ENTITY RECOMMENDATION AND SEARCH IN HETEROGENEOUS INFORMATION NETWORKS

BY

XIAO YU

DISSERTATION

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 Urbana-Champaign, 2014

Urbana, Illinois

Doctoral Committee:

Professor Jiawei Han, Chair & Director of Research
Professor ChengXiang Zhai
Professor Thomas S. Huang
Doctor Hao Ma, Microsoft Research

Abstract

With the rapid development of social media and information network-based web services, data mining studies on network analysis have gained increasing attention in recent years. Many early studies focus on homogeneous network mining, with the assumption that the network nodes and links are of the same type (*e.g.*, social networks). However, real-world data in many domains and applications are often multi-typed and interconnected, forming heterogeneous information networks. The objective of my thesis is to study effective and scalable approaches to help users explore and discover useful information and knowledge in heterogeneous information networks. I also aim to advance the principles and methodologies of mining heterogeneous information networks through these studies.

Specifically, I study and focus on entity recommendation and search related problems in heterogeneous information networks. I investigate and propose data mining methodologies to facilitate the construction of entity recommender systems and search engines for heterogeneous networks. In this thesis, I first propose to study entity recommendation problem in heterogeneous information network scope with implicit feedback. Second, I study a real-world large-scale entity recommendation application with commercial search engine user logs and a web-scale entity graph. Third, I combine text information and heterogeneous relationships between entities to study citation prediction and search problem in bibliographical networks. Fourth, I introduce a user-guided entity similarity search framework in information networks to integrate users' guidance into entity search process, which helps alleviate entity similarity ambiguity problem in heterogeneous networks. The methodologies proposed in this thesis are critically important for information exploration in heterogeneous information networks. The principles and theoretical findings in these studies have potential impact in other information network related research fields and can be applied in a wide range of real-world applications.

To my family for all their love.

Acknowledgements

I would like to thank all of the people and agencies who have contributed to this thesis and have provided great support to me during my Ph.D. study.

First I would like to thank my advisor, Professor Jiawei Han, for his supervision and guidance over the past six years. His exceptional expertise in data mining and related areas, his dedication and passion to scientific research, and his kindness and patience have been motivating and inspiring me during my Ph.D. study. He is a great advisor and friend. Without his help and support, this thesis would not have been possible.

Special thanks to my thesis committee members, Professor ChengXiang Zhai, Professor Thomas S. Huang, and Doctor Hao Ma for their invaluable suggestions and comments towards my research and thesis. Professor ChengXiang Zhai has provided detailed feedback and insightful suggestions to many of my research projects, which broadened the vision and improved the quality of these studies. Professor Thomas S. Huang has raised many questions and gave lots of great comments to this thesis, which increased the potential impact of my studies in different domains. Doctor Hao Ma was my mentor during my internship at Microsoft Research. He has been giving me consistent help to my research and thesis as well as career guidance.

Many thanks to my collaborators, including Yizhou Sun, Lu-An Tang, Paul (Bo-June) Hsu, Quanquan Gu, Xiang Ren, Fangbo Tao, Chi Wang, Brandon Norick, Jing Gao, Peixiang Zhao, Zhenhui Li, Xin Jin, Tim Weninger, Bradley Sturt, Urvashi Khandelwal, Mianwei Zhou, Liang Ge, Ang Pan and Tiancheng Mao.

Many thanks to other senior students or past data mining group members including Bolin Ding, Hongbo Deng, Manish Gupta, Ming Ji, Hyung Sul Kim, Xide Lin, Lidan Wang, Tianyi Wu, Dong Xin, Xiaoxin Yin, Zhijun Yin, Bo Zhao and Feida Zhu, who have provided helpful discussions, and great career advice.

Many thank to other data mining group members and visitors, who have built such a great and fun environment for research. I have benefited a lot from discussions and studying with them.

Thanks to all the faculty members and students in the DAIS group for their great help, discussions and feedback, especially Professor Kevin Chang, Professor Marianne Winslett, Rui Li, Yanen Li, Yue Lu, Arash Termehchy, Hongning Wang and Duo Zhang.

I also want to thank Professor Vince Calhoun from University of New Mexico for sparking my initial interest in research. His expertise and dedication motivated me to pursuit a career as a researcher.

Finally and above all, I want to thank my parents and my family for their unconditional love and support. They have been caring, understanding, encouraging and supporting all the time. This thesis is dedicated to them.

The thesis was supported in part by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF- 09-2-0053 (NS-CTA) and W911NF-11-2-0086 (Cyber-Security), the U.S. Army Research Office under Cooperative Agreement No. W911NF-13-1-0193, DTRA, DHS, and U.S. National Science Foundation grants CNS-0931975, IIS-1017362, IIS-1320617, IIS-1354329.

Table of Contents

Chapter 1

Introduction

Real-world data are often multi-typed and interconnected. Such data are often referred as *heterogeneous information networks*. Heterogeneous information networks have wide applications in different domains including social media, e-commerce, biology, medical care, economics and so on. Due to the ubiquity of this data structure, studies regarding mining heterogeneous information networks have been actively engaged from different perspectives.

The goal of this thesis is to study principles and methodologies to facilitate users with their information and knowledge discovery process in heterogeneous information networks. Specifically, I study entity recommendation and search related problems within the scope of heterogeneous information networks. The unique properties of this interconnected data structure bring both challenges and opportunities into these studies. The proposed methodologies and findings of this thesis not only lead to new entity recommendation and search techniques, but also can potentially impact other network related data mining tasks.

In this chapter, I first introduce the motivation and overview of this thesis in Section 1.1, and then introduce the organization of this thesis in Section 1.2.

1.1 Motivation and Overview

We live in the booming age of heterogeneous information networks. This ubiquitous data structure is often built with multi-typed entities and multi-typed relationships. Heterogeneous information networks can be extracted from different data sources or applications. For example, the DBLP network[a] extracted from the computer science bibliographic database contains papers, authors, publication venues, and text information; the IMDb network[b] extracted from the Internet movie

[a] http://www.informatik.uni-trier.de/~ley/db/
[b] http://www.imdb.com

database contains movies, actors/actresses, directors and reviews; and the social media networks often contain users, pages, groups, events and other entities. These heterogeneous networks are filled with information, semantics and knowledge, which attracts the attention of researchers to study this data model from different perspectives.

Besides ubiquity, heterogeneous information networks are often large-scale. Considering the size of popular social media services or the size of multi-domain knowledge graphs like Freebase[c] and DBpedia[d], users may feel it is difficult to view the content of these networks. New methodologies are urgently needed to help users explore heterogeneous networks, find desired information and discover interesting knowledge. To achieve this goal, we can either actively look for content in information networks which may interest users based on their past behaviors, or we can answer queries composed by users to express their information needs. The first technique is commonly known as recommendation, while the second method is known as search. Following this paradigm, I study entity recommendation and search in heterogeneous information networks to facilitate users' information and knowledge discovery process.

Entity recommendation in heterogeneous information networks resembles previous studies in the category of network-based hybrid recommendation. Most of the previous studies in this category only consider a single relationship type, such as friendships in a social network, which limits the applications of these studies to homogeneous networks. In this thesis, I examine the entity recommendation problem in a heterogeneous information network environment. Different types of relationships, rich semantics and various attributes from the networks not only can be used to improve the recommendation quality, but also allow us to build recommender systems which can handle the data sparsity issue, recommend entities from other domains, and explain recommendation results with network information.

Entity search problems in heterogeneous information networks are related to traditional graph-based search systems. Previous studies mostly consider homogeneous graphs with labels, and these systems usually look for subgraphs which can match or partially match the given queries. Compared with search problems in homogeneous graphs, many different and novel search problems can be proposed in heterogeneous networks. In this thesis, I focus on relevant entity search in

[c] http://www.freebase.com
[d] http://www.dbpedia.org

heterogeneous information networks. Chapter 5 studies citation prediction problem and builds a citation search system over a bibliographic heterogeneous network. Chapter 6 introduces the similarity ambiguity problem in heterogeneous networks and proposes a two-step entity similarity search framework to answer entity similarity queries from users.

I will introduce the details of these studies along with the proposed methodologies, empirical studies and important findings in the rest of this thesis.

1.2 Thesis Organization

The first chapter introduces the problems of entity recommendation and search in heterogeneous information networks. Chapter 2 presents the preliminaries and definitions used in the later chapters. After the introduction and preliminaries, two chapters present the findings and methodologies regarding entity recommendation and two chapters introduce the studies and proposed techniques along the line of entity search.

The major contents of these studies are summarized as follows:

- **Chapter 3: Personalized Recommendation in Heterogeneous Networks [73].** We study the entity recommendation problem in heterogeneous information networks with implicit feedback. Specifically, we propose to combine heterogeneous relationship information for each user differently and aim to provide high-quality personalized recommendation results using user implicit feedback data and personalized recommendation models. In order to take full advantage of the relationship heterogeneity in information networks, we first introduce meta-path-based latent features to represent the connectivity between users and items along different types of paths. We then define recommendation models at both global and personalized levels and use Bayesian ranking optimization techniques to estimate the proposed models.

- **Chapter 4: Recommendation with User Log and Freebase [70].** Due to their commercial value, *search engines* and *recommender systems* have become two popular research topics in both industry and academia over the past decade. Although these two fields have been actively and extensively studied separately, researchers are beginning to realize the im-

portance of the scenarios at their intersection: providing an integrated search and information discovery user experience. In this Chapter, we study a novel application, *i.e.*, personalized entity recommendation for search engine users, by utilizing user click log and the knowledge extracted from Freebase. To better bridge the gap between search engines and recommender systems, we first discuss important heuristics and features of the datasets. We then propose a generic, robust, and time-aware personalized recommendation framework to utilize these heuristics and features at different granularity levels. Using movie recommendation as a case study, with user click log dataset collected from a widely used commercial search engine, we demonstrate the effectiveness of our proposed framework over other popular and state-of-the-art recommendation techniques.

- **Chapter 5: Citation Prediction in Bibliographic Networks [69].** To reveal information hiding in link space of bibliographical networks, link analysis has been studied from different perspectives in recent years. In this work, we address a novel problem namely citation prediction, that is: given information about authors, topics, target publication venues as well as time of a certain research paper (the query paper), finding and predicting the citation relationship between the query paper and a set of previous papers. Considering the gigantic size of relevant papers, the loosely connected citation network structure as well as the highly skewed citation relation distribution, citation prediction is more challenging than other link prediction problems studied before. By building a meta-path-based prediction model on a topic discriminative search space, we here propose a two-phase citation probability learning approach, in order to predict citation relationship effectively and efficiently. Given this problem definition, one can notice that this study can be applied to build reference search engines for scientific researchers. Experiments are performed on a real-world dataset with comprehensive measurements, which demonstrate that our framework has substantial advantages over commonly used link prediction approaches in predicting citation relations in bibliographical networks.

- **Chapter 6: User-Guided Entity Similarity Search [75].** With the emergence of web-based social and information applications, entity similarity search in information networks, aiming to find entities with high similarity to a given query entity, has gained wide attention.

However, due to the diverse semantic meanings in heterogeneous information networks, similarity measurement can be ambiguous without context. In this work, we investigate entity similarity search and the resulting ambiguity problem in heterogeneous information networks. We propose to use a meta-path-based ranking model ensemble to represent various semantic meanings for similarity queries, and exploit user-guidance to help the framework understand the query, and generate final search results with related ranking models.

After studies on entity recommendation and search are introduced, Chapter 7 presents the related works of this thesis and Chapter 8 summaries the contributions, concludes this thesis and proposes new research directions as future work.

Chapter 2

Background and Preliminaries

We discuss the background and preliminaries of heterogeneous information network analysis and mining in this chapter.

2.1 Information Network

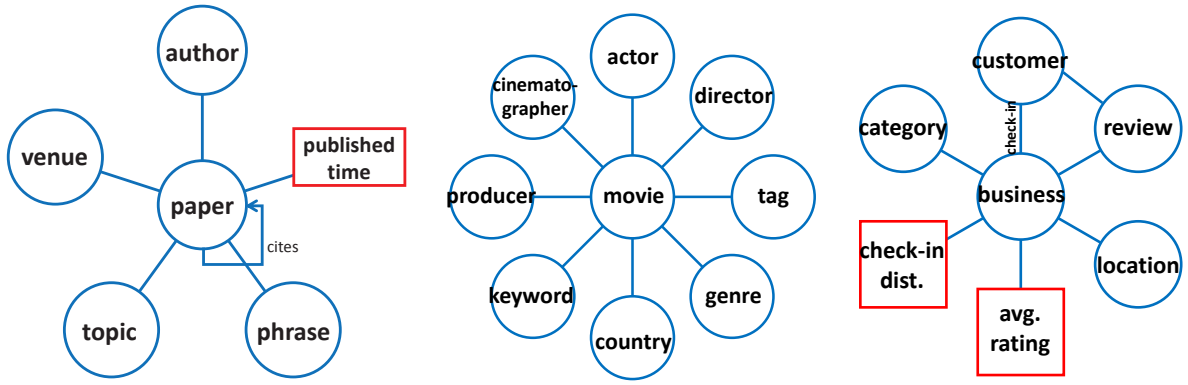
A heterogeneous information network is a directed graph, which contains multiple types of entities and/or links. We first introduce the definitions of information network and network schema as defined in [58].

Definition 2.1 (Information Network) *An information network is defined as a directed graph $G = (V, E)$ with an entity type mapping function $\phi : V \to \mathcal{A}$ and a link type mapping function $\psi : E \to \mathcal{R}$, where each entity $v \in V$ belongs to one particular entity type $\phi(v) \in \mathcal{A}$, and each link $e \in E$ belongs to a particular relation type $\psi(e) \in \mathcal{R}$.*

When the types of entities $|\mathcal{A}| > 1$ and the types of links $|\mathcal{R}| > 1$, the network is called *heterogeneous information network*; otherwise it is a *homogeneous information network*.

In this information network definition, we specify both the network structure and types for entities and links. Following this definition, one can notice that types for two entities associated to one link can be different. Without loss of generality, we denote the relation associated to a link as $R(\phi(v), \phi(v')) = \psi(e)$, where v and v' are the two entities associated with link e. We use $dom(R) = \phi(v)$ to denote the domain of R, and $range(R) = \phi(v')$ as the range. We use R^{-1} to denote the inverse relation of R, so $dom(R) = range(R^{-1})$ and $range(R) = dom(R^{-1})$.

Definition 2.2 (Network Schema) *The network schema is a meta template of a heterogeneous network $G = (V, E)$ with the entity type mapping $\phi : V \to \mathcal{A}$ and the link mapping $\psi : E \to \mathcal{R}$. It*



(a) DBLP Network Schema (b) IMDb Network Schema (c) Yelp Network Schema

Figure 2.1: Information network schemas: circles represent entity types while rectangles represent attribute types

is a directed graph defined over entity types $\mathcal{A}$, with links as relations from $\mathcal{R}$. We denote network schema using $T_G = (\mathcal{A}, \mathcal{R})$.

The definition of network schema is similar to the ER (Entity-Relationship) model in database systems. It serves as a template for a concrete network, and defines the rules of how entities exist and how links should be created. Examples of heterogeneous information networks and their schemas can be found as follows:

Example 2.1 *DBLP*[a] *(Digital Bibliography & Library Project) is a computer science bibliographic database, which can be described as a heterogeneous information network. It contains four different types of entities (papers, venues, authors and terms). Relationships exist between papers and authors, papers and venues, papers and terms as well as within paper entities, representing citation relations (Figure 2.1(a)).*

Example 2.2 *IMDb*[b] *(Internet Movie Database) is an online database of information related to films, television programs, and video games. It contains various types of entities including actors, directors, genre, and so on. Relationships exist between movies and other entities, e.g., movies and actors, movies and directors, movies and genres (Figure 2.1(b)).*

Example 2.3 *Yelp*[c] *is website which provides directory services and allows users to review local*

[a] http://www.informatik.uni-trier.de/~ley/db/
[b] http://www.imdb.com
[c] http://www.yelp.com

businesses with social networking features. It contains entities like businesses, categories, reviews, locations as well as relationships between businesses and users, businesses and reviews, and businesses and categories (Figure 2.1(c)).

2.2 Meta-Path-Based Entity Similarity

In a heterogeneous information network schema, two entity types can be connected via different paths. In order to distinguish paths in the network schema from path instances in a concrete network, we introduce the definition of Meta-Path as proposed in [59] as follows:

Definition 2.3 (Meta-Path) *A meta-path* $\mathcal{P} = A_0 \xrightarrow{R_1} A_1 \xrightarrow{R_2} ... \xrightarrow{R_l} A_l$ *is a path defined on the graph of network schema* $T_G = (\mathcal{A}, \mathcal{R})$*, which defines a new composite relation* $R_1 R_2 ... R_l$ *between type* A_0 *and* A_l*, where* $A_i \in \mathcal{A}$ *and* $R_i \in \mathcal{R}$ *for* $i = 0, \ldots, l$*,* $A_0 = dom(R_1)$*,* $A_l = range(R_l)$ *and* $A_i = range(R_i) = dom(R_{i+1})$ *for* $i = 1, \ldots, l-1$*.*

Notice that a meta-path represents a new composite relation over A_0 and A_l, and obviously we have $dom(\mathcal{P}) = A_0$ and $range(\mathcal{P}) = A_l$. By examining the network schema of DBLP (Figure 2.1(a)), one can enumerate meta-paths between two entities with a length constraint. For example, meta-paths between paper entity types with length less than or equal to 2 can are listed as follows:

$$\mathcal{P}_1 : paper \xrightarrow{cites} paper$$
$$\mathcal{P}_2 : paper \xrightarrow{cites} paper \xrightarrow{cites} paper$$
$$\mathcal{P}_3 : paper \xrightarrow{uses} term \xrightarrow{used} paper$$
$$\mathcal{P}_4 : paper \xrightarrow{written-by} author \xrightarrow{writes} paper$$
$$\mathcal{P}_5 : paper \xrightarrow{published-by} venue \xrightarrow{publishes} paper$$

Different meta-paths link entities through different relationships, and thus could convey various relationship meanings. Comparing $\mathcal{P}_3$ and $\mathcal{P}_4$ in the previous example, one can see that two paper entities which are connected following $\mathcal{P}_3$ share common terms, while two paper entities are connected with $\mathcal{P}_4$ share common authors.

Additionally, attributes of meta-paths or similarity measurements defined along meta-paths can be used to measure similarity between entities quantitatively. For meta-path $\mathcal{P}_3$, if the number of terms shared by paper a and paper b is larger than the number of terms shared by paper b and paper c, one may claim that paper a is more similar to b than c. Although using term count as a meta-path similarity measurement can lead to biased results, this simple example demonstrates that meta-paths can be used to measure similarity from a specific semantic perspective – sharing terms / topics. In the rest of this paper, we omit the relation name and use the abbreviation of entity types to represent a meta-path, if this causes no ambiguity. For example, $\mathcal{P}_1$ will be represented as $P \to P$, and $\mathcal{P}_3$ will be written as $P - T - P$.

