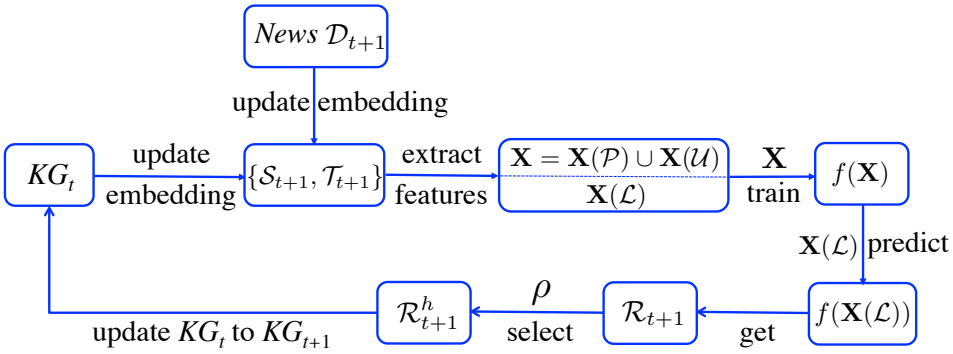


Figure 12. The update procedure of the incremental HEER.

TABLE XVIII

Statistics of the Yahoo! and BBC datasets.

	News	Heterogeneous textual graph					Knowledge Graph		Classification instances		
Dataset	$\lvert\mathcal{D}\rvert$	$\lvert E_{news}\rvert$	$\lvert C_{news}\rvert$	$\lvert\mathcal{E}_{ee}\rvert$	$\lvert\mathcal{E}_{ec}\rvert$	$\lvert\mathcal{E}_{cc}\rvert$	$\lvert E_{kg}\rvert$	$\lvert\mathcal{E}_{kg}\rvert$	$\lvert\mathcal{P}\rvert$	$\lvert\mathcal{U}_p\rvert$	$\lvert\mathcal{U}_n\rvert$
Yahoo!	6,209,256	13,801	61,705	20,136	398,466	697,804	22,157	710,994	3,297	9,246	12,543
BBC	44,088	2,556	7,273	873	19,206	57,373	2,030	43,689	167	575	742

Assume we collect news at regular time intervals, e.g., one day, one week or one month. All the news texts at current time t are $\mathcal{D}_t$ and the current KG is $\mathcal{G}^t_{kg}$. The proposed HEER can learn entity embeddings $\mathcal{S}_t$ and contextual word embeddings $\mathcal{T}_t$ from both $\mathcal{D}_t$ and $\mathcal{G}^t_{kg}$. We denote the embedding results as $\{\mathcal{S}_t, \mathcal{T}_t\}$. At time $t+1$, the news will be updated as $\mathcal{D}_{t+1}$, including $\mathcal{D}_t$ and the newly arrived news between time t and $t+1$. Note that there is no need to retrain the embeddings from scratch. Instead, we can reuse the previous trained embeddings $\{\mathcal{S}_t, \mathcal{T}_t\}$ at time t as initialization in Algorithm 3 to learn $\{\mathcal{S}_{t+1}, \mathcal{T}_{t+1}\}$ at time $t+1$.

With the updated embeddings, HEER can detect a set of emerging relations $\mathcal{R}_{t+1}$ from $\mathcal{D}_{t+1}$. In order to keep the KG up to date, we should add $\mathcal{R}_{t+1}$ into $\mathcal{G}_{kg}^{t}$. However, there are some false positive relations in $\mathcal{R}_{t+1}$. If we add all the detected relations into the KG, it will increase the noisy information and reduce the quality of the KG. Therefore, we select those highly-reliable emerging relations from $\mathcal{R}_{t+1}$ with a threshold ρ. Specifically, given the feature set $\mathbf{x}_{t+1}$ for a detected relation in $\mathcal{R}_{t+1}$, if the PU learning probability $f(\mathbf{x}_{t+1}) \geq \rho$, we will add the relation into $\mathcal{G}_{kg}^{t}$. Otherwise, we will discard it. We denote these highly-reliable emerging relations as $\mathcal{R}_{t+1}^{h}$. They can be further considered as positive instances during the PU learning process. With these highly-reliable emerging relations, we can get a new KG $\mathcal{G}_{kg}^{t+1}$. The update procedure of the incremental HEER is illustrated in Figure 12.

4.4 Experiments

In this section, we conduct extensive experiments to evaluate the proposed HEER framework. After introducing the datasets and the experimental settings, we compare different baseline methods.

4.4.1 Data Processing

Two real-world news datasets are used in the experiment.

• **Yahoo! News**[1]: We collect a large set of online English news from Yahoo! News in October 2015. Only the headline information is considered for Yahoo! News dataset.

---

[1] `https://www.dropbox.com/s/yad2tfaj9ve3vuf/yahoo_news_titles.tar.gz?dl=0`

• **BBC News**[1] (95): Documents in five topical areas are collected from the BBC news website from 2004 to 2005. We consider each sentence in the document as a piece of news.

We annotate entities and contextual words according to the method in Section 5.3.1. In order to find new entities and emerging relations, we map the entities to the knowledge graph DBpedia with exact match. The entities that cannot be matched are considered as new entities. To avoid the high-cost of human labeling, we focus on the existing relations in DBpedia to evaluate the effectiveness of the proposed HEER framework.

For all the entities E_{news} in the existing relations, we randomly select half of them as new entities. We denote these new entities as E_n and the remaining half as E_o. The entity pairs with KG labels of $z = 1$ and with both entities in E_o are regarded as positive instances, denoted as $\mathcal{P}$. The entity pairs with KG labels of $z = 1$ and with at least one entity in E_n are regarded as emerging relations. Since the emerging relations are unlabeled positive instances, we denote them as $\mathcal{U}_p$. The entity pairs with KG labels of $z = 0$ and with both entities in E_{news} are regarded as unlabeled negative instances, denoted as $\mathcal{U}_n$. Thus, the unlabeled instances, denoted as $\mathcal{U}$, are the union of positive and negative unlabeled instances, i.e. $\mathcal{U} = \mathcal{U}_p \cup \mathcal{U}_n$. The statistics of the two datasets are summarized in Table XVIII.

4.4.2 Compared Methods

In order to show that the HEER model can effectively detect emerging relations, we compare the following methods.

---

[1] `http://mlg.ucd.ie/files/datasets/bbc-fulltext.zip`

TABLE XIX

The classification performance on emerging relation detection task.
(a) Results on the Yahoo! News dataset.

Methods	Criteria		
	AUC ↑	Accuracy ↑	F1 ↑
BOW	0.562±0.006 (10)	0.510±0.001 (11)	0.118±0.002 (10)
LS	0.653±0.005 (4)	0.506±0.002 (12)	0.026±0.005 (12)
SG	0.547±0.006 (12)	0.543±0.004 (7)	0.188±0.014 (8)
DW	0.587±0.010 (8)	0.516±0.001 (10)	0.064±0.004 (11)
LINE	0.600±0.012 (6)	0.533±0.003 (9)	0.129±0.011 (9)
PTE	0.732±0.008 (3)	0.502±0.000 (13)	0.010±0.002 (13)
BOW-PU	0.559±0.006 (11)	0.541±0.007 (8)	0.567±0.011 (3)
LS-PU	0.647±0.002 (5)	0.610±0.002 (4)	0.481±0.011 (7)
SG-PU	0.545±0.007 (13)	0.610±0.007 (4)	0.517±0.008 (6)
DW-PU	0.587±0.009 (8)	0.577±0.006 (6)	0.545±0.008 (4)
LINE-PU	0.598±0.011 (7)	0.617±0.008 (3)	0.538±0.008 (5)
PTE-PU	0.734±0.007 (2)	0.675±0.006 (2)	0.671±0.006 (2)
HEER	0.786±0.007 (1)	0.717±0.008 (1)	0.716±0.005 (1)

(b) Results on the BBC News dataset.

Methods	Criteria		
	AUC ↑	Accuracy ↑	F1 ↑
BOW	0.552±0.028 (9)	0.496±0.013 (13)	0.053±0.034 (9)
LS	0.632±0.028 (4)	0.501±0.003 (11)	0.005±0.011 (12)
SG	0.571±0.009 (6)	0.520±0.005 (7)	0.127±0.021 (8)
DW	0.516±0.022 (13)	0.506±0.004 (9)	0.034±0.017 (10)
LINE	0.538±0.035 (10)	0.505±0.005 (10)	0.029±0.017 (11)
PTE	0.664±0.029 (2)	0.500±0.001 (12)	0.001±0.001 (13)
BOW-PU	0.560±0.023 (8)	0.544±0.019 (5)	0.531±0.045 (5)
LS-PU	0.624±0.027 (5)	0.606±0.022 (2)	0.468±0.051 (7)
SG-PU	0.571±0.019 (6)	0.598±0.017 (4)	0.553±0.026 (3)
DW-PU	0.520±0.026 (12)	0.517±0.027 (8)	0.520±0.023 (6)
LINE-PU	0.536±0.039 (11)	0.537±0.037 (6)	0.541±0.039 (4)
PTE-PU	0.656±0.017 (3)	0.601±0.014 (3)	0.639±0.016 (2)
HEER	0.712±0.033 (1)	0.644±0.035 (1)	0.649±0.024 (1)

TABLE XX

Examples of emerging relations from Yahoo! News.

Relation		News text
Elizabeth McGovern (out)	Downton Abbey (out)	1. Elizabeth McGovern: Broadway trip by Downton Abbey cast was like being a Beatle.
Alcatel-Lucent (out)	Michel Combes (in)	1. Alcatel-Lucent slashes payout to former chief Michel Combes.
Saudi Arabia (out)	Shaybah (out)	1. Saudi Arabia to invest US$45 billion in Shaybah oil field expansion.
Bernie Sanders (in)	University of Chicago (out)	1. Bernie Sanders speaks at the University of Chicago. 2. Bernie Sanders to speak at University of Chicago Monday. 3. Bernie Sanders calls on students to join his fight during University of Chicago stop.
Ghana (out)	Kwame Nkrumah (out)	1. Ghana celebrates Dr. Kwame Nkrumah today.
David Helfgott (in)	Melbourne (out)	1. David Helfgott, Australia's most well-known classical pianist, is coming to Melbourne.

- BOW: It is based on the traditional "bag-of-words" representation. Each entity pair is represented with a $|C_{news}|$-dimensional vector where the weight of each dimension is calculated by the number of times the word and the entity pair co-occur in news.
- LS: It is the standard **L**ocal **S**entence based classifier (LS) (75) using a variety of sentence-level features, including lexical part-of-speech tags of words and syntactical dependency tree paths.
- SG: It is based on the state-of-the-art word embedding model, **S**kip-**G**ram (SG) (57). It learns embedding vectors for each word in news, where each entity is consider as a word in this chapter.
- DW: It is based on the **D**eep**W**alk model (DW) (96). DW is only applicable for homogeneous graphs with binary edges. It learns embeddings of nodes by applying truncated random walks on the graph. By viewing entities and contextual words as one type of node, we can build a homogeneous graph from news and apply DW on this graph.

- LINE: It is based on the **L**arge-scale **I**nformation **N**etwork **E**mbedding method (LINE) (94). Similar to DW, LINE treats entities and contextual words as one type of nodes but considers the weights of the edges when learning the embeddings.
- PTE: It is based on the **P**redictive **T**ext **E**mbedding method (PTE) (97). It learns embeddings of nodes from the heterogeneous textual graph and the KG. The trade-off between news and the KG is not considered in PTE.

All the above baseline methods train a standard classifier by treating all the unlabeled instances as negative ones. These baselines can be used to show the effectiveness of the positive-unlabeled (PU) learning for the detection of emerging relations from news. In addition, the PU learning on different kinds of feature sets are compared:

- BOW-PU: It is the **BOW** based **PU** classifier (BOW-PU). We apply the PU learning on the BOW features.
- LS-PU: It is the **L**ocal **S**entence based **PU** classifier (LS-PU). We train a PU classifier on the local sentence features.
- SG-PU: It is the **S**kip-**G**ram based **PU** classifier (SG-PU). After SG gets the embeddings, the PU learning algorithm is applied.
- DW-PU: It is the **D**eep**W**alk based **PU** classifier (DW-PU). After DW obtains the embeddings, the PU learning algorithm is applied.
- LINE-PU: It is the **LINE** based **PU** classifier (LINE-PU). After LINE learns the embeddings, the PU learning algorithm is applied.

- PTE-PU: It is the **PTE** based **PU** classifier (PTE-PU). After PTE learns the embeddings, the PU learning algorithm is applied.
- HEER: It is the **H**eterogeneous graph **E**mbedding for **E**merging **R**elation detection (HEER) proposed in this chapter. We apply the PU learning after embedding the heterogeneous textual graph and the KG.