Many existing similarity measurements which used to be defined over network can be applied along a specific meta-path. We call such measurements meta-path compatible similarity measurements, some of which are listed as follows:

- Count: the number of path instances between entity q and entity r following meta-path $\mathcal{P}$:

$$count(q, r) = |p : dom(p) = q, range(p) = r, p \in \mathcal{P}|$$

- Personalized PageRank score [13]. Personalized PageRank can be viewed as the random walk process on path instances following a given meta-path.

- SimRank score [30]. SimRank can be viewed as the weighted combination of pairwise random walk scores on path instances following a given meta-path.

- PathSim score [59]. PathSim is a newly proposed method which is used to calculate similarity between peers. It is defined as a normalized version of count of path instances between entities following a given meta-path.

Due to the definition differences, these meta-path compatible measurements have different properties even along the same meta-path. For instance, as discussed in [59], personalized PageRank favors "important" entities with a large number of links, while PathSim can find "peer" entities. In the rest of this thesis, we use $\mathbf{P}$ to represent a set of meta-paths and $\mathbf{M}$ a set of meta-path-based entity similarity measurements.

Chapter 3

Personalized Recommendation in Heterogeneous Networks

3.1 Overview

Entity recommendation, as one of the most effective and widely used information filtering and discovery methods, has been actively studied in the past decade in both industry and academia. Among different recommendation techniques, hybrid recommender systems, which combine user feedback data (explicit ratings or implicit user log) and additional information of users or items, can achieve better recommendation results in certain scenarios, based on recent studies [21] [49].

In many recommendation applications, additional information regarding users and items can be obtained, *e.g.*, user demographic attributes, product specifications, or user social network information. Moreover, the entity recommendation problem often exists in a heterogeneous information network environment with different types of attributes and relationships of users, items, and other entities available. An illustration of a movie recommendation problem in a heterogeneous information network can be found in Figure 3.1. In this example, besides users and movies and the user-movie interaction relationships, other types of entities can be found and linked to the movie recommendation problem, such as actors, directors, and genres. Different types of relationships between users (*e.g.*, friendship) and entities (*e.g.*, director directed a movie) can be potentially utilized in hybrid recommender systems.

Previous studies suggest that by utilizing additional user or item relationship information, the quality of the recommender systems can be improved. Our study falls in the category of such hybrid recommender systems. The difference between our work and other link-based hybrid methods is that most previous studies only utilize a single type of relationship, *e.g.*, trust relationship [29], friend relationship [46], or user membership [76]. We propose to study entity recommendation problem in the aforementioned heterogeneous network environment, aiming to take advantage of

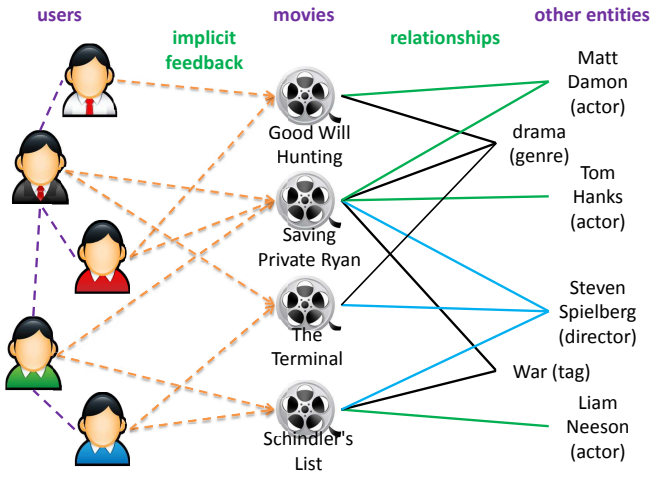


Figure 3.1: A heterogeneous information network snippet with users, movies, actors, directors and tags as entities and the corresponding relationships between these entities

different types of relationship information at the same time.

Previous studies [72] [74] on link-based hybrid recommender systems apply the same recommendation models to all the users when recommending items. They rely on the personal ratings or user feedback data to achieve recommendation personalization. However, such approaches cannot fully distinguish user interests and preferences, and thus may lead to unsatisfying results. For example, Alice and Bob watched the movie "Pacific Rim". Alice watched this movie because she likes robot / monster stories (genre and story of the movie) while Bob watched this movie because his friends watched the same movie. If we apply the same recommendation model (*e.g.*, social network-based collaborative filtering) to both users without understanding or differentiating their motives and interests, the recommendation results may not satisfy the information discovery needs of different users.

In this study, we introduce a novel entity recommendation framework in heterogeneous information networks using implicit feedback data. We combine user feedback with different types of entity relationships in a collaborative filtering way. We personalize recommendation results by both considering personal user implicit feedback data and building personalized recommendation models for different users.

To take advantage of the relationship heterogeneity of the information network, we first diffuse the observed user implicit feedback along different meta-paths to generate possible recommendation

candidates, under the corresponding user interest semantic assumptions. We apply matrix factorization techniques on the diffused user preferences to calculate latent representations for users and items accordingly. We then combine these latent features and define a global recommendation model. To further distinguish user interests, we propose to build recommendation models at personalized level, *i.e.*, *we build different entity recommendation models for different users*. We adopt a Bayesian ranking optimization technique for model estimation. Empirical studies in two real world datasets, IMDb-MovieLens-100K and Yelp, have shown that the proposed recommendation models outperform several state-of-the-art implicit feedback recommendation systems.

The contributions of this study are summarized as follows:

1. We study personalized entity recommendation with implicit user feedback in heterogeneous information networks.

2. To take advantage of the relationship heterogeneity, we propose to diffuse user preferences along different meta-paths in information networks to generate latent features for users and items.

3. The proposed framework generates personalized recommendation models for different users effectively and efficiently.

4. Empirical studies in two real-world datasets, IMDb-MovieLens-100K and Yelp, demonstrate the power of our methodology.

The remainder of this chapter is organized as follows. The preliminaries are introduced in Section 3.2. The meta-path-based latent features and the global recommendation model are presented in Section 3.3. We propose the personalized recommendation framework in Section 3.4 and parameter estimation techniques in 3.5. Experiments and analysis are in Section 3.6. Finally, we conclude the study in Section 3.7.

3.2 Background and Preliminaries

In this section, we present the background and preliminaries of this study. Detailed problem definition is included at the end of this section.

Table 3.1: Notations

Notation	Description
u, e	user, item (or entity)
R	implicit feedback matrix
G, G_T	heterogeneous information network and schema
A, R	entity type and relationship type
$\mathcal{P}$, p	meta-path and path
$\tilde{R}^{(q)}$	diffused user preferences along the q-th meta-path
U, V	low rank representations of users and items
C	user clusters
Θ, $\Theta^{\{\cdot\}}$	global and hidden local model parameters

3.2.1 Binary User Feedback

With m users $\mathcal{U} = \{u_1, ..., u_m\}$ and n items $\mathcal{I} = \{e_1, ..., e_n\}$, we define the user implicit feedback matrix $R \in \mathbb{R}^{m \times n}$ as follows:

$$R_{ij} = \begin{cases} 1, & \text{if } (u_i, e_j) \text{ interaction is observed;} \\ 0, & \text{otherwise.} \end{cases}$$

Notice that the value 1 in R represents interactions between users and items, *e.g.*, users watched a movie or users browsed a restaurant website. The value 1 in the implicit feedback data does not mean that users like the items. A user buys a movie ticket because she or he is interested in the movie but this user might dislike the movie after watching it. Similarly the value 0 in R does not mean that the users dislike the items, but are a mixture of negative feedback (the users are not interested in the items) and unobserved interactions (the users are not aware of these items for now). Several previous studies have additional assumptions about the implicit feedback data, *e.g.*, user-item interaction frequency, or the dwelling time of each interaction. Not to digress from the purpose of this study, we use binary user feedback in its original form as defined above. However, additional information as mentioned can be added into the factorization process of the proposed models accordingly.

3.2.2 Matrix Factorization for Implicit Feedback

Matrix factorization techniques have been used to interpret implicit user feedback in previous studies [15] [27], by learning the low-rank matrix representations for users and items. More specifically,

factorization methods seek to approximate the implicit feedback matrix R with the product of the low-rank matrices as follows:

$$R \approx UV^T \tag{3.1}$$

where $U \in \mathbb{R}^{m \times d}$ are the latent features representing users and $V \in \mathbb{R}^{n \times d}$ are the latent features representing items, with $d < \min(n, m)$.

The recommendation score between u_i and e_j can be computed with the estimated low-rank matrices as $r(u_i, e_j) = U_i V_j^T$, where U_i denotes the i-th row of the matrix U and V_j denotes the j-th row of V. By sorting items with their recommendation scores, we can return the top-k items which u_i has not interacted with before as the recommendation results.

To solve Equation (3.1), the non-negative factorization techniques (NMF) discussed in [15] can be directly employed. More advanced methods [27] [21] [46] have been studied recently to incorporate additional information in order to further improve the performance.

In our proposed recommendation models, when defining user item features, we rely on matrix factorization methods to derive low-rank representations of users and items under different semantics. One should notice that our proposed models are orthogonal to factorization techniques, *i.e.*, one can extend the proposed models easily with advanced factorization techniques. In this study, aiming to propose a generic recommendation framework, we use the basic NMF method in [15] when defining features and models. By utilizing advanced factorization methods, the performance of the our methods can be improved accordingly due to the aforementioned orthogonality.

3.2.3 Problem Definition

We define the recommendation problem which we study in this study as follows:

Definition 3.1 (Problem Definition) *Given a heterogeneous information network G with user implicit feedback R, for a user u_i, we aim to build personalized recommendation model for u_i, and recommend a ranked list of items that are of interest to u_i accordingly.*

Notations which are used in the rest of the paper can be found in Table 3.1.

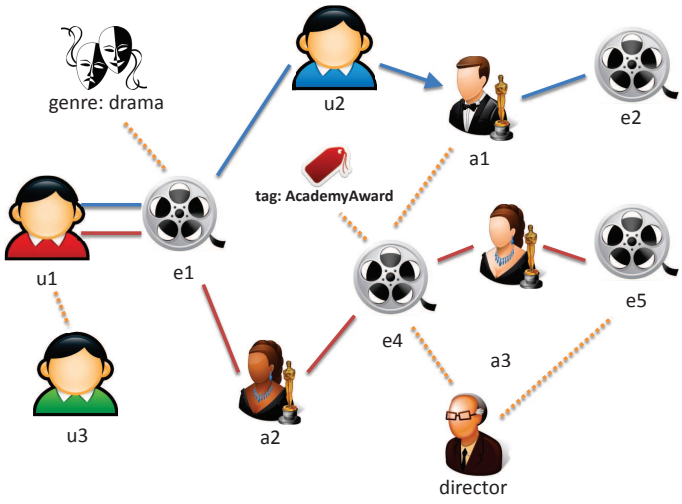


Figure 3.2: User preference discussion along different meta-paths (Example 1)

3.3 Meta-Path-Based Latent Features

In this section, aiming to utilize the rich yet under-discovered information network, we present a user preference diffusion-based feature generation approach, which combines user implicit feedback and heterogeneous entity relationships. We then define a recommendation function with the latent features at a global level at the end of this section. Notice that we use the term "global" to represent the process of applying the same recommendation model to all users. We still utilize personal user implicit feedback data during recommendation with the global recommendation model. We introduce personalized recommendation models in Section 3.4. We discuss the learning algorithms for the recommendation models in Section 3.5.

3.3.1 Meta-Path

From the information network point of view, the entity recommendation problem is to seek certain connectivity between users and items. In heterogeneous information networks, two entities can be connected via different paths (see Figure 3.2 for examples). Due to the multiplicity of entity and relationship types in the information networks, these paths may contain different entity types, relationship types in different orders and they can have various lengths. In order to describe path types in heterogeneous information network, we use the definition of *meta-path* introduced in Chapter 2.2.

We use p to denote the paths in information networks and $\mathcal{P}$ to denote meta-paths. Based on the definition, one can notice that meta-paths are the types for paths in information networks. Previous studies suggest that meta-paths can be used to facilitate entity similarity or proximity measurement and similarity semantic disambiguation [59] [69]. We use the following example to demonstrate the intuition of utilizing meta-paths in entity recommendation problem.

Example 3.1 (Different Meta-Paths in IMDb) *With the graph schema of IMDb defined in Figure 2.1(b), we can derive a number of meta-paths which connect users with movies. We show two possible meta-paths as follows:*

- $\mathcal{P}_1$**:** $user \xrightarrow{Viewed} movie \xrightarrow{Viewed^{-1}} user \xrightarrow{Follows} actor \xrightarrow{StarredIn} movie$
- $\mathcal{P}_2$**:** $user \xrightarrow{Viewed} movie \xrightarrow{StarredIn^{-1}} actor \xrightarrow{StarredIn} movie \xrightarrow{StarredIn^{-1}} actor \xrightarrow{StarredIn} movie$

We give two path examples of the two meta-paths in Figure 3.2 (the solid blue line represents meta-path $\mathcal{P}_1$ and the solid red line represents meta-path $\mathcal{P}_2$). These two meta-paths connect users and movies under different semantic assumptions. $\mathcal{P}_1$ explores the social relationships of other users who watched movies in common with the target user, while $\mathcal{P}_2$ utilizes movie-actor links to build relationships between users and movies. By measuring proximity between users and movies along different meta-paths, we may be able to make movie recommendations to users from different semantic perspectives in information networks.

When representing longer meta-paths, relationship types can be ignored when doing so does not cause ambiguity. The recursive parts of the meta-paths can be compressed with exponentiation notation. For example, we use **user** $-($ **movie** $-$ **actor** $-$ **movie**$)^2$ to represent meta-path $\mathcal{P}_2$ in the above example.

3.3.2 User Preference Diffusion

With the implicit user feedback data defined in Section 3.2 and meta-path defined above, we introduce the the user preference diffusion process along meta-paths. As mentioned before, implicit feedback represents user item interactions. the value 1 in implicit feedback indicates that users are more interested in the corresponding items than the rest of the items. We use the term *user*

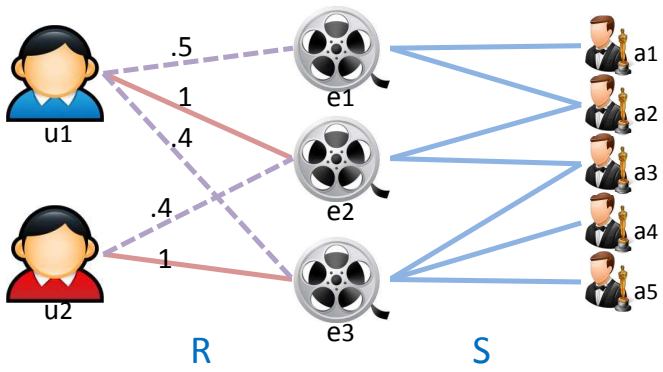


Figure 3.3: User preference diffusion score calculation (Example 2). The solid red links represent observed user implicit feedback while the purple doted links represent diffused user preferences.

preference to represent the user interests in implicit feedback data. Intuitively, if we can understand the semantic meanings of user preferences and find similar items to the ones that the users were interested in, following the discovered semantics, we can make entity recommendations to these users accordingly.

Based on this observation and the problem definition presented in Section 3.2, in this study, we focus on meta-paths in the format of $\textbf{user} - \textbf{item} - * - \textbf{item}$ when building recommendation models. The intuition is we want to diffuse the observed users preferences in implicit feedback data along different meta-paths so that users can be connected with other items. By defining a user preference diffusion score between the target user and all possible items along different meta-paths, we can now measure the possibility of an unobserved user-item interaction in the information network under different semantic assumptions.

Given a meta-path $\mathcal{P} = R_1 R_2 ... R_k$ with $dom(\mathcal{P}) = \textbf{user}$ and $range(\mathcal{P}) = \textbf{item}$, let $\mathcal{P}' = R_2 ... R_k$ with $dom(\mathcal{P}') = \textbf{item}$ and $range(\mathcal{P}') = range(\mathcal{P}) = \textbf{item}$. We define the user preference diffusion score between user i and item j along $\mathcal{P}$ by extending PathSim measurement proposed in [59] as follows:

$$
\begin{aligned}
& s(u_i, e_j|\mathcal{P}) \qquad (3.2) \\
= & \sum_{e \in \mathcal{I}} \frac{2 \times R_{u_i,e} \times |\{p_{e \rightsquigarrow e_j} : p_{e \rightsquigarrow e_j} \in \mathcal{P}'\}|}{|\{p_{e \rightsquigarrow e} : p_{e \rightsquigarrow e} \in \mathcal{P}'\}| + |\{p_{e_j \rightsquigarrow e_j} : p_{e_j \rightsquigarrow e_j} \in \mathcal{P}'\}|}
\end{aligned}
$$

where $p_{e \rightsquigarrow e_j}$ is a path between e and e_j, $p_{e \rightsquigarrow e}$ is a path between e and e, and $p_{e_j \rightsquigarrow e_j}$ is a path between e_j and e_j.

The user preference diffusion score between user u_i and item e_j contains two parts: (1) the observed user-item interactions associated with u_i, and (2) the connectivity between the items that u_i is interested in and potential items of interest, which are represented by e_j in Equation (3.2). Notice the connectivity between items is defined as the number of paths between these items following meta-path $\mathcal{P}$ and normalized by the visibility of the items so the diffusion score does not overly favor popular items. We demonstrate the user preference diffusion process with a toy example as in Figure 3.3.

Example 3.2 (User Preference Diffusion Score) *In this toy example, we use a small information network which contains two users (u_1 and u_2), three movies (e_1, e_2 and e_3), and 5 actors (a_1, ..., a_5). These entities are interconnected as shown in Figure 3.3 (The solid red links represent observed user implicit feedback while the purple doted links represent the diffused user preferences). We use* ***user*** $-$ ***movie*** $-$ ***actor*** $-$ ***movie*** *as meta-path $\mathcal{P}$ when calculating the diffusion score. Based on implicit feedback data R, we know that u_1 watched movie e_2. Based on the information network structure, there is* 1 *path between e_1 and e_2 following the aforementioned meta-path,* 2 *paths between e_1 and e_1 and* 2 *paths between e_2 and e_2. By plugging the implicit feedback data and the numbers of paths described above into Equation* (3.2)*, we can get that the user preference diffusion score under meta-path $\mathcal{P}$ from u_1 to e_1 is* 0.5*. Other diffusion scores in this example can be calculated accordingly.*

By measuring the diffusion scores between all users and all items along meta-path $\mathcal{P}$, we can generate a diffused user preference matrix $\tilde{R} \in \mathbb{R}^{m \times n}$. $\tilde{R}_i$ represents the possible preferences of user u_i if he or she explores the network for new content following meta-path $\mathcal{P}$ (u_i watches movies with certain genres or u_i watches movies from certain directors). By repeating this process, with L different meta-paths, we can calculate L different diffused user preference matrices accordingly. We denote these user preference matrices as $\tilde{R}^{(1)}, \tilde{R}^{(2)}, ..., \tilde{R}^{(L)}$.

This process propagates user preferences along different meta-paths in the heterogeneous information networks, and it mimics users' information discovery process. The diffusion score indicates

the possibility of certain user-item interaction under certain meta-path semantic. The diffusion scores can be used to define recommendation models.

The length of the meta-path of each diffused preference matrix is positively correlated with computational time. The longer the meta-path we utilize, the longer time it takes to generate the diffused preference matrix. In this study, we compute all the features off-line, although partial materialization with on-line computation is also possible. We also found out that, the length of the meta-path does not indicate the quality of the feature. In our empirical study, the IM100K dataset, which is relatively denser, benefits less from the information network structure. Shorter meta-path carries more weights in recommendation model for this dataset. On the other hand, Yelp dataset, which is relatively sparser, benefits more from the information network, and longer meta-paths carry more weights in the recommendation model.

3.3.3 Global Recommendation Model

We denote $\tilde{R}^{(q)}$ as the diffused user preference matrix along the q-th meta-path. Following the intuition and principle of matrix factorization-based recommendation methods, we can derive low-rank user and item matrices from each diffused preference matrix accordingly. These low-rank matrices are the latent representations of users and items, under the semantic meaning of the corresponding meta-path. With low-rank matrix factorization technique, we can factorize the diffused matrix $\tilde{R}^{(q)}$ as follows:

$$\begin{aligned} (\hat{U}^{(q)}, \hat{V}^{(q)}) &= \text{argmin}_{U,V} \|\tilde{R}^{(q)} - UV^T\|_F^2 \\ \text{s.t.} \quad & U \geq 0, \quad V \geq 0, \end{aligned} \tag{3.3}$$

where $\hat{U}^{(q)} \in \mathbb{R}^{m \times d}$ represents users and $\hat{V}^{(q)} \in \mathbb{R}^{n \times d}$ represents items, with $d < \min(n, m)$. $\hat{U}_i^{(q)}$ is the latent feature for user u_i along the q-th meta-path and $\hat{V}_j^{(q)}$ is the latent feature for e_j along the q-th meta-path respectively.

As discussed in Section 3.2, we apply the generic NMF method to solve Equation (3.3), although more advanced factorization techniques can be used instead. By repeating the above process for all L diffused user preference matrices, we can now generate L pairs of representations of users

and items ($\hat{U}^{(1)}$, $\hat{V}^{(1)}$), ..., ($\hat{U}^{(L)}$, $\hat{V}^{(L)}$). Each low-rank feature pair represents users and items under a specific relationship semantic due to the user preference diffusion process. When defining recommendation models with these latent-features, different feature pairs may have different importance. For example, users are more likely to follow certain actors when choosing movies rather than consider which movie studios these movies are made. With this observation, following [74], we define a global recommendation model as follows:

$$\hat{r}(u_i, e_j) = \sum_{q=1}^{L} \theta_q \cdot \hat{U}_i^{(q)} \hat{V}_j^{(q)T} \tag{3.4}$$

where θ_q is the weight for the q-th user and item low-rank representation pair. Based on the non-negative property of the features, we add $\theta_q \geq 0$ as an optimization constraint.

The proposed recommendation framework not only can improve the recommendation quality, our framework could also help with the serendipity feature of the recommender systems. Compared with single source recommendation frameworks (*e.g.*, collaborative filtering or only utilizing single entity relationship), relationship heterogeneity can find recommendation candidates along different meta-paths (different similarity semantics). Our framework although is trained based on recommendation relevance, we can revise the objective functions and use the proposed framework to promote serendipity results rather than relevant results. We discuss this possible revision in Section 8.2.1.

With the recommendation model in Equation (3.4), given a user, we can now assign recommendation scores to all items, and then rank these items accordingly. We return the top-K results as the recommendation results. We will discuss how to estimate the parameters in the recommendation model in Section 3.5.

3.4 Personalized Recommendation Model

We proposed to diffuse user preferences along different meta-paths in heterogeneous information networks, and calculate latent features for users and items under various semantic assumptions with matrix factorization techniques. We then defined the global recommendation model with these latent features, which essentially combines the observed implicit feedback from users and

different types of relationships in the information network together. During recommendation, we apply the global model to all users and utilize users' personal feedback to generate recommendation results. However, such a solution does not distinguish user interests or behavior patterns at the model level. For example, the learned global model may suggest that majority users watch popular movies featuring famous actors, but this rule might not be true for all individuals.

In this section, we extend the proposed global recommendation model to a finer level of granularity. Rather than learn one recommendation model for all the users, we aim to calculate different recommendation models for different users to better capture user preferences and interests. The straightforward way of learning personalized models is to estimate the recommendation model defined in Equation (3.4) with each user's own implicit feedback data. However, the number of feedback per user follows power law distribution (we demonstrate this claim in Section 3.6), which means we do not have enough feedback data to learn personalized models for most of the users.

Although users may have different behaviors from each other, a subgroup of users can share similar interests and preferences from certain perspectives. For example, comic fans are interested in super-hero, fantasy and adventure movies while fans of Steven Spielberg follow movies directed by him. Motivated by this observation, we propose to first cluster users based on their interests, and then learn a recommendation models within each cluster. Notice that one user can belong to different user clusters (one can be comic fan and Spielberg fan at the same time). When recommending, we first calculate the personalized recommendation model for the target user by combining the recommendation models of the related user clusters, and then calculate the recommendation results with the personalized model of the target user. We define the personalized recommendation function for user u_i as follows:

$$\hat{r}^*(u_i, e_j) = \sum_{k=1}^{c} sim(C_k, u_i) \sum_{q=1}^{L} \theta_q^{\{k\}} \cdot \hat{U}_i^{(q)} \hat{V}_j^{(q)T} \tag{3.5}$$

where C represents user clusters related to target user u_i and function $sim(\cdot, \cdot)$ defines the cosine similarity between the center of cluster C_k and u_i. $\theta_q^{\{k\}}$ represents the recommendation model defined in cluster C_k. We define $\Theta^{\{\cdot\}} = \theta^{\{1\}}, ..., \theta^{\{c\}}$ as the recommendation model parameters for this approach.

Compared with L parameters in the global recommendation model (Equation (3.4)), the per-

sonalized recommendation approach has a total $c \times L$ parameters, where c is the number of clusters. With a relatively larger parameter space, now we can generate personalized recommendation models efficiently and represent different user interests or behavior patterns effectively. We discuss the user clustering and the model learning algorithm in details in Section 3.5.

After estimating recommendation models for all the user clusters $\Theta^{\{\cdot\}}$, when recommending items for u_i, we first find the clusters that u_i is related to (has a high similarity to the cluster center), and then combine the related user clustering parameters following Equation (3.5). With the calculated personalized recommendation model, we can assign recommendation score to each item for u_i. Recommendation can be made by sorting all items with their recommendation scores and returning the top-k items.

The number of clusters could be essential to this method. If the cluster number c is too small, we may not be able to distinguish user interests well. If the cluster number is too big, the number of users in each cluster becomes very small. In this case, we may not have enough training data to learn the recommendation models. A good estimation of the optimal number of clusters can be achieved by cross-validation using training data. We discuss the performance change with different parameter c in Section 3.6.5.

3.5 Model Learning with Implicit Feedback

In this section, we introduce learning algorithms for both global and personalized recommendation models. We first discuss parameter estimation method for global recommendation model (Equation (3.4)) and then extend the learning algorithm to personalized recommendation models.

Recommendation models proposed in this study take advantage of the heterogeneous entity relationships in information networks. More specifically, we combine network diffusion-based latent features with parameters indicating the importance of the corresponding meta-path in the recommendation process. To learn the importance of the latent features, we use user implicit feedback as training data. As discussed in Section 3.2, the value 1 in implicit feedback data represents positive feedback (users are interested in such items) while the value 0 represents a mixture of negative feedback (users are not interested in such items) and unobserved potential interactions (users are not aware of such items). Traditional learning methods adopt classification or learning-to-rank

objective functions and usually treat 1s in training dataset as positives and 0s as negatives. As we discussed, such methods do not fit in the definition of implicit feedback data and they cannot generate high quality recommendation models.

Motivated by [54], we employ a different learning approach by considering the correct item pair orders. We define an objective function to order 1 values before 0 values for each user. The assumption behind this objective function is that users are more interested in the items with value 1 in R than the rest of the items, which is a weaker and more plausible assumption compared with the traditional approaches.

3.5.1 Bayesian Ranking-Based Optimization

We use $p(e_a > e_b; u_i|\theta)$ to denote the probability that user u_i prefers e_a over e_b. The Bayesian formulation of the optimization criterion is to maximize the posterior probability as follows:

$$p(\Theta|R) \propto p(R|\Theta)p(\Theta), \tag{3.6}$$

where $\Theta = \{\theta_1, ..., \theta_L\}$ represents the global model parameters, and $p(R|\Theta)$ represent the probability that all item pairs can be ranked correctly for all users according to R, *i.e.*, for each user, items with feedback 1 can be ranked before items with feedback 0.

With the assumption that both user preferences and ordering of the item pairs are independent, we can expand the likelihood function $p(R|\Theta)$ as follows:

$$\begin{aligned} p(R|\Theta) &= \prod_{u_i \in \mathcal{U}} p(R_i|\Theta) \\ &= \prod_{u_i \in \mathcal{U}} \prod_{(e_a > e_b) \in R_i} p(e_a > e_b; u_i|\Theta) \end{aligned} \tag{3.7}$$

where $(e_a > e_b) \in R_i$ represent all item pairs with the correct orders in the observed implicit feedback of u_i.

We define $p(e_a > e_b; u_i|\theta)$ as:

$$p(e_a > e_b; u_i|\Theta) = \sigma(\hat{r}(u_i, e_a) - \hat{r}(u_i, e_b)), \tag{3.8}$$

where σ is the logistic sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$.

We assume that $p(\Theta)$ is a Gaussian distribution with zero mean and variance-covariance matrix $\Sigma_\Theta = \lambda I$. With the probability and the likelihood defined above, we can derive the objective function as follows:

$$\begin{aligned} O &= -\ln p(\Theta|R) = -\ln p(R|\Theta)p(\Theta) \\ &= -\sum_{u_i \in \mathcal{U}} \sum_{(e_a > e_b) \in R_i} \ln p(e_a > e_b; u_i|\theta) + \lambda \|\Theta\|_2^2 \\ &= -\sum_{u_i \in \mathcal{U}} \sum_{(e_a > e_b) \in R_i} \ln \sigma(\hat{r}(u_i, e_a) - \hat{r}(u_i, e_b)) + \lambda \|\Theta\|_2^2 \end{aligned} \tag{3.9}$$

where $\lambda\|\Theta\|_2^2$ is a data dependent L_2 regularization term.

By minimizing O in Equation (3.9), we can estimate the recommendation parameter Θ from the implicit feedback data.

3.5.2 The Optimization Algorithm

Notice that Equation (3.9) is differentiable, many optimization techniques (*e.g.*, SGD [9], BFGS-B method [11]) can be utilized to estimate parameter Θ. The gradient of Equation (3.9) with respect to Θ can be calculated as follows:

$$\begin{aligned} \frac{\partial O}{\partial \Theta} &= -\sum_{u_i \in \mathcal{U}} \sum_{(e_a > e_b) \in R_i} \frac{\partial}{\partial \Theta} \sigma(\hat{r}_{i,ab}) + \frac{\lambda}{2} \frac{\partial}{\partial \Theta} \|\Theta\|_2^2 \\ &= -\sum_{u_i \in \mathcal{U}} \sum_{(e_a > e_b) \in R_i} \frac{e^{-\hat{r}_{i,ab}}}{1 + e^{-\hat{r}_{i,ab}}} \frac{\partial}{\partial \Theta} \hat{r}_{i,ab} + \lambda \Theta, \end{aligned}$$

where $\hat{r}_{i,ab} = \hat{r}(u_i, e_a) - \hat{r}(u_i, e_b)$.

Considering the data size of the real-world recommender systems, with the above gradient, we employed the stochastic gradient descent (SGD) method [9] to estimate the parameters in our empirical studies. Notice that the time complexity of this proposed learning process is $O(mn^2)$ where m is the number of users and n is the number of items. In large datasets this can be overwhelming. With SGD, we only need to estimate the gradient with a very small subset (10^{-5} sample rate) of training pairs sampled from R at each iteration. We discuss sample rate selection

for parameter estimation in Section 3.6.

3.5.3 Learning Personalized Recommendation Models

As discussed in Section 3.4, the proposed global recommendation model failed to distinguish individual interests and behavior differences, and thus the quality of the results may not be satisfying. We observed that although users may have different interests from each other, a subgroup of users can share similar interests or behavior patterns. Instead of learning one global model, which cannot represent user individuality, or learning personalized models directly, which can be time consuming and may cause model over-fitting, we propose to learn recommendation models for user subgroups. With recommendation models for user subgroups, when making recommendations, we can calculate the personalized recommendation model for a target user by combining the recommendation models he or she is most related to, and generate recommendation results accordingly.

In order to learn recommendation models for user subgroups, we first need to cluster users based on their interests and preferences, by examining the user implicit feedback data. Considering the sparsity of R, we first learn low-rank representation for users by applying non-negative matrix factorization on R directly. With the low dimension user matrix U, we apply the well-studied *k-means* algorithm with a cosine function as similarity measurement between users, to finally cluster users into subgroups. For each cluster, we apply the techniques we discussed above to learn a recommendation model. The learning algorithm of personalized recommendation models can be found in Algorithm 1.

After estimating parameters of the recommendation models, given a target user, we can calculate the corresponding personalized recommendation model with Equation (3.5), and make personalized entity recommendation by using both the personalized recommendation model and his or her personal feedback data accordingly.

3.6 Empirical Study

We present the empirical studies of the proposed recommendation framework in this section. We implemented both global and personalized recommendation models proposed in Section 3.3 and 3.4 along with several popularly deployed or the state-of-the-art implicit feedback recommendation

Algorithm 1: Learning Personalized Recommendation Models

```
// input:  implicit feedback and information network
// output:  recommendation models for user clusters
```
Input: R, G
Output: $\Theta^{\{\cdot\}}$
Prepare L meta-paths in the format of **user** $-$ **item** $-*-$ **item**
```
// User preference diffusion along meta-paths
```
for $q \leftarrow 1$ ***to*** L **do**
  foreach u_i *and* e_j **do**
    $\tilde{R}^{(q)}_{u_i,e_j} = s(u_i, e_j|\mathcal{P}^{(q)})$ (Equation (3.2))
  end
  Calculate latent features $\hat{U}^{(q)}, \hat{V}^{(q)}$ with $\tilde{R}^{(q)}$ (Equation (3.3))
end
```
// Clustering users into subgroups
```
Factorize R and derive U, V
$C = \text{k-means}(U)$
```
// Learn recommendation models
```
foreach C_k *in* C **do**
  Optimize $\Theta^{\{k\}}$ with implicit feedback in user subgroup C_k (Equation (3.9))
end

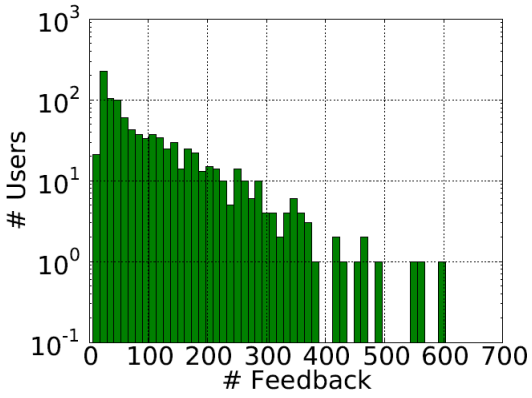
techniques. We applied these methods on two real-world datasets and performed a series of experiments to demonstrate the effectiveness of the proposed approach. We present both experimental results and discussion with analysis.

3.6.1 Data

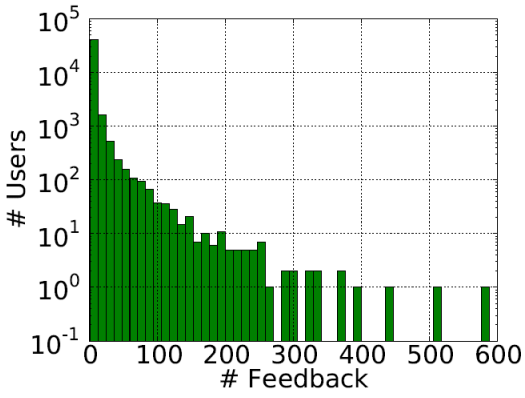
To demonstrate the effectiveness of the proposed recommendation framework, we choose two datasets from different domains (movie and local business) for empirical studies. The first dataset is built by combining the popular MovieLens-100K dataset and the corresponding IMDb dataset together. We name this dataset IMDb-MovieLens-100K (IM100K). We use MovieLens user ratings as user feedback and we build a corresponding heterogeneous information network from the IMDb dataset. If users watched a certain movie and wrote a review about this movie, no matter whether they liked the movie or not, we say we observed the user-item interaction and we set this feedback as 1, otherwise, we set as 0. When building this dataset, we mapped two datasets using titles and release date of the movies, which could be erroneous on certain movies so the results we presented below are lower-bound of the actual performances.

Name	#Items	#Users	#Ratings	#Entities	#Links
IM100K	943	1360	89,626	60,905	146,013
Yelp	11,537	43,873	229,907	285,317	570,634

(a) Datasets Description



(b) IMDb Feedback Distribution



(c) Yelp Feedback Distribution

Figure 3.4: IM100K and Yelp Datasets

The second dataset is the Yelp challenge dataset[a]. This dataset contains both user reviews and local business information (an information network). When a user wrote a review for a restaurant, we set the feedback as 1, otherwise it would be set to 0. We summarize these two datasets in Figure 3.4(a), the schemas of which can be found in Figure 2.1. Notice that the Yelp dataset is much sparser than the IM100K dataset, so the performances of all methods decline accordingly. The distributions of the user feedback can be found in Figure 3.4.

Both datasets have timestamps with each user item interaction. We split the feedback matrix R of both datasets for each user into training and test based on timestamps, *i.e.*, we use 80% of the "past" feedback to predict 20% of "future" feedback. In Yelp dataset, we have to filter out all the users who have less or equal to 2 reviews otherwise we can not create training and test data properly.