For fair comparisons, random forest classifier is used for all the above approaches and the number of trees in the forest is set as 100. The number of BOW features is 61,705 and 7,273 for Yahoo! News and BBC News, respectively. The number of sentence-level features is 286,461 and 21,481 for Yahoo! News and BBC News, respectively. For the embedding methods, the dimensionality of the embeddings is set to 50 and the average embedding of each entity pair is the input feature of the classifier. The guiding parameter θ of HEER is set to 0.2 for both datasets. Other settings of the graph embedding are the same as in (97; 94). When estimating ε in the PU learning, as in (93), 10% of instances are used for the estimation.

To evaluate the performance of the compared approaches, we randomly sample 80% of instances in $\mathcal{U}$ and keep all the instances in $\mathcal{P}$ as the training set, and use the remaining 20% of instances in $\mathcal{U}$ as the testing set. This random sampling experiment is repeated 5 times. We consider the AUC, accuracy and F1 score as the evaluation metrics.

The average performance with the rank of each method is reported in Table XXIII. It can be observed that PU learning methods perform much better than the standard positive-negative learning methods on the F1 metric. In addition, the proposed HEER consistently outperforms other PU baseline methods on both datasets.

We can find that the baselines without PU learning have similar accuracies but SG performs best on both datasets. It indicates the embedding methods can perform better than the traditional BOW and sentence-level features. However, since these baselines simply treat the unlabeled instances as negative ones, they perform worse than the PU learning methods for the task of emerging relation detection.

Among the baselines using the PU learning technique, we can observe that LS-PU performs worst on F1 although the feature size of LS-PU is the largest. The reason is that LS-PU cannot capture enough information for the classification task due to the sparsity of the sentence-level features. The recall of LS-PU is very low. However, the proposed HEER method captures global corpus information in news by building a heterogeneous textual graph. Therefore, with only 50 embedding features, HEER can have a much higher recall than LS-PU, thereby achieving a higher F1 score. Another discovery is that the homogeneous graph embedding models DW-PU and LINE-PU do not perform well on both datasets because they do not take the heterogeneity of news into account. In addition, the news and the KG are two heterogeneous data sources. If we simply combine them together as a homogeneous graph, DW-PU and LINE-PU will perform much worse though more information in the KG is considered. Hence we only show the better performing version of DW-PU and LINE-PU without using the KG in Table XXIII. Furthermore, PTE-PU performs well on both datasets because it considers the heterogeneity in news and the heterogeneity between news and the KG. However, HEER outperforms PTE-PU because it considers the trade-offs between news and the KG.

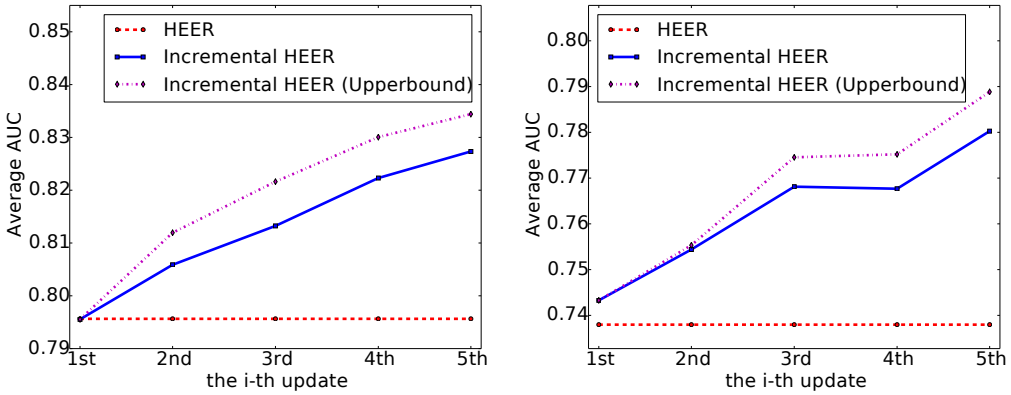
In summary, with the heterogeneous textual graph modeling, embedding and PU learning, the proposed HEER outperforms the baseline methods for both datasets.

4.4.3 Case Study

In this subsection, we present a case study to further show the effectiveness of the proposed HEER framework. We focus on the Yahoo! News dataset and show several examples of the discovered emerging relations in Table XX. The tag "(in)" and "(out)" next to an entity indicates whether this entity is in the KG or not. These relations are detected only by HEER and all the other baselines fail to discover these relations. It can be observed that most of these relations appear only once in Yahoo! News. For example, there is only one piece of news talking about the entity pair (*Elizabeth McGovern, Downton Abbey*). When there are rare appearances of the emerging relations, the word-level based and the sentence-level based models cannot extract sufficient features. Therefore, they cannot detect these emerging relations effectively. Since the proposed HEER framework builds a heterogeneous textual graph from news, it can capture the global corpus information for emerging relations with few sentences.

4.4.4 Incremental HEER

When we incrementally update the KG with the detected emerging relations, HEER can make use of an up-to-date KG during the embedding process to discover the latest emerging relations more accurately. However, the incremental HEER method may add some false positive relations into the KG and reduce the quality of the KG. With more incremental updates, the noisy information may be accumulated and the error rate of HEER may increase. In this



(a) Yahoo! News (b) BBC News

Figure 13. The performance of the incremental HEER.

subsection, we analyze how HEER performs on the two datasets when we incrementally update the KG with some false positive emerging relations.

We take the KG and 50% of news to pre-train the embeddings and split the remaining news randomly into five equal chunks for testing the incremental HEER. During each incremental update procedure, we detect the emerging relations on one chunk and update the KG with those highly-reliable emerging relations. In the incremental HEER, we would select highly-reliable emerging relations to update the KG. In the experiment, we set the threshold $\rho = 0.95$ to select those highly-reliable emerging relations.

In order to evaluate the effectiveness of the proposed incremental HEER, two baselines are compared. One baseline is the HEER method without updating the KG. We just test the performance of HEER on each chunk using the model trained on the original 50% of news. The other one is the incremental HEER method that updates the KG with all the true positive emerging relations in the given chunk. Actually this is the optimal method to update the KG.

It can show how the false positive emerging relations will affect the results. Since the order of chunks may influence the updating performance, in the experiment, we repeat 5 times with different orders of chunks. We only report the average AUC scores in Figure 13 because we have similar observations in other metrics.

It can be observed that the incremental HEER always outperforms HEER on both datasets no matter how many updates we do. With more updates, the performance is improving because more information is utilized to detect emerging relations. We can also observe that the difference between the incremental HEER and the optimal method is quite small, indicating that the added false positive emerging relations have little influence on the performance during each incremental update.

4.4.5 Parameter Analysis

In the proposed HEER framework, θ controls the relative importance of the KG guidance in detecting emerging relations. Here we assess the benefit of HEER with different values of the guiding parameter θ. Figure 14 represents the classification AUC scores and accuracies on both datasets. We observe that HEER can discover emerging relations more effectively with the help of the KG. Particularly, when there is no guidance of the KG, i.e., $\theta = 0$, HEER can perform well but not the best. When we increase θ, the performance becomes best for Yahoo! News when θ is around 0.2. For BBC News, $\theta \in [0.1, 0.4]$ yields similar good results. Figure 14 also shows that it is much easier to detect emerging relations from news than from the KG, because the performance drops when we increase θ from 0.5 to 1.

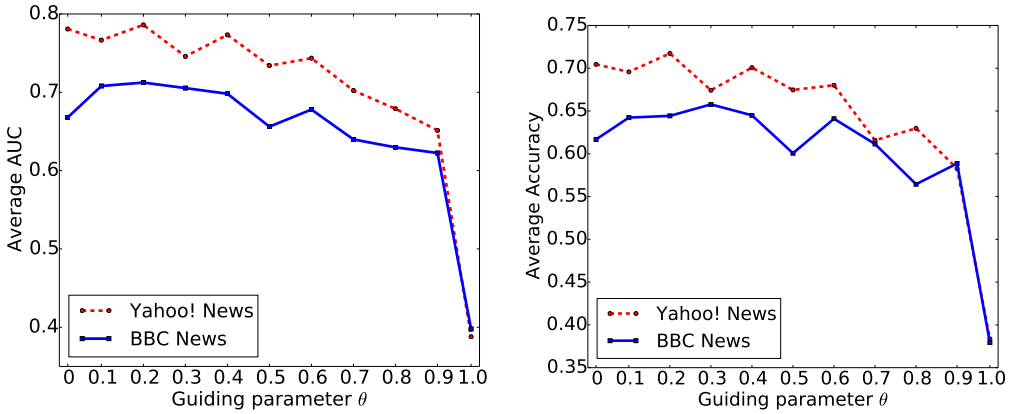


Figure 14. The performance of HEER with different guiding parameters.

We also demonstrate the effect of the dimensionality of embeddings by fixing the other parameters. As shown in Figure 15, the best accuracy is achieved when the embedding dimension is 50 for both datasets. We can observe that the performance of HEER fluctuates a lot on BBC News. The reason may be that Yahoo! News is a larger dataset and HEER can be more stable on a larger dataset. We further analyze the performance of HEER with different numbers of iteration T. Figure 16 reports the average AUC and accuracy results. It can be seen that HEER converges after around 200 iterations on both datasets.

4.5 Related Work

Relation extraction from texts has been well studied in recent years (98; 74; 75; 76; 99). For example, a distantly-supervised learning is performed in (75) by using relation instances in the knowledge graph of Freebase instead of annotated texts as their source of supervision. It extracts a large amount of sentence-level features including lexical part-of-speech tags of words, syntactical dependency tree paths, etc.. However, there are few sentences about the emerging

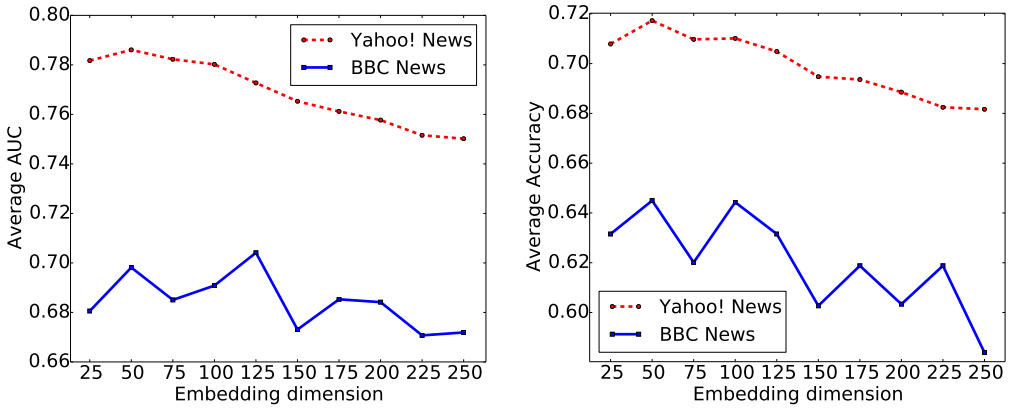


Figure 15. The performance of HEER with different embedding dimensions.

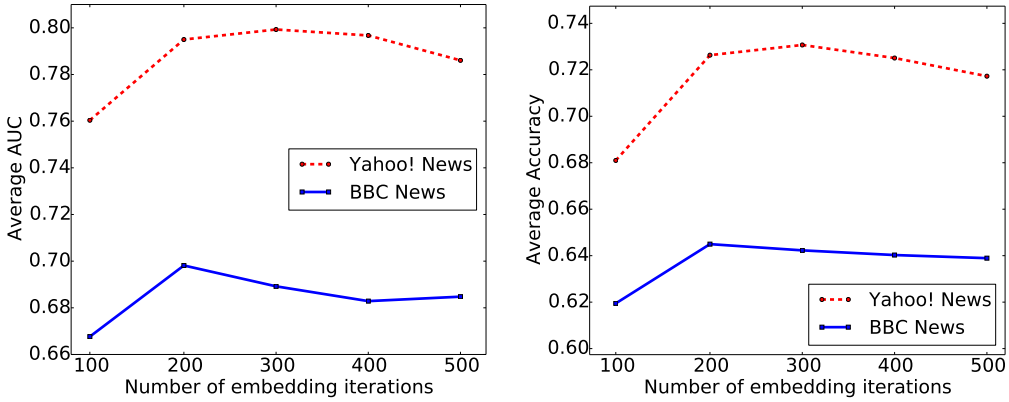


Figure 16. The performance of HEER with different embedding iterations.

relations at the beginning. Simply relying on sentence-level features could lead to sub-optimal results for emerging relation detection.

Due to the limited coverage of the existing knowledge graph (KG) (78), the task of KG completion has received a lot of attention (81; 83; 100; 84; 86). There are two branches for this task. One is to learn embedding representations of entities and relations in the KG and

use these embeddings to infer missing relations (89; 90; 80; 91; 85). The other branch is to predict missing relations from a graph view (87; 82; 88; 101). For instance, the Path Ranking Algorithm (PRA) (87; 88) performs link prediction in the KG via a random walk inference technique. In addition, the research work (101) combines methods of the above two branches by using a recursive neural network to create embedded representations of PRA-style paths. In our work, the emerging relations have new entities that are not included in the KG. Hence, it is impossible to apply these techniques directly. Furthermore, some work tries to embed the knowledge graph and the texts jointly (102). Our work is different since we focus on the emerging relations from news and model the news into a heterogeneous textual graph. We further update the KG incrementally with the detected emerging relations.

Our work is also related to the problem of information network modeling and mining (1; 2). Recently, there are some work aiming at embedding real-world large-scale networks(96; 97; 94). DeepWalk and LINE are proposed in (96) and (94), respectively. These two models can only handle homogeneous networks. PTE is proposed in (97) to deal with heterogeneous networks.

The positive-unlabeled (PU) learning techniques have been proposed for many years (93; 103; 104). We apply the PU learning technique proposed in (93) for the detection of emerging relations since it can easily adjust a standard positive-negative classifier to a positive-unlabeled classifier with a constant factor.

CHAPTER 5

CONNECTING EMERGING RELATION TYPES FROM NEWS TO KNOWLEDGE GRAPHS

(This chapter includes my paper under review as Jingyuan Zhang, Chun-Ta Lu, Bokai Cao, Yi Chang and Philip S. Yu, "Connecting Emerging Relationships from News via Tensor Factorization" for *KDD'17* (4).)

5.1 Introduction

Knowledge graphs (KGs), such as Freebase and DBpedia, have been widely used to represent relationships between entities in the form of triplets (h, r, t). Here h and t are two entities (head and tail) and r is a relation. Each triplet is a relation instance[1]. Entities can be persons, organizations, locations, etc., and examples of relations can be person-affiliation and organization-location. The KGs with relations and entities are useful sources for many real-word applications in information extraction, natural language understanding and information retrieval. However, current KGs have limited coverage of real-world relationships (78), especially for new entities that arise with new relationships emerging over time (105; 3). Fortunately, with the latest information in news, there is an opportunity to connect emerging relationships from news timely. Consider Figure 17 as an example, where an emerging rela-

[1]For simplicity, a relation means a relation type in KGs, and a relationship means a triple instance (h, r, t).

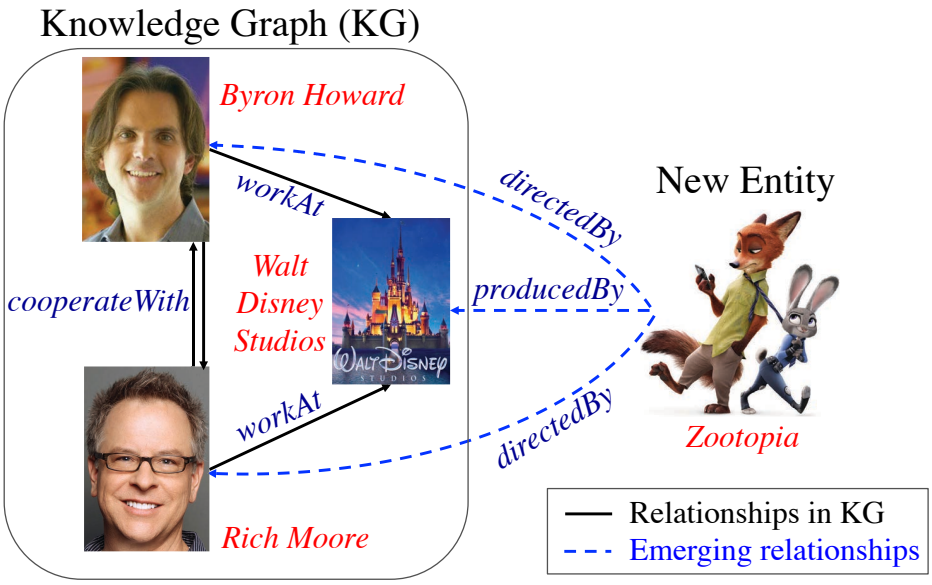


Figure 17. An example of emerging relationships.

tionship appears when the publisher "Walt Disney Studios" produces a new movie "Zootopia". Although there is no information regarding Zootopia in KGs, many pieces of news are talking about this new movie, the publisher and the directors. Detecting such emerging relationships has many benefits in practice. For example, the current KGs can be expanded and updated with emerging relationships. In addition, emerging relationships can help news related tasks, such as news retrieval and ranking, event detection, etc..

However, learning emerging relationships from text descriptions is a challenging task due to the source heterogeneity of structured KGs and unstructured news texts. Although many research works try to mitigate the problem of knowledge sparsity in KGs (80; 90; 81; 84; 85; 86), the main focus of these works is to utilize the structural information to fill in the missing

relationships in KGs. For instance, Path Ranking Algorithm (PRA) (87; 88) completes KGs by performing random walk techniques; some other studies embed entities and relations into a low-dimensional space and infer missing relationships by translating relations from head entities to tail entities in KGs (89; 86; 100). However, it is nontrivial to incorporate news texts into these structure-based methods to connect emerging relationships.

From the other aspect, there are several studies attempting to embed large-scale texts (57; 96; 94; 97). For example, LINE is proposed in (94) to embed texts into a low-dimensional space by constructing a homogeneous word co-occurrence network from texts. Later PTE is proposed in (97) to improve the LINE method by building a heterogeneous text network. These methods focus on embedding every single word in texts but ignoring the semantic relations among words. Therefore, they would fail to capture emerging relationships from news texts. Recently, some work tries to embed the KG and the texts jointly (102; 3). However, the method in (102) embeds the KG and texts separately. So their indirect inference according to texts cannot help connect emerging relationships effectively. Though the method in (3) aims to detect emerging relationships, it ignores the relation types.

To address the issue, in this chapter, we propose a tensor-based framework to combine KGs and news texts effectively for emerging relationship connection. A fourth-order tensor structure is used to capture the hidden connection between multiple relations in KGs and multiple text descriptions of relations in news. Specifically, though the tensor product of feature spaces of entities, relations and texts in the built tensor structure, we capture the multimodal interactions among head entities, relations, tail entities, and text descriptions. Figure 18 shows the tensor

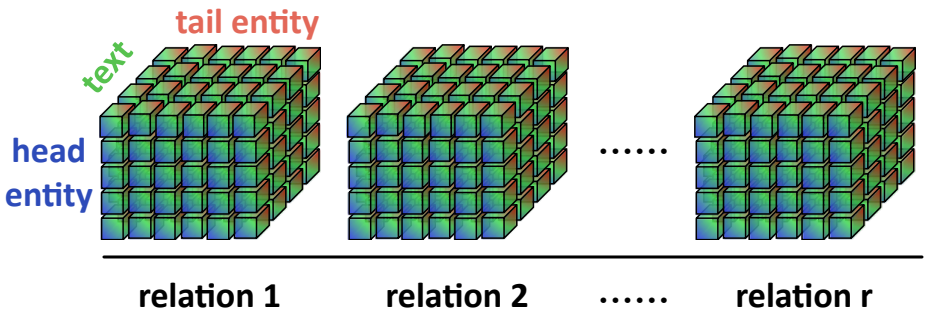


Figure 18. The built fourth-order tensors.

structure constructed from KGs and news. As the interactions (i.e., tensor product) of entities, relations and text descriptions can reflect the connection between KGs and news, we use them to learn the embedding representations of entities and relations. In this manner, it can deal with multiple relations without difficulty.

Noteworthily, directly learning from the tensor structure would be problematic. First, the space complexity of building the fourth-order tensor is polynomial to the numbers of entities, relations and text descriptions, making it challenging to fit the tensor into memory. Second, due to the large number of interaction parameters, it is time-consuming to decompose the built tensor directly and it is prone to overfitting. Last but not least, how to learn a meaningful representation of a given news sentence, which consists of the head and tail entities with the other text descriptions, and connect the representation to the relation types in KGs is challenging.

In order to solve the above challenges, we further develop a **T**ext-**A**ware **MU**lti-**RE**lational learning method (TAMURE) to learn the embeddings of entities and relations from both news

and KGs. By factorizing the interaction parameters, the proposed TAMURE method can efficiently learn the latent representation of entities and text descriptions, without physically building the tensor. Since the parameters are learned jointly through the factorization, it also improves the accuracy of the parameter estimation under sparsity and provides the model with the power of avoiding overfitting. In summary, our contributions are as follows:

- We formulate a new task, connecting emerging relationships, which is to discover relations with new entities by fusing information from heterogeneous sources, i.e., the structured KGs and the unstructured news texts.
- We introduce a novel tensor-based framework to connect emerging relationships from news. The news texts and KGs are fused into an elegant fourth-order tensor formulation, where the complex multiple interactions among relations, entities and text descriptions are embedded within the tensor structure.
- The proposed TAMURE method can efficiently recognize the emerging relationships from news and KGs, by learning the joint representation of entities and text descriptions. Furthermore, the complexity of TAMURE is linear to the number of interaction parameters, making it scalable to large KGs and news texts.
- We demonstrate the effectiveness of TAMURE by comparing it with eight state-of-the-art methods via TensorFlow on real-world KG and news data.

5.2 Preliminary

In this chapter, we study the problem of connecting emerging relationships from news. Before proceeding, we first introduce the related concepts, and then state the problem of emerging relationship detection from news. Table XXI lists basic symbols that will be used throughout the chapter.

5.2.1 Basic Concepts

Definition 12 ***Entity, Relation and Relationship:*** *An entity e can represent a person, an organization, or a location, etc.. A relation can be a person-affiliation type or an organization-location type. A relationship is defined in the form of triplets (h, r, t), where h is a head entity, t is a tail entity and r is a relation. For each possible triple (h, r, t), we use $y \in \{0, 1\}$ to indicate whether the triple exists.*

Definition 13 ***Knowledge Graph (KG):*** *A knowledge graph is denoted as a directed graph $\mathcal{G}_{kg} = (E_{kg}, \mathcal{E}_{kg})$, where E_{kg} is the set of entities and $\mathcal{E}_{kg}$ is the set of known relationships. Each directed edge in $\mathcal{E}_{kg}$ can be represented by a triplet (h, r, t), where the entities h and $t \in E_{kg}$, and the relation r is one type of the relations in the relation set R.*

Nowadays, due to the rapid growth of real-world knowledge, large volumes of *emerging relationships* are arising with time. An emerging relationship is defined as follows:

Definition 14 ***Emerging Relationship:*** *An emerging relationship (h, r, t) exists, if its label $y = 1$ in the real world and at least one entity is not included in the given KG (i.e., $h \notin E_{kg}$ or $t \notin E_{kg}$).*

TABLE XXI

List of basic symbols in Chapter 5.

Symbol	Definition and description
(h, r, t)	a relationship
(h, r, t, d)	a relationship with a text description d
x	a scale is represented by each lowercase letter
$\mathbf{x}$	a vector is represented by each boldface lowercase letter
$\mathbf{X}$	a matrix is represented by each boldface uppercase letter
$\mathcal{X}$	a tensor, set or space is represented by each calligraphic letter
$[1 : M]$	a integer set with the range of 1 to M inclusively
$\langle \cdot, \cdot \rangle$	inner product
$\circ$	tensor product (outer product)
$*$	Hadamard (element-wise) product

For example, in Figure 17, (*Zootopia, producedBy, Walt Disney Studios*) is an emerging relationship with $y = 1$ since *Zootopia* is a new movie entity and it is produced by the publisher entity *Walt Disney Studios*. Similarly, (*Zootopia, directedBy, Byron Howard*) and (*Zootopia, directedBy, Rich Moore*) are also examples of emerging relationships.

The key of this work is to apply the tensor structure to fuse the KG and the news for connecting emerging relationships. In the following, we introduce some related concepts and notations about the tensor.

Definition 15 ***Tensor:*** *Tensors are higher order arrays with the generalization of the first order vector notions and second order matrix notions. Following (106), we denote an M-th*

order tensor as $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_M}$ *and its elements by* $x_{i_1,\cdots,i_M}$. *A lowercase letter denotes an index. The range of the index spans from* 1 *to the uppercase letter of the index,* e.g., $i = 1, 2, \cdots, I$. *All vectors are column vectors in this chapter.*

We denote the i-th row and j-th column vector of an arbitrary matrix $\mathbf{X} \in \mathbb{R}^{I \times J}$ as $\mathbf{x}^i$ and $\mathbf{x}_j$, respectively. We denote the inner product of two same-sized tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times \cdots \times I_M}$ as $\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1=1}^{I_1} \cdots \sum_{i_1=1}^{I_M} x_{i_1,\cdots,i_M} y_{i_1,\cdots,i_M}$. We define the outer product of M vectors $\mathbf{x}^{(m)} \in \mathbb{R}^{I_m}$ for $m \in [1:M]$ as an M-th order tensor. The elementwise can be represented by $\left(\mathbf{x}^{(1)} \circ \cdots \circ \mathbf{x}^{(M)}\right)_{i_1,\cdots,i_M} = x_{i_1}^{(1)} \cdots x_{i_M}^{(M)}$ for all index values. Specifically, it holds that

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \prod_{m=1}^{M} \langle \mathbf{x}^{(m)}, \mathbf{y}^{(m)} \rangle = \prod_{m=1}^{M} {\mathbf{x}^{(m)}}^{\mathrm{T}} \mathbf{y}^{(m)} \tag{5.1}$$

for $\mathcal{X} = \mathbf{x}^{(1)} \circ \cdots \circ \mathbf{x}^{(M)}$ and $\mathcal{Y} = \mathbf{y}^{(1)} \circ \cdots \circ \mathbf{y}^{(M)}$.