3.6.2 Competitors and Evaluation Metrics

We implement several widely deployed or the state-of-the-art recommendation approaches as comparison methods as follows:

- **Popularity:** Recommend the popular items to users.

[a] `http://www.yelp.com/dataset_challenge/`

Table 3.2: Meta-path examples (we set $n = 1$ and 2. "biz" is short for "local business")

Network	Meta-Path
IM100K	user−(movie−tag−movie)^n
	user−(movie−director−movie)^n
	user−(movie−genre−movie)^n
	user−movie−plot−movie
Yelp	user−(biz−category−movie)^n
	user−(biz−customer−biz)^n
	user−biz−checkin−biz
	user−biz−location−biz

- **Co-Click:** Estimate conditional probabilities between items and recommend items with an aggregated conditional probability calculated using the training data of the target user.
- **NMF:** Non-negative matrix factorization on R, details of which is discussed in Section 3.2.2
- **Hybrid-SVM:** Use SVM-based ranking function [33] to learn a global recommendation model with user implicit feedback and meta-paths based similarity measures [59].

We use *HeteRec-g* to denote the proposed global recommendation model and *HeteRec-p* to represent the personalized recommendation models derived from user subgroup recommendation models. We utilize 10 different meta-paths in each information network, including the most simple meta-path $user - item$, which means user preferences can only be propagated to items with observed positive feedback. We list some other meta-paths and / or attribute similarity measures in Table 3.2. For explicit feedback recommendation evaluation, measures like *root mean square error* (RMSE) are the standard evaluation metric. However, these metrics do not suit the definition of implicit feedback problem. In this study, we test all methods as ranking models and use the well-studied information retrieval metrics including precision-at-position and top-10 *mean reciprocal rank* (MRR, Equation (3.10)) to evaluate and compare the performance of these methods.

$$\text{MRR}_K = \frac{1}{m}\sum_{i=1}^{m}\left(\sum_{e \in test(u_i)} \frac{1}{rank(u_i, e)}\right) \tag{3.10}$$

3.6.3 Performance Comparison

The performance of all 6 methods in the two datasets can be found in Table 3.3.

Table 3.3: Performance Comparison

Method	IM100K				Yelp			
	Prec1	Prec5	Prec10	MRR	Prec1	Prec5	Prec10	MRR
Popularity	0.0731	0.0513	0.0489	0.1923	0.00747	0.00825	0.00780	0.0228
Co-Click	0.0668	0.0558	0.0538	0.2041	0.0147	0.0126	0.01132	0.0371
NMF	0.2064	0.1661	0.1491	0.4938	0.0162	0.0131	0.0110	0.0382
Hybrid-SVM	0.2087	0.1441	0.1241	0.4493	0.0122	0.0121	0.0110	0.0337
HeteRec-g	0.2094	0.1791	0.1614	0.5249	0.0165	0.0144	0.0129	0.0422
HeteRec-p	**0.2121**	**0.1932**	**0.1681**	**0.5530**	**0.0213**	**0.0171**	**0.0150**	**0.0513**

Based on Figure 3.4, user feedback data follow power law distribution, *i.e.*, a very small number of items have interaction with a large number of users. Due to this property, recommending the popular items to users has a decent performance (MRR=0.1923 in IM100K). Co-click method, as one of the most widely deployed technique, achieves MRR = 0.2041 in IM100K and has a similar performance as the NMF method in Yelp (MRR=0.0371).

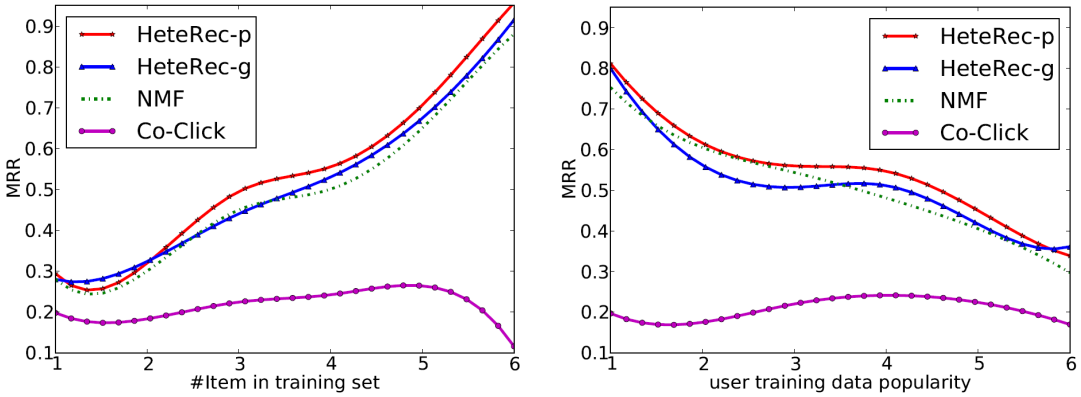
We implemented the NMF as the CF baseline (details of this method can be found in Section 3.2.2). We set the dimensionality of the low-rank representations $d = 20$ in IMD100K and $d = 60$ in Yelp with cross validation in training dataset. We use the same method and settings in the diffusion-based latent feature generation method for the proposed approaches. With parameter tuning and additional information (*e.g.*, [27]), NMF may perform better than the results in Table 3.3. However, the same performance improvement can be achieved in our methods accordingly by replacing the NMF solver in Equation (3.3) with a more advanced technique. As presented in Table 3.3, NMF achieved MRR = 0.4938 and Prec1 = 0.2061 in IM100K dataset and MRR = 0.0382 and Prec1 = 0.0162 in Yelp dataset. This method outperforms other baselines methods in both datasets.

Hybrid-SVM method is a hybrid recommendation approach which uses the same amount of information as our proposed methods. This method combines both implicit feedback and heterogeneous relationship information following the intuitions of our study. However, it adopts an SVM based ranking framework [33] and uses PathSim [59] measures as features when defining the recommendation model. Without the proposed diffusion-based feature generation method, the learning algorithm and the recommendation model personalization, Hybrid-SVM can not fully take advantage of the feedback data and the heterogeneity of the information network. This method can only

achieve MRR = 0.4493 in IM100K (compared to 0.4938 with NMF) and MRR = 0.0337 (compared to 0.0371 with Co-Click method). The low-performance of the Hybrid-SVM method proves the effectiveness of the proposed framework.

Our proposed global recommendation model (HeteRec-g), which takes advantage of both user feedback and the related information network, and users the same amount of information as Hybrid-SVM, beats all baseline methods in both datasets. It improves MRR by 6.1% compared to NMF in IM100K and 10.4% in Yelp dataset. This proves our assumption that adding information network as external knowledge with the proposed approach can alleviate the data sparsity issue and improve the recommendation quality. Morever, HeteRec-g produces much more accurate recommendation results compared to Hybrid-SVM (in IM100K, MRR of HeteRec-g is 0.5249 while MRR of Hybrid-SVM is only 0.4493). Both methods utilize the same set of meta-paths, and use the same sample rate during training. Both methods define "global" recommendation models since they apply the same model to all the users. The performance increase of HeteRec-g proves the effectiveness of our diffusion-based latent feature generation method. Another interesting observation is that the MRR gain of HeteRec-g compared with NMF in the relatively dense IM100K dataset is less than it is in the sparer Yelp dataset (6.1% v.s. 10.4%), which fits our intuition that when feedback dataset is sparser, the informative network-based recommendation approach can improve the performance even more. When training, we employ a uniform sample rate 10^{-5} in SGD and we apply the same rate to all supervised approaches in this experiment. Parameter tuning of the sample rate is discussed later this section.

HeteRec-p as the personalized recommendation approach can further improve the performance in both datasets. HeteRec-p method clusters users based on their interests and utilizes personalized model parameters when recommending. This approach can distinguish user behaviors while HeteRec-g treats all users as the same. We use $c = 10$ in IM100K dataset and $c = 100$ in Yelp dataset. We discuss the strategy of choosing the correct granularity later this section. Compared with the global recommendation model (HeteRec-g), personalized models can provide higher quality recommendation results in both datasets. It improves Prec5 by 7.9% and MRR by 5.4% in IM100K, and improves Prec1 by 29% and MRR by 21.5% in Yelp. This verifies that different users indeed have different interests and behavior patterns. Applying global recommendation model to



(a) Performance Change with User Feedback Number

(b) Performance Change with User Feedback Popularity

Figure 3.5: Performance Analysis and Parameter Tuning

all users can not distinguish user differences properly. Recommendation models which are learned within user subgroups can represent user interests and behaviors at a finer granularity level, and thus HeteRec-p can generates better results. Compared with baseline methods which only utilize user feedback data, HeteRec-p surpasses NMF by 12.0% in terms of MRR in IM100K and beats the MRR score of Co-Click by 38.3% in Yelp dataset.

Overall, the proposed recommendation models outperform all comparison methods in both IM100K and Yelp datasets. The experiments verify that using heterogeneous information networks in recommender systems can improve recommendation quality. Personalized recommendation models can better distinguish user interests and behaviors, and thus can lead to personalized and more accurate recommendation results.

3.6.4 Performance Analysis

We analyze the performance of Co-Click, NMF, HeteRec-g and HeteRec-p in different recommendation scenarios. We ran the following analytical experiments on both datasets and observed similar performance change in both datasets. Due to the page limitation, we only present these findings with IM100K dataset.

We first study the correlation between the performance of different recommendation methods and the training feedback data size of each user. We split all users into 6 groups based on their individual training data size. Users in group 1 provided very limited number of feedback (average

size is 13) while users in group 6 provided the most amount of feedback (average size is 224). We apply all 4 methods in each group. The results of this study can be found in Figure 3.5(a). One can notice that overall HeteRec-p outperforms all the other methods for users with different feedback sizes. When the user feedback size is small, *i.e.*, the data sparsity issue is severe, the performances of all methods are bad. When the feedback size of each user increases, the performances increase accordingly. The performance of Co-Click however does not change as much among different groups. This proves that all CF based methods could be affected when data are sparse. When feedback data are insufficient, it is critical to utilize information network as external knowledge to alleviate this issue.

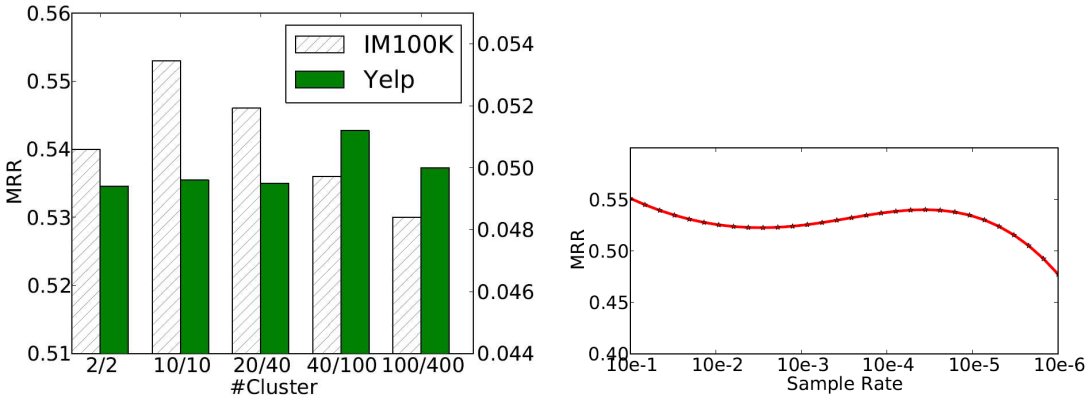
We then study the correlation between performances of the 4 methods and the popularity of the items that users interacted in the feedback dataset. We split users into 6 groups based on the average popularity of the movies they interacted with in the training dataset. Users in group 1 prefer less popular movies (average popularity of items is 71) whle users in group 6 prefer the most popular movies (average popularity is 281). The results of this study can be found in Figure 3.5(b). Similar to the previous study, overall HeteRec-p outperforms all the other methods for different user groups. Interestingly, all CF based methods perform better for users who prefer unpopular movies. This finding may be counter-intuitive. It would seem like popular movies are easier to handle since the related data are sufficient. However, users who prefer popular movies usually do not have specific interests (they watch anything popular without considering genres, stories or any other types of information). Recommending movies to such users is always challenging. One possible way of handling this problem is to identify such users and use popularity based methods for the recommendation.

3.6.5 Parameter Tuning

The proposed methods have several additional parameters compared with other methods.

In Equation (3.9), λ controls L_2 regularization of the function. We cross-validated this parameter and set it to 0.1 when optimizing objective function. Another parameter is the sampling rate in SGD when estimating parameters. As mentioned in Section 3.5, with the proposed objective function, the scale of the training dataset is $O(mn^2)$ which can be overwhelming in large datasets (this

number in Yelp dataset is approximately 10^{12}). Instead of using the entire dataset, we only sample a subset during training. We study the relationship between the sample rate and the performance of HeteRec-g in IM100K (Figure 3.6(b)).



(a) Local Model Cluster Number Tuning

(b) Sample Rate Tuning

Figure 3.6: Parameter Tuning

Notice the x-axis in Figure 3.6(b) is at log scale. When sample rate is more than 10^{-5}, the performance of the global model is relatively stable which means we do not need all the data to learn a high quality model with the proposed approach. However, inefficient training data size could harm the performance when we only supply 10^{-6} or less training data.

Besides pairwise training in parameter estimation, user preference propagation along longer meta-path and factorization with dense matrix can both be computationally expensive. In this study, we use generic methods for these calculations, which are slow and not scalable. However, other more advanced methods can be utilized in this framework, *e.g.*, distributed NMF [43] and partial materialization during preference propagation (*e.g.*, utilize symmetric partial meta-path results). These methods should provide accurate or approximate results and speed up the proposed framework when dealing with large scale datasets.

Another parameter for HeteRec-p is the number of user clusters. As presented in Figure 3.6(a), although not very sensitive to this parameter, HeteRec-p does perform differently with different numbers of clusters. It peaks at $c = 10$ in IM100K and $c = 100$ in Yelp dataset compared with other parameters tested. When the number of clusters is small, HeteRec-p could not distinguish users behavior very well while a large cluster number could lead to training data deficiency for each subgroup model, and thus lead to performance decreasing.

3.7 Conclusion

In this chapter, we study recommendation in the scope of heterogeneous information networks. We propose a generic recommendation framework for implicit feedback dataset by taking advantage of different types of entity relationships in heterogeneous information networks. We define recommendation models at both global and personalized levels. Personalized recommendation models can be efficiently generated on the fly, and this approach can provide high quality personalized recommendation results compared to other recommendation methods. A Bayesian ranking process is utilized to estimate the weights of the recommendation models. We compared the proposed approaches with several widely employed or state-of-the-art implicit feedback recommendation techniques, and empirical study demonstrates the effectiveness of our methods. We also analyzed the performance of these methods under different scenarios and explained the reasons of the performance drift. Interesting future work includes on-line recommendation model update with users providing model feedback, large scale recommendation model in information networks as well as approximate learning process with low time complexity.

Chapter 4

Recommendation with User Log and Freebase

4.1 Overview

In order to meet web users' ever-increasing information needs, search engines are shifting their focus from the top-10-blue-link query answering paradigm to information and knowledge discovery. Instead of passively matching users' queries to web documents, today's search engines are striving to proactively provide valuable and related information to users. Inspired by this paradigm change, many techniques have been proposed to help users explore web content, *e.g.*, query suggestion [12] and website recommendation [45] [67]. However, such efforts are conducted on a relatively coarse granularity. For example, though website recommender systems suggest websites of potential interest to users, they treat sites as black boxes without considering the content, and thus they cannot present detailed information to users directly.

With the active study and rapid evolution of semantic web techniques, entity graphs which contain entities, attributes, relationships as well as other structural data become widely accessible. Most recently, several commercial search engines have enhanced their search experience by displaying related entity information from entity graphs along with web search results. For example, when answering a query like "skyfall", besides traditional web search results and the entity attributes of

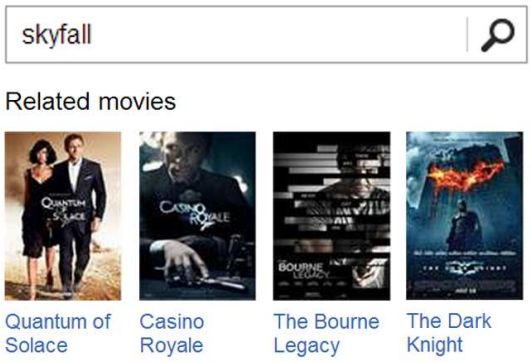


Figure 4.1: Related entity suggestion feature on major search engines

Figure 4.2: Personalized entity recommendation scenario studied in this work. When a user types movie related queries in a search engine, like "movies", "skyfall", we show personalized movie recommendations to this user.

this movie, as shown in Figure 4.1, search engines also suggest "related movies" or display "people also search for" results. These features take an important step forward in unifying search and information discovery experience, as the recommendations are now generated at the entity level.

However, to our best knowledge, no existing search engine currently provides personalized entity recommendations based on users' past behaviors. To further enhance user search experience and improve information recommendation quality in search engines, we study personalized entity recommender systems for search engine users in this study.

By taking advantage of user click log and knowledge extracted from Freebase[a], the proposed framework analyzes users' preferences and interests, and then recommends entities of interest to search engine users during search sessions. By deploying such a system, when a user searches for movie-related web content, in addition to suggesting "related movies", we present personalized movie recommendations as well, illustrated in Figure 4.2. Notice that in this work, personalized entity recommendations may not directly depend on the queries. Instead, we use the search queries only to determine the domain of recommendation.

In order to build such a system, we are facing many technical as well as data related challenges, including but not limited to:

- How to map users' clicked URLs to specific entities in the knowledge base.
- How to take advantage of entity relationships and different types of attributes in the user log and knowledge base when defining recommendation models.
- How to model users' drift of interests over time.

[a] `http://www.freebase.com/`

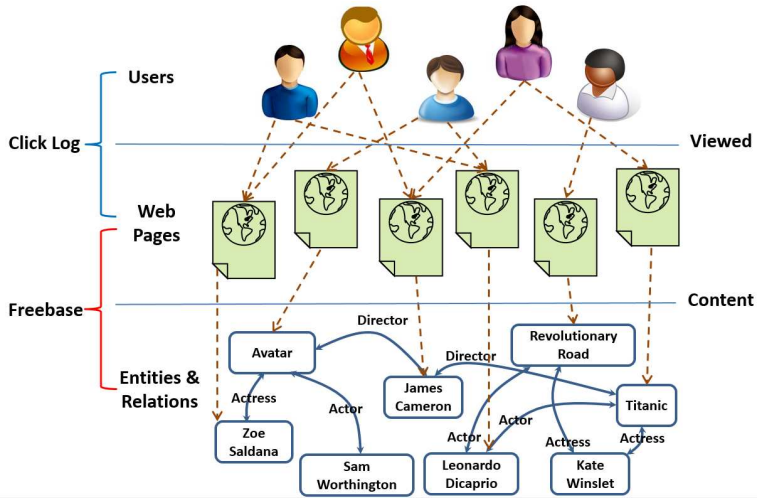


Figure 4.3: Heterogeneous relations between users, web pages and entities

- How to utilize users' non-entity related click logs.

Aiming to solve the problems mentioned above, we first explore heuristics and features which are critical to the entity recommendation problem, such as different types of entity relationships in the knowledge base, the consistency and drift of users' interests, and cross-domain correlations. Then we propose a global recommendation model which utilizes the aforementioned heuristics and features. We apply the same global recommendation model to different user click logs to achieve personalized recommendation results for different users. We then propose personalized recommendation models for different users to further distinguish users' behaviors at the model level. Empirical studies and analysis in Section 4.5 demonstrate the effectiveness of the proposed framework over traditional entity recommendation techniques.

The major contributions of this work are summarized as follows:

- **Conceptual**: In order to improve user information discovery experience, we study a novel application, *i.e.*, personalized entity recommender system for search engine users, which requires good understanding on both user click log and the Freebase knowledge base.

- **Modeling**: We design a general learning framework which can model different features from both user click log and Freebase entity graph at different granularity levels.

- **Experimental**: We conduct experiments on user click data extracted from a commercial

search engine. The results demonstrate the effectiveness of our proposed framework in comparison with several popular recommendation algorithms.

The remainder of this chapter is organized as follows. Section 4.2 introduces the background and preliminaries of this study. Section 4.3 discusses features and heuristics in user click log and the Freebase entity graph. Section 4.4 introduces the proposed entity recommender framework. Experiments and results are discussed in Section 4.5. Conclusions and future works are presented at the end of the paper.

4.2 Background and Preliminaries

In this section we present background and preliminaries of this study, including input data format and the definition of the entity recommendation problem of our study.

4.2.1 User Click Log

User click log contains the web pages users visited when using search engines. For each user, a user log sequence can be collected by sequentially connecting web pages this user visited following an ascending timestamp order. We denote user click log sequence for user u as

$$L^u = \langle e_1^u, e_2^u, ..., e_t^u, ..., e_T^u \rangle,$$

where e_t^u is the web page u visited at timestamp t. Attributes including timestamp, language of the web page, dwelling time of the visiting event, time of day, *etc.*, may be utilized for entity recommendation task as well.

Additionally, we use L_t^u to represent the user log history of user u from timestamp 1 to $t-1$, *i.e.*, $L_t^u = \langle e_1^u, e_2^u, ..., e_{t-1}^u \rangle$. We name timestamp T, the most recent timestamp in the user log sequence of u, the *target timestamp*. When evaluating recommendation results, we use L_T^u as input and recommend potential entities of interest to user u at timestamp T. We define e_T^u as the entity page that u actually visited at T, and use this entity page as the ground truth when comparing different recommender systems in Section 4.5.

The top two layers ("Users" and "Web Pages") in Figure 4.3 illustrate the relationship between users and web pages in user click log.

4.2.2 Freebase Entity Graph

Freebase [8] is a large collaborative knowledge base consisting of entities and relationships composed mainly by its community members. It now contains 1.9 billion instances of relationships between 40 million entities. Freebase is widely used in academia and industry in many research problems and applications [41] [51].

Each entity in Freebase are associated with some URLs that are related to this entity. For example, for the movie entity "Jobs"[b], by utilizing the relationships "/common/topic/official_website" and "/common/topic/topic_equivalent_webpage", we can obtain this movie's official site, IMDb pages as well as Wikipedia pages. Moreover, we can also get other types of entities related to this movie, including actors, directors, genres, producers, *etc.*

The bottom two layers ("Web Pages" and "Entities & Relations") in Figure 4.3 present the relationship between web pages and Freebase entity graph. Moreover, in this study, we use the phrases "Freebase", "knowledge base" and "entity graph" interchangeably.

4.2.3 Mapping URLs to Entities

With the definitions in Section 4.2.1 and Section 4.2.2, we can now map users' clicked URLs in user log to the corresponding entities in Freebase, as demonstrated in Figure 4.3. A user's interests can now be represented by a set of entities and web pages this user visited before. The visited entities and web pages as well as various relationships between these entities can be utilized for making personalized recommendation for this user.

4.2.4 Problem Definition

With the definitions of user click log and Freebase entity graph, we define entity recommendation problem for search engine users as follows:

[b] `http://www.freebase.com/m/0j7j4ls`

Given a set of user click log sequences L, the Freebase entity graph G, and a specific click log sequence L_T^u of user u, an entity recommendation model for search engine users should be able to recommend potential entities of interest to user u at timestamp T. We denote this recommendation result as $\hat{e}_T^u$.

Important notations used in the rest of the paper can be found in Table 4.1.

4.3 Exploring The Data

In this section, we discuss features and heuristics in search engine user click log and the Freebase entity graph, which can be advantageous when building entity recommender systems for search engine users.

4.3.1 Consistency and Drift of User Interest

Modeling user interests is crucial in building entity recommender systems. For a specific user, his or her interests usually cover only a small number of topics over a certain time period. A user may be interested in "X-Files" related information this week, and then begins to search for romantic movie related web pages the next week. Recommending related entities in a timely manner can assist the information gathering process of this user.

One possible way of recommending entities to users with this motivation is to estimate conditional probabilities (*a.k.a* co-click or co-occurrence) between entities in recent user click log. With learned conditional probabilities, given a user log sequence L_T^u, recommendation score for certain entity $\hat{e}_T^u$ can be calculated as follows:

$$r(\hat{e}_T^u, L_T^u) \propto \sum_{e_t^u \in L_T^u} P(\hat{e}_T^u | e_t^u). \tag{4.1}$$

Conditional probability between entities is easy to estimate and can be very effective with sufficient training data. However, such approach can only recommend entities that have previously been observed in the user log before, hence it suffers from the *cold start* problem for newly introduced entities.

Although user interests might follow one theme or topic in a certain time period, eventually

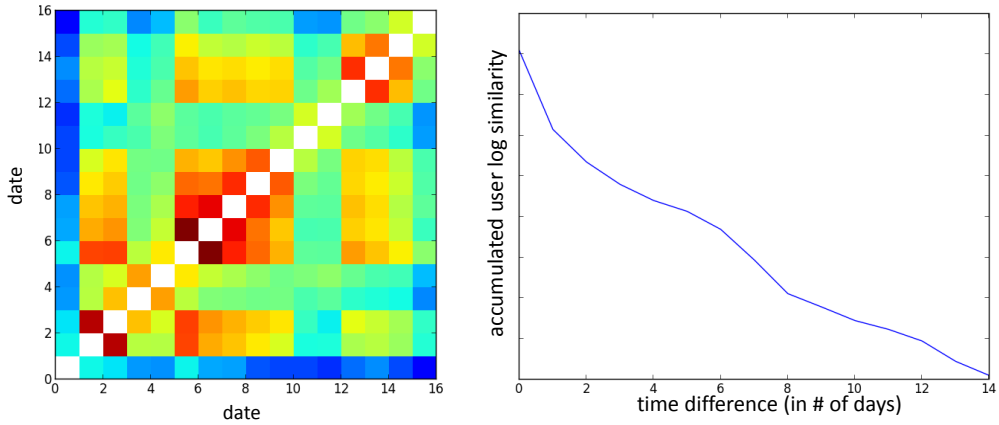
Table 4.1: Notations

Notation	Description
e_t^u	entity page user u visited at timestamp t
L, G	user log and entity graph
$w_t(\cdot,\cdot)$	temporal similarity of two page visiting events
$w_u(\cdot,\cdot)$	similarity between two user log sequences
$\hat{e}_T^u$	recommended entity to user u at timestamp T
L_t^u	user log sequence of u from timestamp 1 to $t-1$
$N(\cdot)$	user log subsequence neighbors
$S(\cdot,\cdot)$	a user log or entity graph pairwise feature
θ	parameters for recommendation models

they change over time. We demonstrate this phenomenon by measuring similarities between entities (using pairwise shortest distance-based method in entity graph) from different days in the same user log sequence. In Figure 4.4(a), we sum up user log similarities of all users for each day and plot the accumulated similarity difference between days with a heat map (both axes represent date, 0 is March 1st, 1 is March 2nd, *etc*). Red color indicates high similarity while blue indicates dissimilarity. One can observe that the accumulated user log similarity is higher when the two dates are closer. Notice that the heat map possesses a 7-day periodic pattern, which is caused by users' weekly behavior repeat.

To eliminate such periodic patterns, we aggregate similarity differences between days and plot it against day difference (1 means the entity similarities are one day apart) (Figure 4.4(b)). The scale of y-axis has been removed to comply with the company's non-disclosure policy. In both plots, users' interests stay relatively consistent when the time interval is small (1 or 2 days), but change dramatically when the time interval is big (2 weeks).

In order to provide satisfying recommendation results, an entity recommender system should model both the consistency and the drift of user interest simultaneously. When recommending entities, the recommender system should rely more on recent behaviors, less on the old behaviors and age user log with a time decay function accordingly.



(a) (b)

Figure 4.4: User interests drift over time ((a) is the user interest similarity heat map and (b) is the aggregated user interest similarity plot again time difference)

4.3.2 Entity Relationships in Entity Graph

The entity relationship heterogeneity of the Freebase entity graph provides possibilities of measuring entity similarities from different perspectives. Taking the movie entity graph schema in Figure 2.1(b) as an example, we can claim two movies are similar if they share the same genres (*movie-genre* relation), cast (*movie-actor* relation), or directors (*movie-director* relation). Additionally, previous studies [38] [59] introduced approaches to quantitatively measure entity similarities in entity graphs, and suggested that entities which are connected by different types of paths could be similar for different reasons. Entity relationships and similarity measurements can help capture users' different interests when defining entity recommendation models.

4.3.3 Cross-Domain Correlation

When building recommendation models in one domain, *e.g.*, movie recommendation, certain seemingly irrelevant information can also contribute (*e.g.*, users' preferences in books and music). In order to demonstrate this cross-domain correlation heuristic, we calculate the conditional probabilities between movie entities and two non-movie websites with user click log from summer 2012. The top 10 movies correlated with each website can be found in Table 4.2.

We first observe that the two ranking lists are substantially different. Comic Book Resources[c]

[c]comicbookresources.com, a comic book discussion website

Table 4.2: Cross-Domain Correlation Example

Rank	comicbookresources.com	ruelala.com
1	The Avengers	Magic Mike
2	Spider-Man	The Avengers
3	The Dark Knight Rises	Prometheus
4	Prometheus	Moonrise Kingdom
5	Men In Black 3	Ted
6	Iron Man 2	Snow White and Huntsman
7	Superman : The Man of Steel	Savages
8	Thor	Hunger Games
9	Snow White and huntsman	Rock of Ages
10	Battleship	The Best Exotic Marigold Hotel

users are mostly interested in comic book based sci-fi movies, including Spider-Man, Superman, Thor, *etc.* Rue La La[d] users, however, prefer romantic movies, *e.g.*, Magic Mike, Moonrise Kingdom, and Rock of Ages. Based on the above observation, we believe by incorporating cross-domain information when defining recommendation models, the quality of the recommendation results could be potentially improved. Most importantly, it can also greatly alleviate the cold start problem. When we know nothing about a user in a target domain, cross-domain information becomes critical in personalizing recommendation process.

4.3.4 Feature Calculation

Based on the above discussion, in Table 4.3, we present representative features generated from both user click log and Freebase, which can be potentially useful in building movie recommendation models. We group these features into four categories: entity graph path features which measure similarities between movies using the PathSim method [59]; entity graph binary features which are defined with important entity graph relationships; entity graph content features which utilize content based attributes to measure entity similarities; user log features which are generated from user click log following different heuristics.

Entity graph path features are defined along different types of paths in entity graph. We use the entity types on paths to represent the path types when there is no ambiguity (*e.g.*, movie−actor−movie). If two entities are linked together by paths following one certain path type,

[d] ruelala.com, a fashion boutique with mostly female customers

Table 4.3: Representative Features

Entity Graph Path Features	
movie−actor−movie	movies with the same actors
movie−director−movie	movies with the same directors
movie−producer−movie	movies with the same producers
movie−star−movie	movies with the same stars
movie−writer−movie	movies with the same writers
movie−genre−movie	movies with the same genres
movie−language−movie	movies with the same language
Entity Graph Binary Features	
is_prequel	$movie1$ is a prequel of $movie2$
is_sequel	$movie1$ is a sequel of $movie2$
actor−movie	actor appears in the movie
director−movie	director directs the movie
producer−movie	producer produces the movie
Entity Graph Content Features	
release date	two movie with close release dates
description similarity	text similarity in movie descriptions
User Log Features	
co-click	conditional probability between entities
global popularity	movie popularity of all time
local popularity	movie popularity today
cross-domain	cross-domain correlation

following [59], we can calculate the similarity score between these two entities with Equation 4.2:

$$S_{\mathcal{P}}(e_i, e_j) = \frac{2 \times |\{p_{e_i \rightsquigarrow e_j} : p_{e_i \rightsquigarrow e_j} \in \mathcal{P}\}|}{|\{p_{e_i \rightsquigarrow e_i} : p_{e_i \rightsquigarrow e_i} \in \mathcal{P}\}| + |\{p_{e_j \rightsquigarrow e_j} : p_{e_j \rightsquigarrow e_j} \in \mathcal{P}\}|} \tag{4.2}$$

where $\mathcal{P}$ represents the path type. $p_{e_i \rightsquigarrow e_j}$ is a path between e_i and e_j, $p_{e_i \rightsquigarrow e_i}$ is a path between e_i and e_i, and $p_{e_j \rightsquigarrow e_j}$ is a path between e_j and e_j.

Entity graph binary features are defined with the existence of certain entity relationship. Entity graph content features are defined based on the content type. For example, we use exponential decay function to define the movie release date similarity and cosine similarity to measure movie description similarity.

All features we used in this project are normalized to the range of $[0, 1]$. One may notice that, although most of the features are pairwise features defined between two entities, point-wise features like popularity can facilitate the recommendation as well. In order to incorporate such point-wise features into the recommendation model, we rewrite such features in a pairwise format

$S(e_j) = S(\cdot, e_j)$. For such a pseudo-pairwise functions, no matter what the first parameter is, they always return the point-wise function value based on e_j.

4.4 Recommendation Framework

With the features and heuristics discussed in Section 4.1 and 4.3, we present the proposed entity recommendation framework in this section.

4.4.1 Global Recommendation Model

With features generated from user click log and the Freebase dataset, considering the consistency and drift of user interest over time, we propose the following global entity recommendation model (Equation 4.3):

$$r(\hat{e}_T^u; L_T^u, \theta) = \sum_{e_t^u \in L_T^u} w_t(\hat{e}_T^u, e_t^u) \sum_{k=1}^{K} \theta_k S_k(\hat{e}_T^u, e_t^u), \tag{4.3}$$

where $w_t(e_T, e_t)$ is the temporal similarity function between two entity page visiting events, defined as $w_t(e_T, e_t) = \beta e^{-\alpha(T-t)}$. Intuitively, if timestamp T and t are nearby, w_t will assign a high similarity to these two events. However, if T and t are two remote timestamps, w_t will be small, which means e_t^u will have less effect on the recommendation results at T.

$S(\cdot, \cdot)$ are pairwise features as defined in the previous section, and K is the total number of features. θ are the parameters for the global model, representing different weights of these features. With the global recommendation model, given a user log sequence L_T^u, we can assign recommendation scores to possible entities and return the entities with high scores to user u as the recommendation results.

To estimate θ in global recommendation model, we first denote the margin between the ground truth entity e_T^u (the entity u actually visited at T) and the entity with the highest recommendation score besides e_T^u, as follows:

$$g(e_T^u; L_T^u, \theta) = r(e_T^u; L_T^u, \theta) - r(e_m^u; L_T^u, \theta), \tag{4.4}$$

where $e_m^u = argmax_{e \neq e_T^u} r(e; L_T^u, \theta)$. As explained above, e_m^u is the entity with the highest recom-

mendation besides e_T^u. θ can be estimated by minimizing the following objective function:

$$O_g(\theta) = -\sum_u g(e_T^u; L_T^u, \theta) + \frac{\lambda}{2}\|\theta\|_2^2, \tag{4.5}$$

where λ controls L_2 regularization to prevent over-fitting.

Objective function defined in Equation 4.5 is one possible way to estimate global model parameters. One can adopt other margin measures when defining g, like hinge loss, or kernel based similarity. Moreover, we can also directly optimize ranking-based metrics like reciprocal rank (Equation 4.12) by defining the margin as follows:

$$g\prime(e_T^u; L_T^u, \theta) = RR(e_T^u; L_T^u, \theta) - RR(e_m^u; L_T^u, \theta), \tag{4.6}$$

where $RR(e; L_T, \theta)$ is the reciprocal rank score of entity e given θ and L_T. However, due to scalability consideration, we need to choose the most computational efficient method. Equation 4.4 only requires the calculation of the recommendation scores of e_T^u and e_m^u. While to utilize Equation 4.6, we need to calculate the recommendation scores of all possible entities and rank them accordingly. Preliminary evaluation on a small dataset yields similar recommendation results from these two methods, suggesting that Equation 4.4 can achieve similar effectiveness as Equation 4.6. Hence, in this study, we use Equation 4.4 as the margin definition.

Global recommendation model takes advantage of various features from both user click log and the Freebase entity graph. However, when recommending entities to search engine users, we apply the same model to all the users, which may not be sufficient to capture user interest diversity.

4.4.2 Personalized Recommendation Model

In reality, search engine users have different interests and preferences. Moreover, they may like the same entities for different reasons. Some users may like comedy movies while others may prefer movies with famous actors. Even for the same movie, *e.g.*, "The Dark Knight Rises", some users are interested in this movie due to the superhero theme while other viewers might be the fans of the director. With the global model, the recommender system cannot distinguish users' different preferences properly, which may lead to unsatisfying recommendation results. To better model

search engine users' interests and preferences, we propose a personalized recommendation model as follows:

$$r(\hat{e}^u_T; L^u_T, \theta^u_T) = \sum_{e^u_t \in L^u_T} w_t(\hat{e}^u_T, e^u_t) \sum_{k=1}^{K} S_k(\hat{e}^u_T, e^u_t)\theta^u_k. \tag{4.7}$$

Different from Equation 4.3, personalized recommendation model requires a set of parameters θ^u for each user.