The CANDECOMP / PARAFAC (CP) factorization for a general tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_M}$ is

$$\mathcal{X} = \sum_{k=1}^{K} \mathbf{x}_k^{(1)} \circ \cdots \circ \mathbf{x}_k^{(M)} = [\![\mathbf{X}^{(1)}, \ldots, \mathbf{X}^{(M)}]\!]. \tag{5.2}$$

Here $\mathbf{X}^{(m)} = [\mathbf{x}_1^{(m)}, \cdots, \mathbf{x}_K^{(m)}]$ are factor matrices of size $I_m \times K$ for $m \in [1:M]$. K is the number of factors. We use $[\![\cdot]\!]$ for shorthand.

5.2.2 Problem Statement

Given a large collection of news and the existing KG $\mathcal{G}_{kg}$, the task of connecting emerging relationships aims to determine the existence of multiple types of relations among entities. Assume we can extract a set of relationship candidates, each of which is represented by a

triplet (h, r, t) and is associated with a vector of text descriptions $\mathbf{d}$ from news. Since the entities in the candidates may not exist in the entity set E_{kg} of the KG, we denote E_{news} as the set of entities that appear in the news, and denote $E = E_{kg} \cup E_{news}$ as the set of all the entities. The task of connecting emerging relationships is to learn a score function $f : (h, r, t, \mathbf{d}) \rightarrow \{0, 1\}$ that correctly predicts the label of the test instance, where the entities h and $t \in E$, $r \in R$ and $(h, r, t) \notin \mathcal{E}_{kg}$.

5.3 Proposed Method

In this section, we first introduce how to design the tensor-based score function for fusing the KG and the news texts. Then we derive an efficient **T**ext-**A**ware **MU**lti-**RE**lational learning method (TAMURE) that learns the embeddings of entities and relations in linear complexity.

5.3.1 Tensor-based Score Function

We begin by introducing how to design the score function if only one of the sources (the KG or the news texts) is available. We then show that both sources can be integrated into an elegant tensor-based model.

Without considering the existence of the KG, the most common approach is using a linear score function for each relation r based on the text description vector $\mathbf{d} \in \mathbb{R}^{I_D}$ extracted from news:

$$f_{\text{news}}(r, \mathbf{d}) = \sum_{i=1}^{I_D} u_{r,i} d_i = \mathbf{d}^{\mathrm{T}} \mathbf{u}_r, \tag{5.3}$$

where $\mathbf{u}_r \in \mathbb{R}^{I_D}$ is the weight vector for relation r. For learning multiple relations at the same time, we let $I_R = |R|$ denote the number of relations and $\mathbf{U} \in \mathbb{R}^{I_D \times I_R}$ be the weight matrix to be learned. The column vectors are $\mathbf{u}_r$. Let $\mathbf{e}_r \in \mathbb{R}^{I_R}$ denote the relation indicator vector

$$\mathbf{e}_r = [\underbrace{0, \cdots, 0}_{\text{r-1}}, 1, 0, \cdots, 0]^{\mathrm{T}}.$$

We can observe that $\mathbf{U}$ is actually the weight matrix of a bilinear feature map for modeling the second-order interactions between the text description vector and the relation:

$$f_{\text{news}}(r, \mathbf{d}) = \mathbf{d}^{\mathrm{T}} \mathbf{U} \mathbf{e}_r = \langle \mathbf{U}, \mathbf{d} \circ \mathbf{e}_r \rangle . \tag{5.4}$$

Given the KG only, we can use a similar score function for modeling the interactions between the head entity, tail entity and relation. Let $I_E = |E|$ denote the number of entities, and let $\mathbf{e}_h \in \mathbb{R}^{I_E}$ and $\mathbf{e}_t \in \mathbb{R}^{I_E}$ denote the head and tail entity indicators, respectively. We can form the multilinear score function by

$$f_{\text{KG}}(h, r, t) = \mathbf{e}_h^{\mathrm{T}} \mathbf{V}_r \mathbf{e}_t = \langle \mathcal{V}, \mathbf{e}_r \circ \mathbf{e}_h \circ \mathbf{e}_t \rangle . \tag{5.5}$$

where $\mathbf{V}_r \in \mathbb{R}^{I_E \times I_E}$ is the weight matrix between head entities and tail entities for relation r, and $\mathcal{V} \in \mathbb{R}^{I_R \times I_E \times I_E}$ is a stacked tensor with each slice $\mathcal{V}_{r,:,:} = \mathbf{V}_r$.

Obviously, we can fuse the news and the KG together by formulating the score function as follows:

$$f(h, r, t, \mathbf{d}) = \langle \mathcal{W}, \mathbf{e}_r \circ \mathbf{e}_h \circ \mathbf{e}_t \circ \mathbf{d} \rangle \,, \tag{5.6}$$

where $\mathcal{W} \in \mathbb{R}^{I_R \times I_E \times I_E \times I_D}$ is the weight tensor to be learned.

It is worthy to be noted that some existing relationships in the KG might not have the text description $\mathbf{d}$, such that the text description $\mathbf{d} = \mathbf{0}$. Besides, in order to include relationships that are extracted from the news but not available in the KG, we add a "co-occurrence" relation type to the existing relations during the learning process and do not test this relation type in the testing phase.

5.3.2 Text-aware Multi-relational Learning

Now the news texts and the KG have been fused into a fourth-order tensor formulation that embeds the multiple complex interactions among relations, entities and text descriptions. However, the space complexity of building the fourth-order tensor is $O(I_R \times I_E \times I_E \times I_D)$. With large volumes of emerging relationships in news, it is impractical to physically build the tensor. Moreover, directly learning the weight tensor $\mathcal{W}$ would be problematic. First, due to the large number of parameters $I_R \times I_E \times I_E \times I_D$, the learning procedure is prone to overfitting and less effective coupled with its sparse counterpart $\mathbf{e}_r \circ \mathbf{e}_h \circ \mathbf{e}_t \circ \mathbf{d}$. Second, since the weight parameters are learned independently, it cannot model the interactions that never appear. To address such issues, we propose an efficient method based on tensor factorization.

By assuming that there is a low rank for the effect of interactions, we factorize the weight tensor $\mathcal{W}$ as

$$\mathcal{W} = [\![\mathbf{M}^{(r)}, \mathbf{M}^{(h)}, \mathbf{M}^{(t)}, \mathbf{M}^{(d)}]\!],$$

where $\mathbf{M}^{(r)} \in \mathbb{R}^{I_R \times K}$ and $\mathbf{M}^{(h)}$, $\mathbf{M}^{(t)} \in \mathbb{R}^{I_E \times K}$ represent the embedding matrices for relations, head entities and tail entities, respectively; $\mathbf{M}^{(d)} \in \mathbb{R}^{I_D \times K}$ represents the weight matrix for text descriptions. From Equation 5.1 and Equation 5.6, we can easily derive that

$$\begin{aligned} f(h, r, t, \mathbf{d}) &= \sum_{k=1}^{K} \left\langle \mathbf{M}^{(r)}_{:,k} \circ \mathbf{M}^{(h)}_{:,k} \circ \mathbf{M}^{(t)}_{:,k} \circ \mathbf{M}^{(d)}_{:,k} \ , \ \mathbf{e}_r \circ \mathbf{e}_h \circ \mathbf{e}_t \circ \mathbf{d} \right\rangle \\ &= \sum_{k=1}^{K} \left(\mathbf{e}_r^{\mathrm{T}} \mathbf{M}^{(r)}_{:,k}\right) \left(\mathbf{e}_h^{\mathrm{T}} \mathbf{M}^{(h)}_{:,k}\right) \left(\mathbf{e}_t^{\mathrm{T}} \mathbf{M}^{(t)}_{:,k}\right) \left(\mathbf{d}^{\mathrm{T}} \mathbf{M}^{(d)}_{:,k}\right) \\ &= \left(\mathbf{e}_r^{\mathrm{T}} \mathbf{M}^{(r)}\right)^{\mathrm{T}} \left(\left(\mathbf{e}_h^{\mathrm{T}} \mathbf{M}^{(h)}\right) * \left(\mathbf{e}_t^{\mathrm{T}} \mathbf{M}^{(t)}\right) * \left(\mathbf{d}^{\mathrm{T}} \mathbf{M}^{(d)}\right)\right) \\ &= \mathbf{r}^{\mathrm{T}} \left(\mathbf{h} * \mathbf{t} * (\mathbf{d}^{\mathrm{T}} \mathbf{M}^{(d)})\right), \end{aligned} \tag{5.7}$$

where $*$ is the Hadamard (elementwise) product, $\mathbf{r} = \mathbf{e}_r^{\mathrm{T}} \mathbf{M}^{(r)}$, $\mathbf{h} = \mathbf{e}_h^{\mathrm{T}} \mathbf{M}^{(h)}$ and $\mathbf{t} = \mathbf{e}_t^{\mathrm{T}} \mathbf{M}^{(t)}$ are the embedding vectors learned for the relation r, head entity h and tail entity t, respectively.

One can notice that Equation 5.7 only models the fourth-order interactions between the specific relation r, entities h and t, and the associated text description $\mathbf{d}$. However, the lower-order interactions can also be discriminative for determining the existence of the relationship. For example, if the head entity is a person, e.g., "Rich Moore", the relation r is unlikely to be "producedBy" no matter which tail entity t is chosen in the sample instance $(h, r, t, \mathbf{d})$. In this case, even the pairwise interaction between h and r can be discriminative. Thus, we consider to

incorporate the lower-order interactions in the predictive model. This can be done by adding bias vectors in Equation 5.7 as follows:

$$f(h, r, t, \mathbf{d}) = \mathbf{r}^{\mathrm{T}} \left((\mathbf{h} + \mathbf{b}_h) * (\mathbf{t} + \mathbf{b}_t) * \left(\mathbf{d}^{\mathrm{T}} \mathbf{M}^{(d)} + \mathbf{b}_v \right) \right), \tag{5.8}$$

where $\mathbf{b}_v \in \mathbb{R}^{I_D}$, $\mathbf{b}_h$ and $\mathbf{b}_t \in \mathbb{R}^{I_E}$ are the bias vectors that are independent to the given instance. To illustrate why the bias vectors can help model the lower-order interactions, we can decompose Equation 5.8 into two parts. The first part $\mathbf{r}^{\mathrm{T}} \left(\mathbf{h} * (\mathbf{t} + \mathbf{b}_t) * \left(\mathbf{d}^{\mathrm{T}} \mathbf{M}^{(d)} + \mathbf{b}_v \right) \right)$ models the fourth-order interactions between $(h, r, t, \mathbf{d})$, while the second part $\mathbf{r}^{\mathrm{T}} \left(\mathbf{b}_h * (\mathbf{t} + \mathbf{b}_t) * \left(\mathbf{d}^{\mathrm{T}} \mathbf{M}^{(d)} + \mathbf{b}_v \right) \right)$ with the bias vector $\mathbf{b}_h$ models the third-order as well as the lower-order interactions between $(r, t, \mathbf{d})$ without the head entity h involved. Other types of the third-order interactions and the lower-order interactions can be derived in a similar way.

We name the model in Equation 5.8 as **T**ext-**A**ware **MU**lti-**RE**lational learning method (TAMURE). The work flow of TAMURE is illustrated in Figure 19. After mapping the head entity, tail entity and text description extracted from news into a common embedding space via $(\mathbf{h} + \mathbf{b}_t) * (\mathbf{t} + \mathbf{b}_t) * \left(\mathbf{d}^{\mathrm{T}} \mathbf{M}^{(d)} + \mathbf{b}_v \right)$, TAMURE learns a meaningful representation to connect relation types in KGs with news texts via $\mathbf{r}^{\mathrm{T}} \left((\mathbf{h} + \mathbf{b}_h) * (\mathbf{t} + \mathbf{b}_t) * \left(\mathbf{d}^{\mathrm{T}} \mathbf{M}^{(d)} + \mathbf{b}_v \right) \right)$.