When learning personalized models, directly maximizing margins between e^u_T and e^u_m as we did when learning the global model might not be as effective. Due to the data sparsity issue, we do not have sufficient data from one user to learn a personalized ranking model (we demonstrate this claim in Section 4.5). To alleviate such problem, when learning personalized models, we use both the user log history L^u_T of the target user, as well as other "similar" user log sequences, namely the *neighbors* of L^u_T, denoted by $N(L^u_T)$. We provide the formal definition of *neighbors* as well as an efficient approach to generate neighbors in Section 4.4.3.

Based on this philosophy, we propose the following per-user objective function to estimate parameters for the personalized models:

$$\begin{aligned} O_u(\theta^u) = \quad & - \quad \sum_{t=1}^{T} w_t(e^u_T, e^u_t) g(e^u_t; L^u_t, \theta^u) + \\ & \lambda_1 \quad \sum_{L^{u\prime}_{T\prime} \in N(L^u_T)} w_u(L^{u\prime}_{T\prime}, L^u_T) g(e^{u\prime}_{T\prime}; L^{u\prime}_{T\prime}, \theta^u) + \\ & \tfrac{1}{2}\lambda_2 \quad \|\theta^u - \theta_g\|_2^2, \end{aligned} \tag{4.8}$$

where $w_t(\cdot,\cdot)$ measures the temporal similarity of two visiting events, and $w_u(\cdot,\cdot)$ measures the similarity between two user log sequences, the definition of which is given in Equation 4.10. We use θ_g to represent the previous learned global model.

The personalized objective function O_u contains three terms. The first term in Equation 4.8 models the consistency and drift of user interest (Section 4.3.1). The personalized model parameters are learned to make predictions at T for each target user. We use these parameters to predict target user's past behavior at a previous timestamp (t, from 1 to T) given the corresponding user log sequence L^u_t. The expected accuracy of such prediction is controlled by $w_t(\cdot,\cdot)$, *i.e.*, θ^u should provide a more accurate prediction if the visiting events happened more recently. The second term

(λ_1) models the similarity between the target user log sequence and the neighbor sequences. We use parameters of u at T to predict the neighbors' behaviors at their corresponding target timestamp $T\prime$. Similarly, the expected accuracy of such prediction is controlled by $w_u(\cdot,\cdot)$, *i.e.*, θ^u should explain the neighbor's behavior more accurately if the neighbor and L_T^u are more similar. The third term (λ_2) defines the L_2 regularization between personalized models and the global model to prevent over-fitting.

The aforementioned data sparsity issue not only complicates parameter estimation process, but may also compromise the quality of the recommendations. With insufficient user log history, even with personalized models, the final recommendation results could be biased or unsatisfying. With the neighbor definition available for L_T^u, we revisit the personalized recommendation model proposed in Equation 4.7 and propose a new neighborhood based personalized recommendation model in Equation 4.9:

$$\begin{aligned} r_n(\hat{e}_t^u; L_T^u, \theta^u) = \quad & r(\hat{e}_t^u; L_T^u, \theta^u) + \\ & \lambda_1 \sum_{L_{T\prime}^{u\prime} \in N(L_T^u)} w_u(L_{T\prime}^{u\prime}, L_T^u) r(\hat{e}_t^u; L_{T\prime}^{u\prime}, \theta^u). \end{aligned} \tag{4.9}$$

In this neighborhood based recommendation function, besides personalized recommendation function defined in Equation 4.7, we also use neighbor log sequences of L_T^u to facilitate recommendation. Neighbors contribute to the recommendation score differently based on the similarity between a neighbor log sequence and the log sequence of the target user L_T^u. λ_1 controls the percentage of the neighborhood recommendation score. When $L_T^u = \emptyset$, *i.e.*, for a new user with no user log history, the first term of $r_n(\cdot)$ becomes zero, and the recommendation score will be decided by only its neighbors. In this situation, the $w_u(\cdot,\cdot)$ function will be 1 for all users in the dataset, *i.e.*, all users are neighbors of L_T^u with the same neighbor similarity. Thus this personalized model degeneralizes to the global model defined in Equation 4.3. When the log history of the target user is long and dense (active users), the recommendation score will mostly rely on the user's previous behavior and the impact from the neighbors will be lessened accordingly.

Discussion and performance comparison between Equation 4.7 and Equation 4.9 are presented

in Section 4.5.

4.4.3 User Log Sequence Neighbors

As stated above, due to data sparsity, we may not have enough data from one user to learn a personalized recommendation model. Also insufficient personal user log sequence L_T^u may compromise recommendation results. To alleviate this issue, we define similar user log sequences of L_T^u as neighbors of the target sequence, denoted by $N(L_T^u)$. We first define user log sequence similarity function as follows:

$$w_u(L_T^u, L_{t'}^{u'}) = \frac{r(E; L_T^u, \theta_g) \cdot r(E; L_{t'}^{u'}, \theta_g)}{\|r(E; L_T^u, \theta_g)\| \|r(E; L_{t'}^{u'}, \theta_g)\|}, \tag{4.10}$$

where $r(E; \cdot)$ represents the corresponding recommendation score vector over the entire entity space given certain user log L and the global model θ_g. $\|\cdot\|$ calculates the L_2 norm of the recommendation score vectors.

When searching for neighbors, we use user log subsequences $L_t = \langle e_1, e_2, ..., e_{t-1} \rangle$ as neighbor candidates. Based on the above definition, traditional *K-NN* method takes $O(N^2)$ to find the nearest neighbors for all user log sequences. However, by employing random projection based locality sensitive hashing, we can estimate the nearest neighbors of all user log sequences in $O(N)$. Details of this method can be found in [65].

With the learned personalized recommendation models for search engine users, for a target user u, the recommender system can now assign recommendation scores to potential entities of interest based on the user log history and the neighborhood information. By presenting the top-K entities when u searches for related web content in search engines, the proposed system can facilitate user's information discovery process and further improve the overall user experience of the search engines.

4.4.4 Parameter Estimation

In Equation 4.4, we utilize the margins between e_T^u, the target entity, and e_m^u, the entity with the highest score besides e_T^u given a certain θ. Notice that the entity ranking order changes with θ, thus e_m^u changes accordingly, which makes g in Equation 4.4 non-continuous and both objective functions (Equations 4.5 and 4.8) non-convex. Non-convex optimization methods, *e.g.*, Powell's method, are mostly computationally intractable. Considering the size of the training dataset (commercial search

engine user log and the Freebase entity graph), we use the following method to approximate the results.

Although Equations 4.5 and 4.8 are non-convex during the entire optimization process, when fixing e_m^u, both functions will become convex and differentiable. Due to the limitation of space, we here only give the partial derivative of Equation 4.4 with a fixed e_m^u as follows. The gradients of the two objective functions can be obtained accordingly:

$$\frac{\partial g}{\partial \theta_k} = \sum_{e_t^u \in L_T^u} w_t(e_T^u, e_t^u)\Big(S_k(e_T^u, e_t^u) - S_k(e_m^u, e_t^u)\Big). \tag{4.11}$$

With this observation, we can now employ the well-studied and efficient iterative convex optimization techniques to estimate the models. In each iteration, given the current θ, we first calculate and fix e_m^u, and then compute the gradient of the objective function accordingly. Similar optimization methods have been employed before in [20], and it is known that such approximation may lead to a local minimum. One possible way to overcome this issue is to execute the optimization process multiple times with randomly initialized θ, and choose the parameters which produce the best results. We compared Powell's method and this method on a small dataset. Similar results are generated separately, which suggests the proposed objective function can be approximately estimated in this way. With the approximated gradient, we use L-BFGS to finally estimate the recommendation models.

4.5 Experiments

In this section, we implement both global and personalized entity recommendation models proposed in Section 4.4 along with several popular and state-of-the-art recommendation methods which fit in the problem definition of this study. We apply these methods on a user log dataset collected from a commercial search engine, and the entity graph extracted from Freebase. We then perform a series of experiments to demonstrate the effectiveness of the proposed models. We present experimental results with discussion and performance analysis in this section.

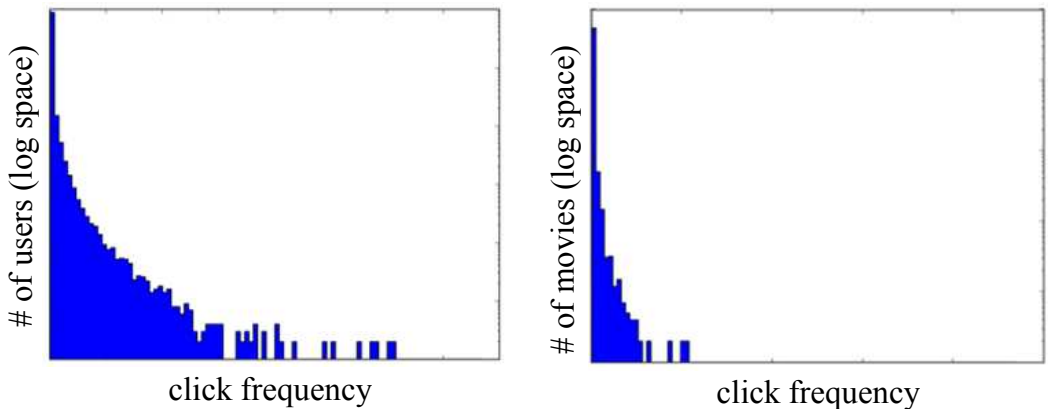


Figure 4.5: User Log Distribution

4.5.1 Datasets

Although the proposed recommendation framework is generic, we take movie related entity recommendation as a case study in the following experiments. We use three months user click log collected from a commercial search engine in 2012 as the user log dataset. We then extracted the corresponding subgraph from Freebase. Note that this extracted Freebase dataset not only contains movie entities, but also contains other related entities as well as the relationships between these entities, including actors, directors, producer, *etc.* Moreover, the URLs related to those movie entities can also obtained in order to map users' clicked URLs to movie related entities. Finally, after jointing the collected user click log and the Freebase entity graph, we sampled approximately $1,000,000$ users with at least one movie related entity page visiting activity in the dataset. Data distribution of the movie entity related user log dataset can be found in Figure 4.5.

To comply with the company's non-disclosure policy, we hide both axis scales of the plots. The first plot is users' entity page visiting frequency (number of users *v.s.* visit frequency) distribution, which follows a power law distribution. One can notice that most users only visited a small number of movies related entity pages in the dataset. Similarly, the second plot, movies' entity page visiting frequency distribution (number of movies *v.s.* visit frequency) is also a power law distribution. From this plot, we can observe that the majority of movie related entity pages have only been visited for a small number of times while only a very small number of movies have been viewed frequently. These two plots demonstrate the sparsity issue of the user log dataset.

4.5.2 Performance Test

We partition the entire user log dataset into two sets: $10,000$ users with their corresponding user log sequences are sampled as the test dataset, and the rest user log data are used as the training dataset. As described in problem definition, we adopt the *leave-one-out* validation method to evaluate the performance, *i.e.*, given the user log sequence L_T^u, which contains entity pages u visited from timestamp 1 to $T-1$, we attempt to predict the entity u visited at T. We call this entity e_T^u the ground truth entity. We use top-10 *mean reciprocal rank* (MRR) as the evaluation metric:

$$MRR = \frac{1}{|D_{test}|} \sum_{i=1}^{|D_{test}|} \frac{1}{rank(e_T^u)}, \tag{4.12}$$

where $|D_{test}|$ is the size of the test dataset and $rank(e_T^u)$ represents the rank of the ground truth entity e_T^u with certain recommendation function. $rank(e_T^u)$ will be 0 if the recommendation model can not rank e_T^u in the top-10 results. Notice that a larger MRR indicates better performance.

Models proposed in this study are implemented as follows:

- Global recommendation model: estimate global recommendation model θ_g using Equation 4.5 and recommend entities with Equation 4.3;
- Personalized recommendation model without neighbor data: estimate personalized recommendation model θ_T^u with Equation 4.8 and recommend entities with Equation 4.7. We abbreviate this method as *PRM*;
- Personalized recommendation model with neighbor data: estimate personalized recommendation model θ^u with Equation 4.8 and recommend entities using Equation 4.9. We abbreviate this method as *PRM-KNN*.

We implement the following popular recommendation approaches besides the proposed methods. Features and models of all methods are learned in the training dataset and MRRs are calculated based on the recommendation performances in test dataset.

- Global Popularity: recommend the most popular (most frequently visited) movies in the entire user log training dataset;

Table 4.4: Experiment Results

Method	MRR
Global Popularity	0.024
Local Popularity	0.043
Cross Domain	0.039
Matrix Factorization	0.160
Co-Click	0.340
Global Model	0.354
PRM	0.361
PRM-KNN	0.451

- Local Popularity: recommend the most popular movies of the day;
- Cross-Domain: recommend movies based on users' non-movie web page visiting log;
- Co-Click: estimate conditional probabilities between entities and recommend movies using Equation 4.1;
- Implicit binary matrix factorization: build binary user-entity implicit feedback matrix with test user log dataset, and apply the matrix factorization method presented in [27], to recommend entities to users.

To make a fair comparison, we apply the same feature set to all three proposed recommendation models. All methods are able to generate a recommendation score given a user log sequence L_T^u and the target entity e_T^u. Ideally we should evaluate all entities with each method and rank these entities accordingly, which can be, however, very time consuming. Also as presented in Figure 4.5, most users are only interested in a small number of entities, which makes evaluating all entities for each user unnecessary and excessive. For simplicity and efficient reasons, each method will only rank a set of most promising candidates, which are pooled by different features. With this evaluation process, the MRR scores reported in this study are the lower bounds of the actual performances of these methods. Performance results of all the recommendation approaches are presented in Table 4.4.

In Cross-Domain method, we estimate the conditional probabilities of movie entities with $3,000$ popular websites. This method gives a similar performance of the popularity based methods. Among these three single feature methods, local popularity approach performs the best with

$MRR = 0.043$. Binary matrix factorization method reaches $MRR = 0.160$ in the dataset. Although matrix factorization performs well in traditional recommendation problem, it could not fully take advantage of the heuristics and features generated from user click log and the entity graph. In order to thoroughly evaluate matrix factorization based recommender systems, we also implemented technique proposed in [35], which utilizes temporal information during matrix factorization process, but the performance does not change as much. One possible explanation is that matrix factorization method heavily depends on the learned user and item factors, U and V; if these two matrices are not accurately learned due to the noises or the data sparsity, adding other factors like temporal information, or other contextual information may not achieve significantly better performance as expected.

Co-Click model (conditional probability between entities) outperforms all baseline methods, which makes it the strongest baseline method for this problem. As stated in Section 4.4, with a sufficient amount of data (approximately 1 million users' log), co-click can be very effective and provide satisfying recommendation results.

The proposed global recommendation model takes advantage of pairwise similarity features extracted from both entity graph and user log. In our experiments, parameters are estimated with *L-BFGS* with a randomly sampled $5,000$ user log sequences. Regularization term λ in Equation 4.5 is set to 0.2 which is determined by cross validation.

Personalized recommendation framework, different from global recommendation model, assigns a different recommendation model to each user. Parameters in each personalized recommendation model are estimated with the target user log sequence L_T^u as well as the neighborhood data. With personalized parameters, users' behaviors and interests patterns can be better interpreted. In this experiment, we use the same set of features when defining personalized models as in the global model. We use θ_g from global recommendation model in the regularization term of Equation 4.7 as well as neighbor similarity function (Equation 4.10). Neighborhood similarity threshold in Equation 4.10 is set to 0.5. In neighbor similarity definition, E represents the entire entity space. For simplicity reason, we only use partial entity space ($5,000$ entities) to estimate neighborhood similarity. This estimation could damage the performance of both personalized recommendation methods (PRM and PRM-KNN).

For computational efficiency, we set the upper bound of neighborhood size of each user to 80, *i.e.*, if the random projection of user log L_T^u is "popular" and a large number of similar neighbors are found, we only randomly sample 80 neighbors when estimating personalized models. Two regularization parameters (λ_1 and λ_2) in Equation 4.8 are set to 0.05 and 0.1 respectively and similar to global recommendation model, these two parameters are estimated with cross validation. We use the global temporal similarity function $w_t(\cdot, \cdot)$ in personalized recommendation models although later experiments suggest personalized temporal similarity function may lead to better performance, since the rates of user interest drift are different.

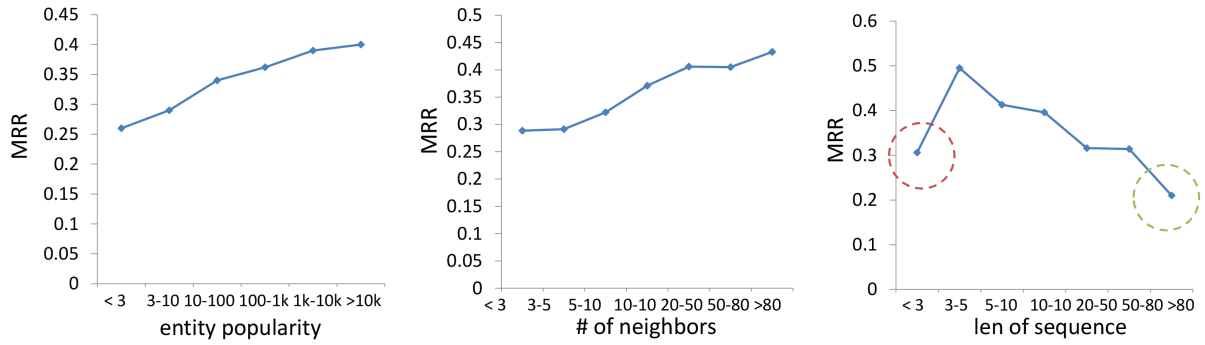
As discussed in previous section, when recommending entities using personalized models, we can either include neighbor data or only use the target user's click log history. Based on experiment results, when only using target user log, the improvement of performance is not as significant compared to global recommendation model (1.98% improvement on MRR). But when including neighbor data during recommendation, a much more significant performance boost can be achieved (27.4%). Analysis and discussions of the personalized recommendation models are presented in the next subsection.

4.5.3 Personalized Recommendation Model Performance

In the following experiments, we analyze the performance change of the proposed personalized recommendation framework in different scenarios.

With Different Entity Frequencies

We first study the correlation between performance of the personalized recommendation method and the frequency of the target movie. We split the test dataset based on the frequency of the target movies and apply personalized recommendation models on each group. The results of this experiment can be found in Figure 4.6(a). Based on the results, one can notice that popular movies (movies which appear frequently in the user log dataset) are easier to predict than other movies. Two reasons may be able to explain this performance change. First, features in recommendation models favor popular movies, *e.g.*, global and local popularity. The co-click feature provides more accurate prediction with popular movies considering the frequency of these movies in the training



(a) Entity Popularity (b) Number of Neighbors (c) Length of Sequence

Figure 4.6: Personalized Framework Performance Analysis

dataset. Second, with sufficient training data for popular movies, high quality recommendation models can be learned in such scenarios, thus MRR will be increased accordingly.