Clearly, the parameters of the interactions among multiple relations, entities, text descriptions are jointly factorized. The joint factorization help improve parameter estimation under sparsity because there are dependencies when the same entities or text descriptions are shared by the interactions. Hence, in highly sparse data, TAMURE can learn the model parameters

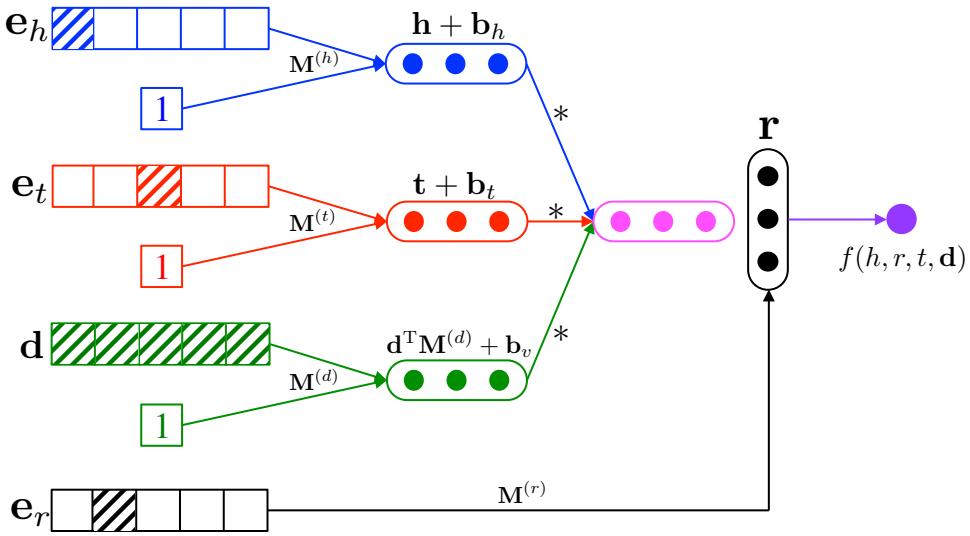


Figure 19. The work flow of TAMURE.

effectively without observing such interactions directly. Moreover, by modeling the interactions with the bias vectors, this joint factorization can deal with missing or incomplete information for relationships easily.

Due to the multilinear analysis, TAMURE has another appealing character of avoiding to physically construct the fourth-order tensor after the factorization of the weight tensor $\mathcal{W}$. Moreover, the model complexity is $O(K(I_R + I_E + I_D))$, which is linear in the number of parameters. With this multilinear property, TAMURE can save memory and speed up the learning procedure.

5.3.3 Learning Procedure of TAMURE

Given the training set $\mathcal{D} = \{(h, r, t, \mathbf{d})\}$, TAMURE learns vector embeddings of the entities, relations and text descriptions via the score function f. Following the same strategy as in (89; 85), we minimize a margin-based ranking criterion over the training set:

$$\mathcal{L} = \sum_{(h,r,t,\mathbf{d})\in\mathcal{D}} \sum_{(h',r,t',\mathbf{d}')\in\mathcal{D}'} \max(0, f(h, r, t, \mathbf{d}) + \gamma - f(h', r, t', \mathbf{d}')), \tag{5.9}$$

where $\gamma > 0$ is a margin hyper-parameter, and

$$\mathcal{D}' = \{(h', r, t, \mathbf{d}')|h' \in E\} \cup \{(h, r, t', \mathbf{d}')|t' \in E\}. \tag{5.10}$$

The set of corrupted relationships $\mathcal{D}'$ is composed of training data with either the head or tail replaced by a random entity (but not both at the same time). For a relationship from the KG, the corrupted data is checked to make sure that it is not in the KG. Notice that since the text description $\mathbf{d}$ is always associated with the entity pairs, when one of the entities is replaced, the text description vector will also be replaced accordingly. The loss function in Equation 5.9 favors higher scores for training data than for corrupted data.

The learning process of TAMURE is carried out by Adam optimizer (107) in mini-batch mode, with the additional max-norm regularization constraint (89; 108), which constrains the L_2-norm of the embeddings of the entities to be no larger than 1. No regularization or norm constraints are given to the relation and text description embeddings. The detailed optimization procedure is described in Algorithm 5. At each main iteration of the algorithm, a small set of

Algorithm 5 The TAMURE algorithm

Input: The training set $\mathcal{D}$, entities E and relations R, margin γ, embedding dimension K.
Output: Embeddings of entities, relations and texts.

Initialize embeddings $\mathbf{h}, \mathbf{r}, \mathbf{t} \leftarrow \text{uniform}(-\frac{1}{\sqrt{K}}, \frac{1}{\sqrt{K}})$ for each $h \in E$, $t \in E$ and $r \in R$
Initialize weight parameters randomly
Normalize entity embeddings
while not convergence **do**
  //sample a mini-batch of size m
  $\mathcal{D}_{batch} \leftarrow \text{sample}(\mathcal{D}, m)$
  $\mathcal{S}_{batch} \leftarrow \emptyset$
  for $(h, r, t, \mathbf{d}) \in \mathcal{D}_{batch}$ **do**
    // Sample a corrupted instance
    $(h', r, t', \mathbf{d}') \leftarrow \text{sample}(\mathcal{D}')$
    $\mathcal{S}_{batch} \leftarrow \cup\{((h, r, t, \mathbf{d}), (h', r, t', \mathbf{d}'))\}$
  end for
  Update embeddings and weight parameters w.r.t. the gradient of Equation 5.9 on $\mathcal{S}_{batch}$
  Constrain entity embeddings with the max-norm regularization
end while

data is sampled from the training set and servers as the training data of the mini-batch. For each existing relationship, a single corrupted relationship is sampled accordingly. The parameters are then updated by taking a gradient step with a constant learning rate. Before the next iteration, the embedding vectors of the entities are normalized via the max-norm regularization constraint.

5.4 Experiments

In this section, we conduct extensive experiments to evaluate the proposed TAMURE method. After introducing the datasets and the experimental settings, we compare different baseline methods.

5.4.1 Data Processing

For the KG, we use the FB15k-237[1] dataset (109) with 14,541 entities and 237 relations extracted from Freebase. FB15k-237 contains news texts extracted from 200 million sentences in the ClueWeb12 corpus coupled with Freebase mention annotations. There are around 3.9 million text descriptions corresponding to the relations in the KG. Almost all entities occur in news (13,937 out of 14,541). These texts are represented as full lexicalized dependency paths. For example, given the news text "Barack Obama is the 44th and current President of United States.", the dependency path for the relation between entity "Barack Obama" and "United States" is represented as "SUBJECT $\xleftarrow{\text{nsubj}}$ *president* $\xrightarrow{\text{prep}}$ *of* $\xrightarrow{\text{obj}}$ OBJECT". In the experiment, we extracted n-gram (n = 1, 2, 3) features from the lexicalized dependency paths and concatenated them together as the features of the text descriptions. In the process of n-gram feature extraction, we considered the lexical dependencies such as "nsubj" and "prep" as words. Those n-gram features with frequencies less than 10 are removed in the experiment. The statistics of the dataset are summarized in Table XXII.

In order to evaluate the effectiveness of the proposed TAMURE method, we randomly select half of the entities in FB15k-237 as new entities to avoid the high-cost of human labeling. Those relationships associated with new entities are emerging relationships. In the experiment, relationships with existing entities are considered as the training set from the KG. The emerging relationships are equally split into the validation set and the testing set. We generate the

---

[1] `https://www.microsoft.com/en-us/download/details.aspx?id=52312`

TABLE XXII. Statistics of the FB15k-237 dataset.

Entities			Relationships			Train / Validation / Test			News		
# Entities	# Existing Entities	# Emerging Entities	# Relationships	# Existing Relationships	# Emerging Relationships	# Training Relationships	# Validation Relationships	# Testing Relationships	# Entities	# Texts	# Text Descriptions
14,541	7,270	7,271	310,116	82,062	228,054	82,062	114,027	114,027	13,937	3,978,014	744,908

negative emerging relationships for the validation and testing sets by replacing each relation type with a random one.

5.4.2 Compared Methods

In order to show that the proposed TAMURE model can effectively connect emerging relationships, we compare the following nine methods.

• TransE: It is the classic KG embedding model by treating relations as translations from head entities to tail entities (89). Each entity is embedded into a low-dimensional vector space and each relation is represented as a translation vector. The score function of TransE is $f(h, r, t) = ||\mathbf{h} + \mathbf{r} - \mathbf{t}||_{\ell_2}$.

• Skip-Gram: It is the state-of-the-art word embedding model (57). It considers the KG and news texts together as a large corpus and learns embedding vectors for each word where each entity or relation is regarded as a word.

• DeepWalk: It is an embedding model for homogeneous graphs with binary edges (96). It learns embeddings of nodes by applying truncated random walks on the graph. By viewing entities, relations and words as the same type of nodes, we can build a homogeneous graph from news texts and KGs. Then we apply the DeepWalk model on this graph.

• LINE: It is the **L**arge-scale **I**nformation **N**etwork **E**mbedding method (LINE) (94). Similar to DeepWalk, LINE treats entities, relations and words as one type of nodes but considers the weights of edges when learning the embeddings. The weights are the frequencies of two nodes co-occurring in news or KGs.

• PTE: It is the **P**redictive **T**ext **E**mbedding method (PTE) (97). It learns embeddings of nodes from a heterogeneous graph built from news texts and KGs. The heterogeneous graph includes four types of graphs: a word-word co-occurrence graph, an entity-entity graph, a word-entity graph and an entity-relation graph.

• TransE+SG: It is based on the pTransE method for relationship inference from news texts and knowledge graphs (102). It first applies TransE for entity embeddings from KGs and Skip-Gram for word embeddings from news texts. Then the two models are combined via aligning the embeddings into the same space. Since the dataset FB15k-237 has already annotated entities in news, there is no need to apply the alignment model.

• RESCAL: It is a relational learning approach based on tensor factorization (110). It focuses on the KG and embeds relations into a matrix space that operates as a bilinear operator on entity embeddings. RESCAL applies a more flexible tensor decomposition than CP, as the relation embedding matrix introduces interaction terms for entity embeddings.

• TAMURE-KG: It is the proposed tensor-based framework on the KG only. We first build a third-order tensor about head entities, relations and tail entities from the KG. Then the proposed multi-relational factorization model is applied to learn embeddings. The score function of TAMURE-KG is $f(h, r, t) = \mathbf{r}^{\mathrm{T}}((\mathbf{h} + \mathbf{b}_h) * (\mathbf{t} + \mathbf{b}_t))$.

• TAMURE: It is the **T**ext-**A**ware **MU**lti-**RE**lational learning model (TAMURE) proposed in this chapter. We build a fourth-order tensor structure to combine the KG and news texts together for connecting emerging relationships.

We implement TAMURE via TensorFlow. For fair comparisons, we set the dimensionality of embeddings as 20 for all the above methods. For a given entity, its embedding vector is the same when the entity appears as the head or as the tail of a relation. The Adam algorithm (107) within TensorFlow is applied as the optimizer for TransE, RESCAL, TAMURE-KG and TAMURE. The learning rate for Adam optimizer is set as 0.01, the mini-batch size is set as 1000, the maximum number of epochs is set as 20, and the margin is set as 1 in the experiments. For the text embedding baselines, we follow the settings in (94; 97; 96; 57).

To evaluate the performance of the compared approaches, we turn the proposed TAMURE model into a binary classifier as in (85) by thresholding the real-valued output scores where the thresholds for each relation are found on the validation set. For the text embedding baselines (i.e., Skip-Gram, DeepWalk, LINE and PTE), we follow (97) to apply the logistic regression model in the LibLinear package[1] after learning the embeddings. The embedding concatenation of head entities, relations and tail entities is considered as the feature during the classification phase. Since there might be multiple relations between two entities, we measure the performance by adopting five popular multi-label evaluation metrics in the literature: *Micro F1* and *Macro F1* that evaluate the classifier's prediction performance of the label set and treat all binary

---

[1] `http://www.csie.ntu.edu.tw/~cjlin/liblinear/`

labels' micro/macro-average of *precision* and *recall* equally (111); *Average Accuracy* (*AvgAcc*), *Average AUC* (*AvgAuc*) and *Hamming Loss* (HL) that evaluate the average accuracy, AUC and error rate over all the binary labels (relations) (112).

5.4.3 Performance Evaluation

In this section, we report the performance of the compared methods. The testing emerging relationships can be categorized into two groups: one is about those with only one emerging entity and the other is about those relationships with both entities not in the KG. In order to show the effectiveness of the proposed TAMURE method, we not only show the performance over all emerging relationships, but also show the results for each group of emerging relationships. The performance with the rank of each method is reported in Table XXIII. It can be observed that the proposed TAMURE method consistently outperforms all the other eight baselines regardless of the groups of emerging relationships. Specifically, compared with the second best baseline, TAMURE achieves an improvement of 51% on the HL metric and 25% on the *AvgAcc* metric. Furthermore, in the case where both entities are emerging entities, TAMURE achieves a significant performance improvement (56% and 36% higher than the second best baseline on HL and *AvgAuc*, respectively), showing the capability of TAMURE capturing the interactions between the relations in KGs and the text descriptions in news.

Compared with the KG embedding baselines, TAMURE performs the best since it incorporates the news texts and the KG into an elegant fourth-order tensor formulation to capture complex interactions among relations, entities and text descriptions. For those emerging relationships with only one new entity, TransE achieves the second best results. However, without

TABLE XXIII. The classification performance on connecting emerging relation types.

(a) Results on emerging relationships with only one entity not in the KG.

	Methods								
	KG Embedding			Text Embedding				KG+Text Embedding	
Criteria	TransE	RESCAL	TAMURE-KG	Skip-Gram	DeepWalk	LINE	PTE	TransE+SG	TAMURE
Micro-F1↑	0.8239 (2)	0.7923 (4)	0.8001 (3)	0.0723 (9)	0.1019 (8)	0.2792 (7)	0.4263 (6)	0.7893 (5)	**0.8738** (1)
Macro-F1 ↑	0.7022 (2)	0.6746 (4)	0.6810 (3)	0.1189 (9)	0.1603 (8)	0.4296 (6)	0.3215 (7)	0.6654 (5)	**0.7827** (1)
AvgAcc ↑	0.5905 (2)	0.5635 (3)	0.5623 (4)	0.3777 (8)	0.3907 (7)	0.5184 (6)	0.3072 (9)	0.5188 (5)	**0.7240** (1)
AvgAuc ↑	0.6425 (2)	0.4860 (5)	0.5759 (3)	0.1420 (9)	0.1860 (8)	0.4752 (6)	0.1947 (7)	0.5187 (4)	**0.7370** (1)
HL ↓	0.4096 (2)	0.4365 (3)	0.4377 (4)	0.6223 (8)	0.6093 (7)	0.4816 (6)	0.6928 (9)	0.4812 (5)	**0.2760** (1)

(b) Results on emerging relationships with both entities not in the KG.

	Methods								
	KG Embedding			Text Embedding				KG+Text Embedding	
Criteria	TransE	RESCAL	TAMURE-KG	Skip-Gram	DeepWalk	LINE	PTE	TransE+SG	TAMURE
Micro-F1↑	0.7625 (5)	0.7742 (2)	0.7704 (3)	0.1703 (9)	0.2540 (8)	0.3751 (7)	0.6125 (6)	0.7656 (4)	**0.8461** (1)
Macro-F1 ↑	0.6450 (5)	0.6564 (2)	0.6502 (4)	0.1936 (9)	0.2637 (8)	0.5103 (7)	0.5748 (6)	0.6513 (3)	**0.7555** (1)
AvgAcc ↑	0.5090 (3)	0.5035 (5)	0.5080 (4)	0.3243 (9)	0.3438 (8)	0.5025 (6)	0.4266 (7)	0.5150 (2)	**0.6885** (1)
AvgAuc ↑	0.5028 (5)	0.5060 (3)	0.5049 (4)	0.1577 (9)	0.2025 (8)	0.4758 (6)	0.2045 (7)	0.5199 (2)	**0.7081** (1)
HL ↓	0.4910 (3)	0.4965 (5)	0.4920 (4)	0.6757 (9)	0.6562 (8)	0.4975 (6)	0.5734 (7)	0.4850 (2)	**0.3115** (1)

(c) Results on all emerging relationships.

	Methods								
	KG Embedding			Text Embedding				KG+Text Embedding	
Criteria	TransE	RESCAL	TAMURE-KG	Skip-Gram	DeepWalk	LINE	PTE	TransE+SG	TAMURE
Micro-F1↑	0.8033 (2)	0.7863 (4)	0.7902 (3)	0.1083 (9)	0.1599 (8)	0.3139 (7)	0.4948 (6)	0.7819 (5)	**0.8647** (1)
Macro-F1 ↑	0.6877 (2)	0.6675 (4)	0.6700 (3)	0.1414 (9)	0.1918 (8)	0.4459 (6)	0.4129 (7)	0.6632 (5)	**0.7754** (1)
AvgAcc ↑	0.5723 (2)	0.5442 (4)	0.5467 (3)	0.3658 (8)	0.3796 (7)	0.5138 (6)	0.3439 (9)	0.5210 (5)	**0.7168** (1)
AvgAuc ↑	0.6050 (2)	0.4932 (5)	0.5517 (3)	0.1531 (9)	0.1966 (8)	0.4768 (6)	0.2514 (7)	0.5073 (4)	**0.7326** (1)
HL ↓	0.4277 (2)	0.4558 (4)	0.4533 (3)	0.6342 (8)	0.6204 (7)	0.4862 (6)	0.6562 (9)	0.4791 (5)	**0.2832** (1)

the help of news texts, TransE cannot perform well if the relationship has both entities not in the KG. Since emerging relationships usually first appear in news, it is important and necessary to consider news during the detection process. With specific designs for the KG embedding, it is difficult to incorporate news texts into TransE and RESCAL. In contrast, the proposed tensor-based framework (TAMURE-KG) can be easily adapted to add news information (TAMURE), thereby connecting emerging relationships effectively.

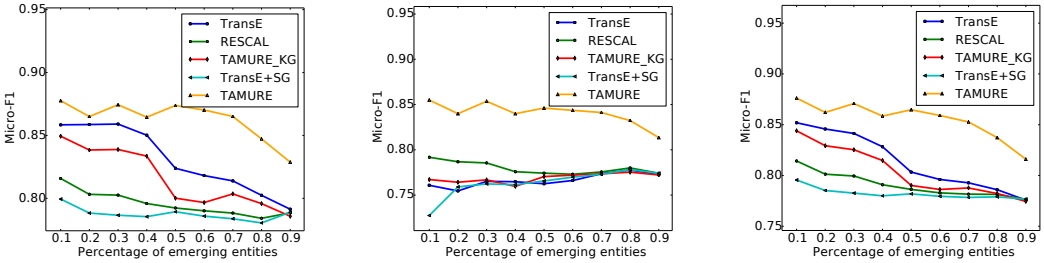
Another observation is that text embedding baselines do not perform well for connecting emerging relationships. The reason lies in the two-step learning of emerging relationships for these models. First the embedding vectors are learned and then the emerging relationships are classified based on the embedding vectors. With two separate steps, it is difficult to connect emerging relationships with the learning of embeddings. Though PTE considers the heterogeneity among entities, relations and words in news texts, it still performs worse than the KG embedding models.

Furthermore, compared with TransE+SG, TAMURE still performs better. TransE+SG combines the KG and news texts together, thereby achieving reasonable performance on emerging relationships with both entities not in the KG. However, TransE+SG embeds the KG and the news separately. The interactions between the relations in KGs and the text descriptions in news are not captured. Hence, it cannot help detect emerging relationships effectively. The proposed TAMURE method in this chapter builds a fourth-order tensor to capture the hidden connections between the KG and the news texts, thereby achieving the best performance.

In summary, with a tensor structure, the proposed TAMURE helps connect emerging relationships from news effectively and outperforms all the eight baseline methods.

5.4.4 Effects of Emerging Entities

As mentioned previously, we randomly select half of the entities in FB15k-237 as new entities to evaluate the effectiveness of TAMURE. In the following, we assess the performance of TAMURE with different percentages of new entities, in the range of 10% to 90%. We select the best five models according to Table XXIII (TransE, RESCAL, TAMURE_KG, TransE+SG



(a) With one emerging entity (b) With two emerging entities (c) With all testing relationships

Figure 20. The performance with different percentages of emerging entities.

and TAMURE) and represent their performance in Figure 20. Due to space limit, we only show the performance on $Micro\text{-}F1$. Similar performance is achieved on the other four metrics.

From Figure 20, we can observe that TAMURE significantly outperforms other baselines regardless of how many percentages of emerging entities we select. In addition, TAMURE can achieve a stable performance when there are less than 70% of emerging entities. With less than 30% of existing entities in the KG, little information can be provided for connecting emerging relationships from news, thereby resulting in a drop of performance for TAMURE. However, TAMURE still achieves the best result compared with other baseline methods since it captures the hidden connections between relations in the KG and text descriptions in the news.

5.4.5 Parameter Analysis

In the following, we analyze the performance of TAMURE with different embedding sizes and epochs.

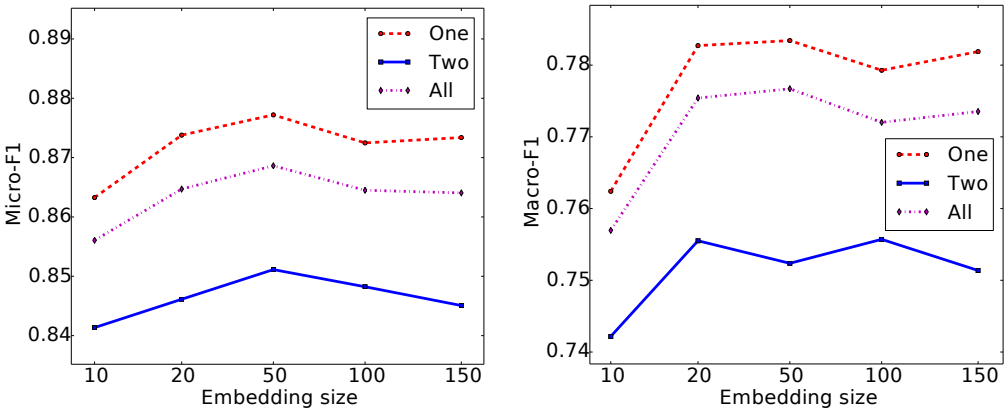


Figure 21. The performance of TAMURE with different embedding sizes.

5.4.5.1 Influence of Embedding Size

We demonstrate the performance of TAMURE with different embedding sizes by fixing the other parameters. In Figure 21, we show the results on $Micro\text{-}F1$ and $Macro\text{-}F1$ metrics, where "One" means the results for emerging relationships with only one entity not in the KG, "Two" is about the relationships with neither entities in the KG, and "All" is about all the emerging relationships. It can be observed that, with a larger embedding size, TAMURE achieves better performance on both $Micro\text{-}F1$ and $Macro\text{-}F1$. However, when the embedding size is larger than 20, the performance of TAMURE becomes more stable with small improvement. Therefore, in our experiment, we set the embedding size as 20. Due to space limit, we do not show the results on $AvgAcc$, $AvgAuc$ and HL metrics, on which similar patterns can be observed.

5.4.5.2 Influence of Epochs

We now investigate the performance of TAMURE with different numbers of epochs. Figure 22 shows the convergence process of the training loss of TAMURE, with different embedding

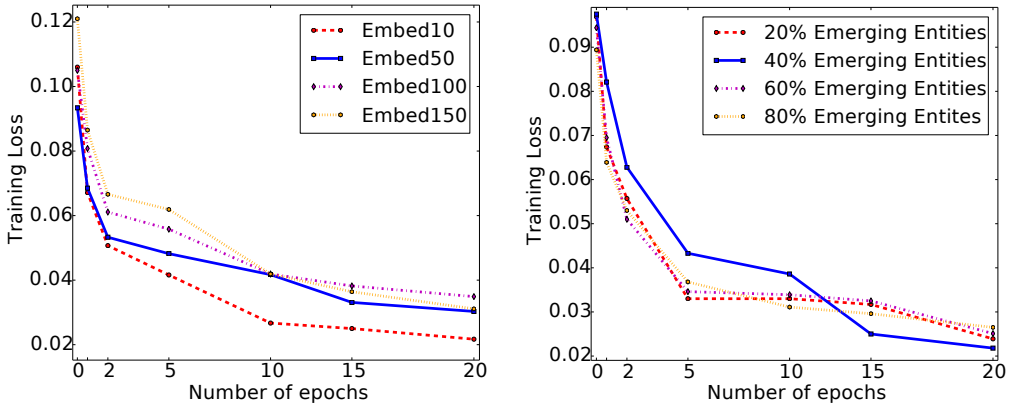


Figure 22. The performance of TAMURE with different numbers of epochs.

sizes and different percentages of emerging entities, respectively. "Embed$k$" on the left of Figure 22 indicates the results with embedding size k. We can observe that the training loss drops quickly at first few epochs and becomes stable after 10 epochs, regardless of the embedding size or the percentage of emerging entities. It demonstrates a fast convergence of the proposed method TAMURE. In our experiment, we set the number of epochs as 20 to achieve a stable performance.

5.5 Related Work

Due to the limited coverage of KGs, the task of KG completion has received a lot of attention (81; 83; 100; 84; 86). Some work learns embedding representations of entities and relations in the KG and use these embeddings to infer missing relations (89; 90; 80; 91; 85). In addition, some studies predict missing relations from a graph view (87; 82; 88; 101). Recently, the research work (101) uses a recursive neural network to create embedded representations of paths learned

from (87; 88). In our work, the emerging relationships have new entities that are not included in the KG. Hence, it is impossible to apply these techniques directly.

Furthermore, some work tries to embed the KG and the texts jointly (102; 3). However, the method in (102) embeds the KG and texts separately. So their indirect inference according to texts cannot help detect emerging relationships effectively. Though the method in (3) aims to connect emerging relationships, it ignores the relation types and focuses on binary relations. In contrast, our proposed TAMURE model handles multi-label relations by building a fourth-order tensor structure.