With Different Neighborhood Sizes

The second experiment we conducted on analyzing personalized recommendation framework is to study the correlation between recommendation performance and the number of neighbors that locality sensitive hashing technique can find for different user log sequences. Results are presented in Figure 4.6(b). From the plot one can notice that the more neighbors each user log sequence has, the better recommendation results our framework can provide. Two reasons might be able to explain this phenomenon. First, based on Equation 4.8, more neighbors indicate more data during parameter estimation, so that a higher quality recommendation model can be obtained in this situation. Second, user log sequences with more neighbors, *a.k.a.*, the "popular" user log sequences, are usually associated with popular movie entities. Based on the previous experiment, user log sequences with more neighbors are usually easier to model which leads to a better recommendation results.

With Different Sequence Lengths

The third performance analysis experiment we conducted is to study whether any correlation exists between personalized recommendation performance and the length of the user log sequences. Similar to the two previous experiments, we split the test users into groups based on the lengths of their user log sequences, and apply personalized recommendation models on each group. The

results of this study can be found in Figure 4.6(c). The result plot indicates that the performance of the proposed method peaks when lengths of user log sequences are from 3 to 5. Shorter and longer user log sequences lead to compromised performances. When a user log sequence is too short (red circle in the plot), not enough data are available when estimating parameters, which could damage the quality of the personalized models. When a user log sequence is too long (green circle in the plot), user's interests may drift overtime severely, and such information varying can be hard to capture with a global temporal similarity function. Another possible explanation for the compromised performance when user log sequence is long, is the existence of Internet-addict users. These users visit a large number of entity pages with a broad interests, which makes building personalized recommendation model for these users very challenging.

One might argue that, since performance peaks when user log sequence length is in the range of 3 to 5, old visiting events might not help as much as the most recent 3 to 5 web page visiting events. To verify this hypothesis, we chop all user log sequences to length 3 (only preserve the most recent 3 web page visiting events) and apply personalized recommendation method on this dataset. MRR dropped to 0.114 which is much lower than the best performance we can obtain when using all user web page visiting events. This experiment demonstrates that previous behavior does help entity recommendation in the near future. As we stated before, personalized temporal similarity function might be able to capture the each user's interest drift more accurately, with which recommendation performance can be improved.

This performance analysis experiment also explains the performance improvement of personalized recommendation method when using neighborhood data during recommendation. As discussed above, the compromised performance in red circle in Figure 4.6(c) is due to lack of data, and as shown in Figure 4.5, the majority of users only have a very limited number of web page visiting events so these users belong to the red circle. Adding neighborhood data during recommendation helps lessen the data sparsity problem so that it can improve the performance for most users resulting the significant result improvement in Table 4.4.

4.6 Conclusion

In this work, we study to bridge the gap between search engines and recommender systems. We propose the entity recommendation problem by utilizing user click log and the Freebase entity graph. We present a generic entity recommendation framework which utilizes various pairwise similarity features extracted from both user log dataset and the entity graph. The proposed recommendation framework takes advantage of the consistency and drift natures of user interest, different types of entity relationships as well as several other heuristics, which are crucial to build such a recommendation system. During empirical studies, we compared our proposed methods with several existing popular and state-of-the-art recommendation approaches and demonstrate the effectiveness of our method. We also analyze the performance of our methods under different scenarios, explain the philosophies behind the proposed models and discuss possible revisions to improve performances in special cases.

With the fast development of search engines and recommender systems, studies on bridging these two popular systems as does this study could benefit both applications and vastly improve user experience when employing such hybrid system. Interesting future studies include recommender techniques which can incorporate users' feedback on-the-fly, systematic pairwise and non-pairwise feature generation given a new domain, as well as a revised on-line version of the framework in which parameters can be estimated efficiently in almost real time.

Chapter 5

Citation Prediction in Bibliographic Networks

5.1 Overview

Searching for related scientific literature is the first and an essential step for nearly all scientific research disciplines. Researchers want to find highly related publications to their own research fields and topics, so that they can learn from related works, compare with previous methods as well as develop new research ideas. However, with the rapid development of science and engineering, a gigantic number of research papers are published each year on various research topics and areas. It is impossible for researchers to follow or read all publications in his/her research fields. Hence a system which could help scientific researchers organize relevant publications is in high demand. Such a system should be able to retrieve high quality publications given research topics, and also measure the relevance between existing publications and the researchers' current works.

Google Scholar, PubMed and other key-word-based literature search engines allow users to query publications based on key-word and other related information, *e.g.*, authors, time period of publication. They also provide related articles by measuring document similarity between papers. Although these systems find relevant papers and make literature search easier than before, key-word-based approach still returns thousands or millions of relevant papers. For example, Google Scholar returns more than 2 million papers with the query "link prediction", and more than 5 million results with the query "citation". Researchers can easily be overwhelmed by this huge number of relevant papers. Instead of going through a large number of papers which match key-word queries, researchers prefer to only review a relatively smaller number of publications closely related to their research topics, research communities and fields, and of high quality, so that they can read these papers and even use them as references directly. To meet this requirement, we here study citation prediction problem in bibliographic information networks. Aiming to help researchers find highly

related publications effectively and efficiently, we propose a new citation prediction model and use this model to answer citation queries.

Citation prediction aims to find the potential citation relationships in bibliographic networks. Yet it is different from other link prediction problems. For example, citation relationship is directed, while friends recommendation or co-authorship prediction are often undirected. Links among co-authors in publication datasets tend to form communities and have high structural locality, because authors tend to collaborate with researchers within their own research group or with researchers they collaborated before. However, high quality and relevant papers can be from anywhere. Due to the evolution of on-line library systems, scientific researchers can easily get access to nearly all available digitized publications. They can find and review any relevant previous paper on bibliographic networks, which leads to a relatively sparse yet even distribution in citation link space. Traditional link prediction methods [21, 40] commonly rely on the aforementioned locality assumption, which makes these methods ineffective for citation prediction.

On the other hand, citation prediction methods should be able to measure document similarity and capture topic distribution in bibliographic information networks. However, solely relying on topic modeling methods is not sufficient for citation prediction either. Although the number of papers is tremendous, research topics are comparably limited. Hundreds or even thousands of papers could share the same topic, which makes topic similarity a very weak evidence in citation prediction. Additionally, many critical features which might be more related to citation prediction cannot be represented by topic similarity easily. For example, if one paper is written by a well-known researcher in the field, the probability of this paper getting cited by a future publication is higher than a paper by a new researcher. Similarly, ranking of the publication venues (conferences / journals) and the reputation of research groups all affect citation probability. Furthermore, researchers tend to cite papers of their own, papers within their research groups, or papers of their peers for different reasons. All these heuristics are hiding among bibliographic network structure, and these information cannot be represented using topic similarity.

In this chapter, we study how to predict citation relationship in bibliographic information network effectively and efficiently. We propose a novel two-step approach, attempt to capture both topic and document similarities as well as hidden network structures that are sensitive to citation

relationships. With this approach, we setup a citation query system. Given author information, target publication venues and certain text description, *e.g.*, title and abstract, of a query paper, our citation query system searches papers in a publication network and returns a list of relevant papers, ranked by the probability of being cited by the query paper. In order to answer citation queries fast and accurately, we first build discriminative term buckets, which is a data structure that can capture document and topic similarities without breaking possible citation relations. We then assign papers into different buckets, which reduces search space for both model learning and citation query answering. Second, we set up a meta-path-based feature space to interpret hidden network information in bibliographic networks, and define citation probability with meta-path-based features. With the help of discriminative term buckets and meta-path-based feature space, it is now possible to learn a citation prediction model and use this model to answer citation queries.

The major contributions of this chapter are summarized as follows:

- We propose the problem of citation prediction in a bibliographic network, and analyze the differences between this problem and the related work, *e.g.*, traditional link prediction solutions.
- We propose a new data structure namely discriminative term buckets in order to capture both document similarity and potential citation relationship.
- We propose to use a meta-path-based feature space to interpret structural information in citation prediction, and define citation probability with these feature.
- Experiments on real dataset show that we can predict citation relationship with high accuracy and efficiency compared with the state-of-the-art link prediction methods.

In the rest of the chapter, we first discuss discriminative term bucket data structure and present how to construct term buckets efficiently in Section 5.2. Meta-path-based feature space building and citation prediction model learning are described in Sections 5.3 and 5.4. Experiments and results are presented followed by the conclusions of this study.

5.2 Discriminative Term Bucketing

For most citation relations, the prerequisite is a positive topic and / or term correlation between two papers, *i.e.*, these two papers belong to the same research area, share similar research topics or try to solve a similar problem. Therefore, the first step of our citation prediction framework is to catch such topic or term correlation and measuring document similarity in the DBLP information network. Topic modeling methods are often chosen to achieve this requirement. By comparing topic distributions of paper pairs, one can calculate document correlation accordingly. However, topic modeling methods might not be suitable for citation prediction for two reasons. First, topic granularity is hard to determine in an unknown document collection. Second, topics might not be citation relationship discriminative. For example, if two papers belong to some background topics, which have high weights towards common words, this topic similarity cannot be used as an evidence for citation prediction.

Aiming to capture citation discriminative term correlation and measure document similarity in the DBLP information network, we propose a novel method named discriminative term bucketing (we refer this method as term bucketing in the rest of the paper). Very similar to the input and output of topic modeling methods, given a certain document collection $\mathbf{D}$, term bucketing generates a number of buckets (similar to topics), each of which contains a set of terms. Papers in the collection can be distributed into different term buckets based on their terms. Moreover, one paper can belong to multiple buckets. Within each bucket, papers have a positive document similarity. The citation relationship probability is also higher among papers which share buckets than papers which do not.

Discriminative term bucketing contains three steps. First, we identify discriminative terms using a link space discriminative measure. This measure can find terms which are sensitive to citation relationships. Second, we treat such terms as the seeds of the buckets, and expand each bucket by adding in similar terms. Finally, we assign the entire paper collection into the bucket data structure.

In order to measure each term's ability of identifying citation relationships, we first define citation discriminative measure. By generating term paper inverted index in $\mathbf{D}$, for each term t, we collect all papers with t, and denote this paper set as P_t. We then treat each paper in P_t as

a node, and build a complete graph. If two papers have a citation relationship, the link between them will be marketed as positive, otherwise negative. We then define the positive-negative ratio as the citation discriminative measure (CDM) for terms as in Equation 5.1.

$$\forall t \in \mathbf{T}, CDM_t = \frac{count(G_t, +1) + 1}{count(G_t, -1) + 1} \tag{5.1}$$

where $\mathbf{T}$ is the training document collection for citation prediction, G_t is a $|P_t| \times |P_t|$ matrix which contains link labels in the complete link space and $count(G, label)$ counts the number of links in G that equals to $label$.

After calculating CDM for each term, we can pick up terms with CDM higher than a pre-defined threshold and use these terms as discriminative bucket seeds. Each seed term t defines a discriminative term bucket B_t. One should notice that, CDM is calculated on the training dataset. However, in order to reduce search space in both model training and query answering processes, we need to categorize all papers in both training and test datasets and put them into corresponding term buckets. Training and test datasets are independent so that term distribution over these two sets might be different. It is possible that discriminative terms generated in the training set do not exist in the test set. So if term buckets only contain terms in the training dataset, categorizing test papers will be difficult. In order to propagate citation discriminative information to the test set, we use term expansion technique to find more terms in $\mathbf{D}$ (from both training and test sets) for each bucket.

We expand each term bucket by introducing more terms which have high mutual information with the bucket seeds. Mutual information can be used to measure the dependence of these two terms. The heuristic is that seed terms are terms with citation information. If other terms are highly dependent on seed terms, they should contain citation information as well. For a specific discriminative term bucket B_{t0} which contains bucket seed t_0, we iterate all terms in document set $\mathbf{D}$, and calculate mutual information between each term t and t_0 using Equation 5.2:

$$\forall t \in \mathbf{D}, I(t_0, t) = Pr(t_0, t) log(\frac{Pr(t_0, t)}{Pr(t_0)Pr(t)}) \tag{5.2}$$

where $\mathbf{D}$ is the entire document collection, $Pr(t_0, t)$ is the probability of both t_0 and t appear in

one document. This probability can be estimated using the number of documents which contains both t_0 and t and the total number of documents in the collection. Similarly, $Pr(t_0)$ and $Pr(t)$ are the marginal probability density functions of terms t_0 and t respectively.

For each term bucket, we select terms which have high mutual information score with the term seed using a pre-defined threshold MI. By adding these terms into term buckets, now each term bucket has multiple discriminative terms. Based on the term distribution, we can categorize both training and test papers into different buckets by checking whether a certain paper contains one or more terms in a bucket. One should notice that one paper can be assigned to more than one term buckets. Within each bucket, papers share terms as well as topic information so that they have high document similarity. Within the same buckets, papers have a higher probability of been cited by other papers. Term bucketing partitions the entire paper dataset into different buckets, and our citation prediction framework will only search within relevant buckets while learning prediction model and answering citation queries. This approach reduces search space. Based on our experiments, this approach improves both accuracy and efficiency.

One should notice that, MI is an important parameter in citation prediction framework. If MI is too low, the number of terms in each bucket will increase severely, which will increase search time complexity. If MI is too high, the number of terms in each bucket will decrease, which means the ability of capture potential citation relations will be compromised. We will discussion this issue with experiments in Section 5.5.

5.3 Meta-Path-Based Feature Space

After constructing discriminative term buckets and categorizing both training and test papers into corresponding buckets, we introduced both document similarity as well as citation information into our framework. Papers within the same bucket or share a number of buckets usually share similar research topics and also have a higher probability to be cited by one another. Although term bucket structure helps reduce search space, the number of search candidates is still very large. We need to utilize structural features to capture more citation information and improve the prediction accuracy. In this section, we discuss meta-path-based features in the DBLP information network. We will introduce citation prediction model learning in the next section.

With meta-path and meta-path-based measurements defined in Chapter 2, a meta-path-based feature space $\mathbf{F}$ can be represented as a Cartesian product of the two sets:

$$\mathbf{F} = \mathbf{P} \times \mathbf{M} \tag{5.3}$$

where $\mathbf{P}$ is the set of possible meta-paths and $\mathbf{M}$ is the set of possible meta-path-based measures.

In a small heterogeneous network with a simple schema, in order to generate a comprehensive feature space, one can enumerate all meta-paths with a length constraint. However, it is impossible to permute all meta-paths in general. It is also not necessary to generate all meta-paths since some paths does not carry sufficient semantic meanings as others. In this project, rather than calculating meta-path-based features along all meta-paths, we select a subset of meta-paths with clear semantic meanings and use this subset to define citation probability.

5.4 Learning Citation Prediction Model

After building discriminative term buckets and meta-path-based feature space, we can now define the citation prediction model accordingly.

5.4.1 Citation Probability

As discussed in Section 5.1, citation probability should be a combination of both document similarity and structural information in a publication network. By utilizing discriminative term buckets and meta-path-based features, we can define citation probability as follows:

$$Pr(label = 1|p^{(1)}, p^{(2)}; \boldsymbol{\theta}) = \frac{e^z}{e^z + 1} \tag{5.4}$$

where $z = \Sigma_{f_i \in F\prime} \theta_i \cdot f_i(p^{(1)}, p^{(2)})$. $Pr(label = 1|p^{(1)}, p^{(2)}; \boldsymbol{\theta})$ is the probability that paper $p^{(1)}$ cites paper $p^{(2)}$. In the definition of citation probability, F' is the feature space defined on the DBLP heterogeneous information network in order to capture citation related information. We define $F' = F \cup F_0$, where F is the meta-path-based feature space from equation 5.3 and F_0 is a set of numerical features for target paper candidates. θ_i is a normalized weight value for feature f_i which indicates

which feature is more important for citation prediction. Notice that we only use this definition when $p^{(1)}$ and $p^{(2)}$ belong to one or more term bucks, otherwise $Pr(label = 1|p^{(1)}, p^{(2)}; \boldsymbol{\theta}) = 0$.

In order to generate a more comprehensive feature space, we add numerical features which can not be represented with meta-paths, including average H-Index [24] and publication venue ranking [61], to paper candidates. These two numerical features help boost prediction accuracy as supplements to meta-path-based feature space. H-Index measures the productivity and impact of the published works of a scholar, and publication venue ranking measures the reputation and the quality of the research paper indirectly. Both measures should be positively correlated with citation relationship. If the authors of one paper have a higher average H-Index and this paper is published in a highly ranked conference, the probability of this paper being cited by the query paper would be higher.

In order to learn the citation prediction model, we generate the training dataset which contains positive and negative examples of citation relations. However, the DBLP information network is extremely large and the citation relations are very sparse and limited. So search on the entire paper network can be time consuming and ineffective. Randomly generated negative examples can be arbitrary and contain very little representative information. In order to collect high quality training dataset, we use discriminative term buckets as a filter to first reduce the size of the information network, and only generate positive and negative training examples within term buckets. In this way, both positive and negative paper pairs have high document similarity and also high probability of forming citation relationships. Learning models on such training dataset improves the quality of the citation prediction model compared with randomly generated training dataset over the network.

We define the training dataset as follows:

$$\mathcal{T} = \{(p_i^{(1)}, p_i^{(2)}, label)|\exists B_t, p_i^{(1)} \in B_t, p_i^{(2)} \in B_t\} \tag{5.5}$$

where $p^{(1)}$ and $p^{(2)}$ are papers in the information network, and they should both belong to at least one discriminative term bucket.

In order to learn citation prediction model, we use logistic regression with L_2 regularization to

estimate the optimal $\boldsymbol{\theta}$ given a training dataset $\mathcal{T}$.

$$argmin_{\boldsymbol{\theta}} \Sigma_{i=1}^{n} - \log Pr(label|p_i^{(1)}, p_i^{(2)}; \boldsymbol{\theta}) + \lambda \|\boldsymbol{\theta}\|_2^2 \tag{5.6}$$

With the objective function above, weights in the citation probability model can be estimated with standard optimization methods. We use MLE (Maximum Likelihood Estimation) in our experiments for parameter learning.

5.4.2 Citation Prediction Model

After learning the citation probability defined in Equation 5.4, we can now define a citation prediction model, and use this model to prediction possible citation relations or answer citation queries. The citation prediction model is defined in Equation 5.7:

$$cs = log(1 + cbn(p^{(1)}, p^{(2)})) \cdot Pr(label = 1|p^{(1)}, p^{(2)}; \boldsymbol{\theta}) \tag{5.7}$$

where cs is short for citation score, and $cbn(p^{(1)}, p^{(2)})$ defines the number of common discriminative term buckets shared by papers $p^{(1)}$ and $p^{(2)}$.