Recently, there are some work about embedding large-scale texts (57; 96; 94; 97). For example, DeepWalk and LINE are proposed in (96) and (94) to embed texts into a low-dimensional space by constructing a homogeneous word co-occurrence network from texts. Later PTE is proposed in (97) to improve the LINE method by building a heterogeneous text network. These methods focus on embedding every single word in texts but ignoring the semantic relations among words. Therefore, they cannot help connect emerging relationships from news texts effectively.

CHAPTER 6

CONCLUSION

(Part of the chapter was previously published in (1; 2; 3).)

In this thesis, we have explored the information network modeling and mining. Towards this direction, we thoroughly studied four different research problems: unsupervised co-ranking in Q&A sites, learning entity types from search query logs, detecting emerging relations from news and connecting emerging relation types from news to knowledge graphs. The effectiveness of the proposed models and algorithms are evaluated by extensive experiments on various real-world datasets. The contributions of our work are summarized as below:

- First, we study the problem of estimating the quality of questions, answers and users simultaneously in heterogeneous Q&A networks. We analyze real-world Yahoo! Answers datasets to demonstrate the interdependent relationships among questions, answers and users. The NCR framework is then introduced to co-rank different types of objects by capturing their interrelationships in an unsupervised way. Extensive experiments are conducted to evaluate the proposed NCR framework on datasets under different topic categories. The effectiveness of the proposed model shows that interdependent relationships play important roles in ranking objects (questions, answers and users) in a heterogeneous Q&A network. Furthermore, we also implement the NCR framework in a distributed version to test its scalability.

- Second, we study the problem of discovering entity types from search query logs. In order to take advantage of word patterns and user feedbacks (e.g., clicked URLs) from query logs, we construct a bipartite graph to encode entities and the important auxiliary information together. Based on this, the framework ELP is proposed to simultaneously learn types of both entities and auxiliary signals. In order to effectively learn node types, two separate strategies LPA and LPD are proposed and incorporated into ELP. Extensive empirical studies are conducted on real-world search logs to evaluate the effectiveness of the proposed ELP framework.
- Third, we define a new concept of "emerging relations" from news and focus on the problem of discovering such relations. We propose a novel **H**eterogeneous graph **E**mbedding framework for **E**merging **R**elation detection (HEER) that learns a classifier from positive and unlabeled instances by utilizing information from both news and the knowledge graph (KG). We show that by modeling news into a heterogeneous textual graph, the proposed method HEER can detect emerging relations effectively. We further implement HEER in an incremental manner to timely update the KG with the latest detected emerging relations.
- Lastly, we formulate a new task of connecting emerging relation types from news to KGs and propose a novel tensor-based framework to combine KGs and news texts effectively for connecting emerging relationships. With an efficient **T**ext-**A**ware **MU**lti-**RE**lational learning method (TAMURE), the complex interactions among relations, entities and text descriptions are jointly factorized without physically building the tensor. Extensive ex-

periments via TensorFlow demonstrate the effectiveness of the proposed TAMURE model compared with eight state-of-the-art methods on real-world datasets.

APPENDICES

.1 ACM Copyright Letter

"Authors can reuse any portion of their own work in a new work of their own (and no fee is expected) as long as a citation and DOI pointer to the Version of Record in the ACM Digital Library are included.

Contributing complete papers to any edited collection of reprints for which the author is not the editor, requires permission and usually a republication fee.

Authors can include partial or complete papers of their own (and no fee is expected) in a dissertation as long as citations and DOI pointers to the Versions of Record in the ACM Digital Library are included. Authors can use any portion of their own work in presentations and in the classroom (and no fee is expected)." [1]

.2 IEEE Copyright Letter

[1] http://authors.acm.org/main.html

Title: HEER: Heterogeneous graph embedding for emerging relation detection from news

Conference Proceedings: Big Data (Big Data), 2016 IEEE International Conference on

Author: Jingyuan Zhang

Publisher: IEEE

Date: Dec. 2016

Thesis / Dissertation Reuse

The IEEE does not require individuals working on a thesis to obtain a formal reuse license, however, you may print out this statement to be used as a permission grant:

Requirements to be followed when using any portion (e.g., figure, graph, table, or textual material) of an IEEE copyrighted paper in a thesis:

1) In the case of textual material (e.g., using short quotes or referring to the work within these papers) users must give full credit to the original source (author, paper, publication) followed by the IEEE copyright line © 2011 IEEE.
2) In the case of illustrations or tabular material, we require that the copyright line © [Year of original publication] IEEE appear prominently with each reprinted figure and/or table.
3) If a substantial portion of the original paper is to be used, and if you are not the senior author, also obtain the senior author's approval.

Requirements to be followed when using an entire IEEE copyrighted paper in a thesis:

1) The following IEEE copyright/ credit notice should be placed prominently in the references: © [year of original publication] IEEE. Reprinted, with permission, from [author names, paper title, IEEE publication title, and month/year of publication]
2) Only the accepted version of an IEEE copyrighted paper can be used when posting the paper or your thesis on-line.
3) In placing the thesis on the author's university website, please display the following message in a prominent place on the website: In reference to IEEE copyrighted material which is used with permission in this thesis, the IEEE does not endorse any of [university/educational entity's name goes here]'s products or services. Internal or personal use of this material is permitted. If interested in reprinting/republishing IEEE copyrighted material for advertising or promotional purposes or for creating new collective works for resale or redistribution, please go to http://www.ieee.org/publications_standards/publications/rights/rights_link.html to learn how to obtain a License from RightsLink.

If applicable, University Microfilms and/or ProQuest Library, or the Archives of Canada may supply single copies of the dissertation.

CITED LITERATURE

1. Zhang, J., Kong, X., Jie, L., Chang, Y., and Yu, P.: Ncr: A scalable network-based approach to co-ranking in question-and-answer sites. In CIKM, 2014.

2. Zhang, J., Jie, L., Rahman, A., Xie, S., Chang, Y., and Yu, P.: Learning entity types from query logs via graph-based modeling. In CIKM, 2015.

3. Zhang, J., Lu, C., Zhou, M., Xie, S., Chang, Y., and Yu, P.: Heer: Heterogeneous graph embedding for emerging relation detection from news. In IEEE BigData, 2016.

4. Zhang, J., Lu, C., Cao, B., Chang, Y., and Yu, P.: Connecting emerging relationships from news via tensor factorization. In KDD under review, 2017.

5. McGee, M.: Yahoo answers hits 300 million questions, but q&a activity is declining. Search Enginel Land, 2012.

6. Li, B., Jin, T., Lyu, M., King, I., and Mak, B.: Analyzing and predicting question quality in community question answering services. In WWW Companion, 2012.

7. Jeon, J., Croft, W., Lee, J., and Park, S.: A framework to predict the quality of answers with non-textual features. In SIGIR, 2006.

8. Wang, X. J., Tu, X., Feng, D., and Zhang, L.: Ranking community answers by modeling question-answer relationships via analogical reasoning. In SIGIR, 2009.

9. Agichtein, E., Castillo, C., Donato, D., Gionis, A., and Mishne, G.: Finding high-quality content in social media. In WSDM, 2008.

10. Shah, C. and Pomerantz, J.: Evaluating and predicting answer quality in community qa. In SIGIR, 2010.

11. McGee, M.: Yahoo answers hits one billion answers. Search Enginel Land, 2010.

12. Bian, J., Liu, Y., Zhou, D., Agichtein, E., and Zha, H.: Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In WWW, 2009.

13. Si, X., Chang, E. Y., Gyöngyi, Z., and Sun, M.: Confucius and its intelligent disciples: integrating social with search. PVLDB, 3(1-2), 2010.

14. Brin, S. and Page, L.: The anatomy of a large-scale hypertextual web search engine. Computer Networks and ISDN Systems, 30(1), 1998.

15. Anderson, A., Huttenlocher, D., Kleinberg, J., and Leskovec, J.: Discovering value from community activity on focused question answering sites: a case study of stack overflow. In KDD, 2012.

16. Olston, C., Reed, B., Srivastava, U., Kumar, R., and Tomkins, A.: Pig latin: a not-so-foreign language for data processing. In SIGMOD, 2008.

17. Jurczyk, P. and Agichtein, E.: Discovering authorities in question answer communities by using link analysis. In CIKM, 2007.

18. Jarvelin, K. and Kekalainen, J.: Cumulated gain-based evaluation of ir techniques. TOIS, 20(4), 2002.

19. Zhang, J., Ackerman, M., and Adamic, L.: Expertise networks in online communities: structure and algorithms. In WWW, 2007.

20. Kleinberg, J.: Authoritative sources in a hyperlinked environment. JACM, 46(5), 1999.

21. Adamic, L., Zhang, J., Bakshy, E., and Ackerman, M.: Knowledge sharing and yahoo answers: everyone knows something. In WWW, 2008.

22. Preece, J., Nonnecke, B., and Andrews, D.: The top five reasons for lurking: improving community experiences for everyone. Computers in human behavior, 20(2), 2004.

23. Aperjis, C., Huberman, B., and Wu, F.: Human speed-accuracy tradeoffs in search. In HICSS, 2011.

24. Wang, G., Gill, K., Mohanlal, M., Zheng, H., and Zhao, B.: Wisdom in the social crowd: an analysis of quora. In WWW, 2013.

25. Dror, G., Pelleg, D., Rokhlenko, O., and Szpektor, I.: Churn prediction in new users of yahoo! answers. In WWW Companion, 2012.

26. Liu, Q., Agichtein, E., Dror, G., Maarek, Y., and Szpektor, I.: When web search fails, searchers become askers: understanding the transition. In SIGIR, 2012.

27. Liu, Q., Agichtein, E., Dror, G., Gabrilovich, E., Maarek, Y., Pelleg, D., and Szpektor, I.: Predicting web searcher satisfaction with existing community-based answers. In SIGIR, 2011.

28. Nam, K., Ackerman, M., and Adamic, L.: Questions in, knowledge in?: a study of naver's question answering community. In CHI, 2009.

29. Yang, J., Adamic, L., and Ackerman, M.: Crowdsourcing and knowledge sharing: strategic user behavior on taskcn. In EC, 2008.

30. Li, B., Liu, Y., and Agichtein, E.: Cocqa: co-training over questions and answers with an application to predicting question subjectivity orientation. In EMNLP, 2008.

31. Bian, J., Liu, Y., Agichtein, E., and Zha, H.: Finding the right facts in the crowd: factoid question answering over social media. In WWW, 2008.

32. Suryanto, M., Lim, E., Sun, A., and Chiang, R.: Quality-aware collaborative question answering: methods and evaluation. In WSDM, 2009.

33. Sakai, T., Ishikawa, D., Kando, N., Seki, Y., Kuriyama, K., and Lin, C. Y.: Using graded-relevance metrics for evaluating community qa answer selection. In WSDM, 2011.

34. Zhou, D., Orshanskiy, S., Zha, H., and Giles, L.: Co-ranking authors and documents in a heterogeneous network. In ICDM, 2007.

35. Sun, Y., Han, J., Zhao, P., Yin, Z., Cheng, H., and Wu, T.: Rankclus: integrating clustering with ranking for heterogeneous information network analysis. In EDBT, 2009.

36. Yin, X., Han, J., and Yu, P.: Truth discovery with multiple conflicting information providers on the web. TKDE, 20(6), 2008.

37. Wang, G., Xie, S., Liu, B., and Yu, P.: Review graph based online store review spammer detection. In ICDM, 2011.

38. Pound, J., Mika, P., and Zaragoza, H.: Ad-hoc object retrieval in the web of data. In WWW, 2010.

39. Lin, T., Pantel, P., Gamon, M., Kannan, A., and Fuxman, A.: Active objects: Actions for entity-centric search. In Proc. of WWW, pages 589–598, 2012.

40. Banko, M., Cafarella, M., Soderland, S., Broadhead, M., and Etzioni, O.: Open information extraction for the web. In IJCAI, 2007.

41. Chaudhuri, S., Ganti, V., and Xin, D.: Exploiting web search to generate synonyms for entities. In WWW, 2009.

42. Pennacchiotti, M. and Pantel, P.: Entity extraction via ensemble semantics. In EMNLP, 2009.

43. Hoffart, J., Altun, Y., and Weikum, G.: Discovering emerging entities with ambiguous names. In WWW, 2014.

44. Li, Y., Wang, C., Han, F., Han, J., Roth, D., and Yan, X.: Mining evidences for named entity disambiguation. In ACM SIGKDD, 2013.

45. Jain, A. and Pennacchiotti, M.: Open entity extraction from web search query logs. In ACL, 2010.

46. Pantel, P., Lin, T., and Gamon, M.: Mining entity types from query logs via user intent modeling. In ACL, 2012.

47. Blanco, R., Ottaviano, G., and Meij, E.: Fast and space-efficient entity linking in queries. In WSDM, 2015.

48. Alasiry, A., Levene, M., and Poulovassilis, A.: Detecting candidate named entities in search queries. In SIGIR, 2012.

49. Jain, A. and Pennacchiotti, M.: Domain-independent entity extraction from web search query logs. In WWW, 2011.

50. Dalvi, N., Olteanu, M., Raghavan, M., and Bohannon, P.: Deduplicating a places database. In WWW, 2014.

51. Zhou, D., Bousquet, O., Lal, T., Weston, J., and Schölkopf, B.: Learning with local and global consistency. In NIPS, 2004.

52. Zhu, X. and Ghahramani, Z.: Learning from labeled and unlabeled data with label propagation. Technical report, Technical Report CMU-CALD-02-107, 2002.

53. Li, X., Wang, Y., and Acero, A.: Learning query intent from regularized click graphs. In SIGIR, 2008.

54. Wei, C., Liu, Y., Zhang, M., Ma, S., Ru, L., and Zhang, K.: Fighting against web spam: a novel propagation method based on click-through data. In SIGIR, 2012.

55. Ding, C., Li, T., and Wang, D.: Label propagation on k-partite graphs. In ICMLA, 2009.

56. Mikolov, T., Chen, K., Corrado, G., and Dean, J.: Efficient estimation of word representations in vector space. In ICLR Workshop, 2013.

57. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J.: Distributed representations of words and phrases and their compositionality. In NIPS, 2013.

58. Mikolov, T., Yih, W., and Zweig, G.: Linguistic regularities in continuous space word representations. In HLT-NAACL, 2013.

59. Baeza-Yates, R. and Tiberi, A.: Extracting semantic relations from query logs. In ACM SIGKDD, 2007.

60. Grove, A. and Schuurmans, D.: Boosting in the limit: Maximizing the margin of learned ensembles. In AAAI, 1998.

61. Bellare, K., Curino, C., Machanavajihala, A., Mika, P., Rahurkar, M., and Sane, A.: Woo: A scalable and multi-tenant platform for continuous knowledge base synthesis. VLDB Endowment, 6(11), 2013.

62. Paşca, M.: Weakly-supervised discovery of named entities using web search queries. In CIKM, 2007.

63. Jaakkola, M. and Szummer, M.: Partially labeled classification with markov random walks. In NIPS, 2002.

64. Chang, S., Qi, G.-J., Aggarwal, C. C., Zhou, J., Wang, M., and Huang, T. S.: Factorized similarity learning in networks. In ICDM, 2014.

65. Belkin, M., Niyogi, P., and Sindhwani, V.: Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. JMLR, 7, 2006.

66. Chang, S., Han, W., Tang, J., Qi, G.-J., Aggarwal, C. C., and Huang, T. S.: Heterogeneous network embedding via deep architectures. In ACM SIGKDD, 2015.

67. Hakkani-Tür, D., Celikyilmaz, A., Heck, L., and Tür, G.: A weakly-supervised approach for discovering new user intents from search query logs. In INTERSPEECH, 2013.

68. Reisinger, J. and Paşca, M.: Fine-grained class label markup of search queries. In ACL, 2011.

69. Paşca, M. and Durme, B. V.: What you seek is what you get: Extraction of class attributes from query logs. In IJCAI, 2007.

70. Paşca, M.: Attribute extraction from conjectural queries. In COLING, 2012.

71. Li, Y., Hsu, B., and Zhai, C.: Unsupervised identification of synonymous query intent templates for attribute intents. In CIKM, 2013.

72. Limaye, G., Sarawagi, S., and Chakrabarti, S.: Annotating and searching web tables using entities, types and relationships. VLDB Endowment, 3(1-2), 2010.

73. Yu, X., Ma, H., Hsu, B., and Han, J.: On building entity recommender systems using user click log and freebase knowledge. In WSDM, 2014.

74. Hoffmann, R., Zhang, C., Ling, X., Zettlemoyer, L., and Weld, D.: Knowledge-based weak supervision for information extraction of overlapping relations. In ACL-HLT, 2011.

75. Mintz, M., Bills, S., Snow, R., and Jurafsky, D.: Distant supervision for relation extraction without labeled data. In ACL-IJCNLP, 2009.

76. Surdeanu, M., Tibshirani, J., Nallapati, R., and Manning, C.: Multi-instance multi-label learning for relation extraction. In EMNLP, 2012.

77. Zeng, D., Liu, K., Lai, S., Zhou, G., and Zhao, J.: Relation classification via convolutional deep neural network. In COLING, 2014.

78. Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., and Zhang, W.: Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In KDD, 2014.

79. West, R., Gabrilovich, E., Murphy, K., Sun, S., Gupta, R., and Lin, D.: Knowledge base completion via search-based question answering. In WWW, 2014.

80. Chang, K., Yih, W., Yang, B., and Meek, C.: Typed tensor decomposition of knowledge bases for relation extraction. In EMNLP, 2014.

81. García-Durán, A., Bordes, A., and Usunier, N.: Effective blending of two and three-way interactions for modeling multi-relational data. In ECML/PKDD, 2014.

82. Gardner, M., Talukdar, P., Kisiel, B., and Mitchell, T.: Improving learning and inference in a large knowledge-base using latent syntactic cues. In EMNLP, 2013.

83. Gardner, M., Talukdar, P., Krishnamurthy, J., and Mitchell, T.: Incorporating vector space similarity in random walk inference over knowledge bases. In EMNLP, 2014.

84. Nickel, M., Jiang, X., and Tresp, V.: Reducing the rank in relational factorization models by including observable patterns. In NIPS, 2014.

85. Socher, R., Chen, D., Manning, C., and Ng, A.: Reasoning with neural tensor networks for knowledge base completion. In NIPS, 2013.

86. Wang, Z., Zhang, J., Feng, J., and Chen, Z.: Knowledge graph embedding by translating on hyperplanes. In AAAI, 2014.

87. Gardner, M. and Mitchell, T.: Efficient and expressive knowledge base completion using subgraph feature extraction. In EMNLP, 2015.

88. Lao, N., Mitchell, T., and Cohen, W.: Random walk inference and learning in a large scale knowledge base. In EMNLP, 2011.

89. Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O.: Translating embeddings for modeling multi-relational data. In NIPS, 2013.

90. Bordes, A., Weston, J., Collobert, R., and Bengio, Y.: Learning structured embeddings of knowledge bases. In AAAI, 2011.

91. Ji, G., He, S., Xu, L., Liu, K., and Zhao, J.: Knowledge graph embedding via dynamic mapping matrix. In ACL, 2015.

92. Weikum, G., Bedathur, S., and Schenkel, R.: Temporal knowledge for timely intelligence. In Enabling Real-Time Business Intelligence. Springer, 2011.

93. Elkan, C. and Noto, K.: Learning classifiers from only positive and unlabeled data. In KDD, 2008.

94. Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q.: Line: Large-scale information network embedding. In WWW, 2015.

95. Greene, D. and Cunningham, P.: Practical solutions to the problem of diagonal dominance in kernel document clustering. In ICML, 2006.

96. Perozzi, B., Al-Rfou, R., and Skiena, S.: Deepwalk: Online learning of social representations. In KDD, 2014.

97. Tang, J., Qu, M., and Mei, Q.: Pte: Predictive text embedding through large-scale heterogeneous text networks. In KDD, 2015.

98. Angeli, G., Tibshirani, J., Wu, J., and Manning, C.: Combining distant and partial supervision for relation extraction. In EMNLP, 2014.

99. Zeng, D., Liu, K., Chen, Y., and Zhao, J.: Distant supervision for relation extraction via piecewise convolutional neural networks. In EMNLP, 2015.

100. Lin, Y., Liu, Z., Sun, M., Liu, Y., and Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In AAAI, 2015.

101. Neelakantan, A., Roth, B., and McCallum, A.: Compositional vector space models for knowledge base completion. In ACL, 2015.

102. Wang, Z., Zhang, J., Feng, J., and Chen, Z.: Knowledge graph and text jointly embedding. In EMNLP, 2014.

103. Liu, B., Dai, Y., Li, X., Lee, W., and Yu, P.: Building text classifiers using positive and unlabeled examples. In ICDM, 2003.

104. Zhang, J., Yu, P., and Zhou, Z.: Meta-path based multi-network collective link prediction. In KDD, 2014.

105. Hoffart, J., Altun, Y., and Weikum, G.: Discovering emerging entities with ambiguous names. In International World Wide Web Conference, 2014.

106. Kolda, T. and Bader, B.: Tensor decompositions and applications. Society for Industrial and Applied Mathematics review, 2009.

107. Kingma, D. and Ba, J.: Adam: A method for stochastic optimization. In International Conference on Learning Representations, 2015.

108. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 2014.

109. Toutanova, K., Chen, D., Pantel, P., Choudhury, P., and Gamon, M.: Representing text for joint embedding of text and knowledge bases. In Annual Meeting of the Association for Computational Linguistics, 2015.

110. Nickel, M., Tresp, V., and Kriegel, H.: A three-way model for collective learning on multi-relational data. In International Conference on Machine Learning, 2011.

111. Ghamrawi, N. and McCallum, A.: Collective multi-label classification. In Conference on Information and Knowledge Management, 2005.

112. Tsoumakas, G., Katakis, I., and Vlahavas, I.: Mining multi-label data. In Data Mining and Knowledge Discovery Handbook. 2009.

VITA

Name: Jingyuan Zhang

EDUCATION:

B.E. in Software Engineering, Dalian University of Technology, 2009.

M.S. in Computer Science, Dalian University of Technology, 2012.

PUBLICATIONS:

- Jingyuan Zhang, Chun-Ta Lu, Mianwei Zhou, Sihong Xie, Yi Chang, and Philip S. Yu. HEER: Heterogeneous Network Embedding for Emerging Relation Detection from News. In *IEEE International Conference on Big Data (IEEE BigData'16)*, 2016.
- Jingyuan Zhang, Bokai Cao, Sihong Xie, Chun-Ta Lu, Philip S. Yu and Ann B. Ragin. Identifying Connectivity Patterns for Brain Diseases via Multi-side-view Guided Deep Architectures. In *SIAM International Conference on Data Mining (SDM'16)*, 2016.
- Sihong Xie, Qingbo Hu, Weixiang Shao, Jingyuan Zhang, Jing Gao, Wei Fan and Philip S. Yu. Efficient Crowd Expertise Modeling via Mining Crowd Relations and Multi-task Learning. In *SIAM International Conference on Data Mining (SDM'16)*, 2016.
- Bokai Cao, Xiangnan Kong, Jingyuan Zhang, Philip S. Yu and Ann B. Ragin. Identifying HIV-induced subgraph patterns in brain networks with side information. **Brain Informatics**, 2015.

- Sihong Xie, Qingbo Hu, Jingyuan Zhang, Jing Gao, Wei Fan and Philip S. Yu. Robust Crowd Bias Correction via Dual Knowledge Transfer from Multiple Overlapping Sources. In *IEEE International Conference on Big Data (IEEE BigData'15)*, 2015.
- Bokai Cao, Xiangnan Kong, Jingyuan Zhang, Philip S. Yu and Ann B. Ragin. Mining Brain Networks using Multiple Side Views for Neurological Disorder Identification. In *IEEE International Conference on Data Mining (ICDM'15)*, 2015.
- Sihong Xie, Qingbo Hu, Jingyuan Zhang and Philip S. Yu. An Effective and Economic Bi-level Approach to Ranking and Rating Spam Detection. In *IEEE International Conference on Data Science and Advanced Analytics (DSAA'15)*, 2015.
- Jingyuan Zhang, Luo Jie, Altaf Rahman, Sihong Xie, Yi Chang and Philip S. Yu. Learning Entity Types from Query Logs via Graph-Based Modeling. In *Proceedings of ACM International Conference on Information and Knowledge Management (CIKM'15)*, 2015.
- Shuyang Lin, Qingbo Hu, Jingyuan Zhang and Philip S. Yu. Discovering Audience Groups and Group-Specific Influencers. In *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML/PKDD'15)*, 2015.
- Jingyuan Zhang, Xiaoxiao Shi, Xiangnan Kong, Hong-Han Shuai and Philip S. Yu. Discovering Organizational Correlations from Social Media. In *IEEE International Conference on Data Mining (ICDM), the 7th International Workshop on Domain Driven Data Mining (DDDM'14)*, 2014.

- Jingyuan Zhang, Xiangnan Kong, Luo Jie, Yi Chang and Philip S. Yu. NCR: A Scalable Network-Based Approach to Co-Ranking in Question-and-Answer Sites. In *Proceedings of ACM International Conference on Information and Knowledge Management (CIKM'14)*, 2014.
- He Jiang, Jingyuan Zhang, Jifeng Xuan, Zhilei Ren and Yan Hu. A Hybrid ACO Algorithm for the Next Release Problem. In *Proceedings of 2nd International Conference on Software Engineering and Data Mining (SEDM'10)*, 2010.
- Lei Zheng, Jingyuan Zhang, Bokai Cao, and Philip S. Yu. A Novel Ensemble Approach on Regionalized Neural Networks for Brain Disorder Prediction. In *ACM SIGAPP Symposium on Applied Computing (SAC'17)*, 2017.
- Jingyuan Zhang, Chun-Ta Lu, Bokai Cao, Yi Chang, and Philip S. Yu. Connecting Emerging Relationships from News via Tensor Factorization. Under submission.