Given a citation query paper p^*, we first look up p^* in discriminative term bucket set $\mathcal{B}$, and find all buckets containing paper p^*. We define this subset as B_{p*}. We then collect the papers belonging to the term bucket subset B_{p*}, and denote this paper subset as $P(B_{p*})$. For each paper p in $P(B_{p*})$, we calculate $cs(p*, p)$. By ranking paper set $P(B_{p*})$, the citation prediction framework can generate a ranked list of papers as the answers to query p^*.

One should notice that, we only calculate cs between p^* and papers which at least share one term bucket with p^*, which is only a subset of the entire paper dataset. In the next section, we conduct a set of experiments and compare our citation prediction framework with the state-of-the-art link prediction methods, which shows that our approach can find citation relations accurately and efficiently.

5.5 Experiments

We apply our citation prediction approach along with two state-of-the-art link prediction methods on a DBLP citation dataset generated by Tang *et al.* [62] in this section and compare their performances.

5.5.1 Dataset and Methods Setup

The original DBLP dataset does not contain citation relations. Tang *et al.* extracted citation information from other sources and generated a DBLP citation dataset. We use the citation information in this dataset as training examples as well as ground truth to verify different methods. Instead of using the entire dataset, we generated a subset which contains 464 publication venues, 29,615 papers and 215,502 citation relations. Papers in this subset focus on one of the four areas: data mining, database, information retrieval and artificial intelligence. Due to the high coherence of these research areas, citation relation distribution in this subset is scattered, which makes the task difficult and challenging. We convert this subset into a heterogeneous information network. This information network contains authors, papers, publication venues, and terms as entities (publication year as attribute), as well as paper author relations, paper venue relations, paper term relations as well as citation relations as links, all together 83,625 entities and 682,725 links.

In this study, in order to comprehensively describe different relationships between paper entities in the DBLP heterogeneous information network, we utilize seven different meta-paths between paper entities, which are $P-A-P$, $(P-A-P)^2$, $(P-A-P)^3$, $P-C-P$, $P-T-P$, $(P-T-P)^2$ and $(P-T-P)^3$. We use two meta-path compatible measures, which are newly proposed PathSim [59] and random walk with restart. By combining seven meta-paths and two meta-path-based measures together, we get total 14 different meta-path-based features. As mentioned in the previous section, we also use two numerical features in our feature space in order to have certain bias towards target paper candidates, which are average author H-Index and publication venue ranking score. In discriminative term bucket building step, in order to capture most term information, we use 0 as citation discriminative measure threshold (Equation 5.1) and 0.0003 as mutual information threshold during term expansion (Equation 5.2). With this setting, discriminative term buckets can capture nearly 90% of citation relations, *i.e.*, for each citation relation in 90% of the total relations,

the two papers which defines this citation can be found in at least one term bucket. Moreover, search space can be reduced by 40% if our citation prediction framework only searches within term buckets.

The state-of-the-art link prediction methods which we compare with our framework in these experiments are personalized PageRank (denoted as *pp* in figures and tables) [13] [71] and path-constrained random walk [38] (denoted as *rw* or referred as trained random walk in figures and tables). Personalized PageRank is a very popular similarity measuring method which has been widely applied in different applications including link prediction, friend recommendation, as well as network clustering. As an unsupervised method, personalized PageRank simulates information passing along links between entities, and estimates similarity by calculating reachability from query node to the other nodes in the network. Similar to our approach, path-constrained random walk method is a supervised method, which first calculates random walk similarity along different paths, and then assigns different weights to different paths by learning with provided examples. We also use random walk features along different paths as part of our meta-path-based feature space. In order to have a fair competition, we use the same set of random walk features in both our approach and path-constrained random walk, and also we use the same training dataset and learning method as well.

5.5.2 Measure as Classification Problem

We first compare our method with path-constrained random walk by modeling citation prediction as a classification problem. Considering the sparseness of citation relations on the entire search space, we first generate a biased sample on $\langle paper, paper \rangle$ search space as defined in the previous section. In the biased sampled data, we have 45% positive labels, *i.e.*, paper pairs which actually define citation relations, and 55% negative labels, *i.e.*, paper pairs which do not possess citation relations. In order to measure the prediction accuracy, we use 5-fold cross-validation to assess the quality of each method. The average precision on training and test datasets can be found as follows:

Table 5.1: Performance using Classification measures

Methods Precision	Training	Test
Trained Random Walk	0.7168	0.6691
Our Method	**0.7555**	**0.7533**

Table 5.2: Performance as Query Processing on DBLP Network

Methods	Group 1			Group 2			Group 3		
	prec@10	prec@20	recall@50	prec@10	prec@20	recall@50	prec@10	prec@20	recall@50
trained rw	0.2000	0.1250	0.1483	0.0857	0.1000	0.1314	0.2167	0.1750	0.1467
pp	0.1833	0.1333	0.1567	0.1143	0.1071	0.1529	0.2000	0.1667	0.1567
w/o bucket	**0.2333**	0.1333	0.2000	0.1429	0.1143	0.1643	**0.3000**	**0.2000**	0.1717
bucket	**0.2333**	**0.1417**	**0.2533**	**0.1714**	**0.1214**	**0.1771**	0.2833	**0.2000**	**0.1867**

Based on Table 5.1, our method outperforms path-constrained random walk in both training set and test set. Both approaches are trained using the same dataset with the same learning method. Compared with path-constrained random walk approach, our method has a larger and more comprehensive hybrid feature space, which contains both meta-path-based features as well as numerical features on target paper candidates. By generating features from a uniform meta-path-based feature space, our approach is capable of capturing more information from the sampled dataset and improves the average precision in training data by 4% and increases average precision in test data by 8.4%.

5.5.3 Measure as Query Problem

Measuring citation prediction as classification simplifies the problem. During biased sampling, we reduced the search space, so both methods can achieve high precisions. However, if we model citation prediction as query problem, *i.e.*, given a paper with author(s), target publication venue(s), abstract as well as publication time stamp, one approach should return a list of previous publications ranked by the probability of being cited by the query paper, the citation prediction problem becomes citation query processing. This problem is a much more difficult than classification, simply because the search space for possible citation paper candidates becomes all the papers in the DBLP information network. In this experiment, we test our approach along with the two link prediction problems by experimenting citation prediction as query processing problem.

Path-constrained random walk and our approach are both supervised, so training datasets need to be generated first before query processing. We randomly partition paper node space in the DBLP information network into five folds. We use four folds as training set, and the rest as test. In this way, we can make sure that, during query processing, all test query nodes are new to all the

ranking models, since we are testing both supervised and unsupervised methods. While searching for citation relations for an unseen paper query during testing, we should search the entire network instead of only within test set, which means, if one approach can find a paper in training dataset, which is cited by the citation query paper, this counts as a hit.

In our approach, we first build discriminative term buckets using training dataset, and then add test papers into the buckets by applying term expansion technique. While generating training dataset, we only focus on positive links and negative links within the same term bucket. While answering queries, very similarly, we only search the term buckets which contain the query paper instead of searching the entire DBLP information network. We use the biased random sampling technique to generate training dataset for the path-constrained random walk approach, and during query answering, the ranking model learned by path-constrained random walk searches the entire network for possible citation relations. Personalized PageRank does not require training, so this approach simply calculates similarity score between the query paper and all other papers using the DBLP network structure, *i.e.*, along paper-venue links, paper-term links as well as paper-author links, and return papers with the highest similarity as the query results. To further demonstrate the power of the discriminative term bucketing technique, we add another competitor method, which uses the same feature space as our method, but searches the entire paper set in the DBLP information network. To distinguish these two methods, we call our method meta-path-based citation prediction framework with discriminative term bucketing (denoted as bucket in tables and figures), and we refer the new competitor method as meta-path-based citation prediction framework without term bucketing (denoted as w/o bucket in tables and figures).

We randomly pick 19 query papers from the test set, and divide them into three groups based on the number of citation relations associated with them. Group 1 contains 6 papers, each of which cites less than 20 papers, group 2 contains 7 papers, each of which cites 20 to 30 papers, and group 3 has 6 papers, each of which cites more than 30 papers. The query processing performance results can be found in Table 5.2, Figures 5.1(a), Figure 5.1(b) and Figure 5.1(c).

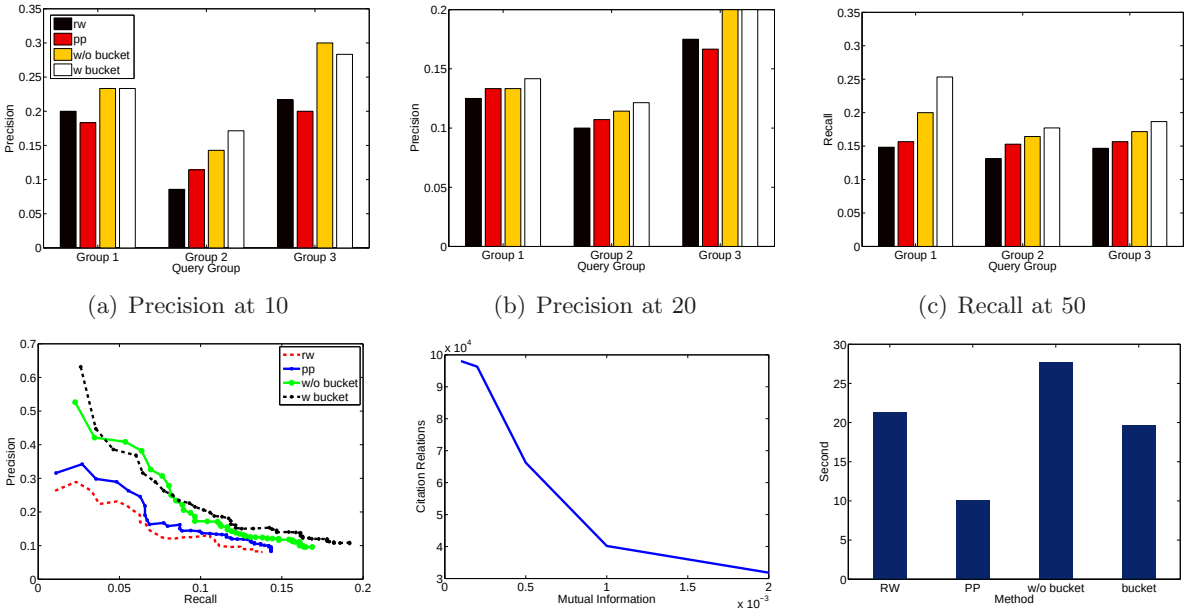
We use three query processing measures to evaluate the performance of each method, which are precision at top 10 query results, precision at top 20 query results and recall at top 50 query results, denoted as prec@10, prec@20 and recall@50, respectively. Based on these measurements,

one can notice that, our methods can find more citation relations than link prediction methods in general. For example, our methods improve recall@50 by 10% in query group 1 compared with link prediction methods, and also increase prec@10 by $7 - 8\%$ in query group 3. Discriminative term bucketing technique helps our method reduce search space by around 40% on average. As we can see in Figures 5.1(a), 5.1(b) and 5.1(c), by eliminating irrelevant citation candidates, meta-path-based prediction model with bucketing outperforms the one searches the entire publication network. Another interesting observation is, personalized PageRank gives a relatively better performance than path-constrained random walk method. The reason is, in path-constrained random walk training process, we only use short meta-paths (length up to 3), so path-constrained random walk model is only able to reach its neighbors which are three steps away from the queries nodes, while personalized PageRank can reach all possible papers on the network since the calculation does not stop until similarity vector converges. This proves our observation in Section 5.1, which is citation relations does not have high locality as other links, and cited papers can be from anywhere in the DBLP information network.

We also use precision-recall plot to demonstrate a more comprehensive comparison of these four different methods in Figure 5.1(d). Based on this plot, we can conclude that meta-path-based prediction model with bucketing gives a good performance overall. The precision of this method can achieve almost 70% when the recall is low (*e.g.*, when we only need top-1 or 2 results). While at the same recall level, link prediction methods can only achieve precision level around 30%. However, the meta-path-based precision model without bucketing outperforms our method when the recall is around 5%, which suggests that only search citation paper candidates within the same buckets as query paper still loses changes of finding citation relations with low document similarity correlation of the query papers (bucketing can only capture 90% citation relations).

5.5.4 Parameter Turning and Time Complexity

As discussed in Section 5.2, mutual information threshold MI controls the size of the entire term bucket set. The lower this threshold is, the more terms and related papers can be introduced into each bucket. If this threshold is too low, the effect of search space reduction of the term bucketing will disappear, which will increase query processing time. On the other hand, if the



(a) Precision at 10 (b) Precision at 20 (c) Recall at 50

(d) Precision Recall (e) Parameter Tune (f) Running Time

Figure 5.1: Performance: (a) Precision@10; (b) Precision@20; (c) Recall@50; (d) Precision Recall Curve; (e) Parameter Tuning; and (f) Running time.

mutual information threshold MI is too high, the number of papers in each bucket will reduce which can break potential citation relations and reduce the performance of the framework. We here study the relationship between mutual information threshold MI and the number of citation relations preserved in term buckets. From Figure 5.1(e), we can see that, with the increase of mutual information threshold, the number of citation relationship we can capture in term buckets decreases exponentially, and search space will be shrinking quickly as well. In our experiment, by parameter tuning, we choose 0.0003 as our mutual information threshold. Using this setting, discriminative term buckets can capture around 90% of citation relations and reduce search space by around 40%, which makes a good balance between the precision upper bound and the size of the search space.

We also studied query processing time using four different methods. In order to answer citation queries efficiently, we need to first train ranking models for supervised methods and calculate associated features off-line. On-line query processing time is related to a couple of factors. The first one is the number of features in the prediction model. The second is the size of the search space. We recorded the average query processing time for all four methods, and the result can be

found in Figure 5.1(f). From the plot we can see that, personalized PageRank is the most efficient method, and it takes less than 10 seconds to search through the entire network and generate citation relations given a citation query, the reason why this method is fast is because it only uses one feature, which is the personalized PageRank score calculated in the entire network. In both meta-path-based prediction methods, we have all together 16 features to process for each citation candidate. However, with discriminative term bucketing technique applying to our ranking model, we do not need to go through the entire DBLP information network anymore, so the query processing time is only around 20 seconds. The query processing time of meta-path-based prediction model without bucketing is 28 seconds which is significantly slower.

5.6 Conclusion

In this chapter, we propose the problem of citation prediction in the DBLP heterogeneous information network. We proposed a two-step approach to answer citation prediction queries effectively and efficiently. By building discriminative term buckets, our approach first eliminates irrelevant paper candidates, and reduces search space. Then we define a hybrid feature space to fully capture citation sensitive signals, which includes meta-path-based features and numerical features. By learning citation probability model within the reduced search space, we can define a citation prediction model using both citation probability and the number of common term buckets shared between query paper and citation candidate.

After learning the citation prediction model, given a query paper as input, our framework first assigns query paper to one or more term buckets, and then generates a set of citation paper candidates by merging papers from related buckets. Citation scores are calculated and assigned to the candidates.

Empirical study shows that our approach can find citation relations with much higher accuracy compared with traditional link prediction methods. Also, by comparing our method with a similar meta-path-based citation prediction approach without bucketing technique, we demonstrate the power of discriminative term bucketing technique, which can reduce search space and improve prediction precision at the same time.

Interesting future work includes citation prediction study on different information networks,

e.g., predicting retweet relations on Twitter, exploring new feature selection methods [18] [19], and meta-path-based feature space index technique to further improve the query processing efficiency.

Chapter 6

User-Guided Entity Similarity Search

6.1 Overview

Among different entity search problems in information networks, entity similarity search, which takes entities (nodes in the network) as examples in the query, and returns relevant entities by measuring similarity across the network, has gained more attentions due to its wide applicability in different domains. Previous studies on query answering in graphs or networks usually follow traditional learning-to-rank procedure and build ranking models by utilizing similarity measurements like Personalized PageRank [31, 63, 13] or SimRank [30], Such studies usually focus on one specific search task, *e.g.*, friends recommendation [71], or co-authorship prediction [58]. A unified entity similarity search framework, which can first understand the similarity semantic of the queries, and then select relevant ranking models to answer the queries, can provide better user experiences compared with vertical search engines. Moreover, users might prefer to express their similarity search requirements using examples rather than explicitly input search intents or look for specific vertical search engine applications. To build such a similarity semantic meaning aware search system, we need to first study the similarity semantic ambiguity problem in heterogeneous information networks.

Examples of similarity query ambiguity can be found in Figure 6.1. From these examples, one can tell similarity queries which share the same format can have different semantic meanings. Both Query 1 and Query 1' aim to find authors similar to "Christos Faloutsos", however, if we use the same ranking function to answer both queries, the results might not be satisfying. In Query 1, the hidden similarity semantic meaning is described by two similar entities "Jimeng Sun" and "Hanghang Tong". Judged with human knowledge, both authors were frequent collaborators of Faloutsos. When users issue Query 1, they likely are looking for other collaborators of Faloutsos.

Query 1	find author similar to "Christos Faloutsos" taking "Jimeng Sun", "Hanghang Tong" as examples	Query 2	find movie similar to "the Dark Knight" taking "Batman Begins", "Batman" as examples
Answer	Agma J. M. Traina, Spiros Papadimitriou, Jure Leskovec	Answer	Batman Returns, Batman Forever, Batman: Mask of the Phantasm
Query 1'	find author similar to "Christos Faloutsos" taking "Philip S. Yu", "Jiawei Han" as examples	Query 2'	find movie similar to "the Dark Knight" taking "Hancock", "The Curious Case of Benjamin Button" as examples
Answer	Hector Garcia-Molina, H. V. Jagadish, Divesh Srivastava	Answer	Iron Man, Cloverfield, Indiana Jones and the Kingdom of the Crystal Skull

Figure 6.1: Entity similarity search with examples

However, in Query 1', users provide "Philip S. Yu" and "Jiawei Han" as similarity examples. In this scenario, users might be looking for other reputed data mining researchers similar to Faloutsos.

Similarly, although both Query 2 and Query 2' are searching for movies similar to "the Dark Knight", Based on the provided examples, one can tell that Query 2 are searching for Batman themed entities, while Query 2' might be looking for movies with similar popularity and similar genre. We argue that a single ranking model cannot accommodate different similarity semantic meanings behind queries, and thus it can not resolve the query ambiguity problem. In order to find accurate search results for users, we need to first understand the hidden similarity semantic meanings of each query. Understanding search intents is an important task in text search engines [4] [22]. However, to the best of our knowledge, this problem has not been studied in heterogeneous networks before. In this study, we address the similarity ambiguity problem with a similarity semantic aware entity search framework, which first tries to understand the user intents behind the similarity queries, and then select related ranking models to generate final results.

The major contributions of this study are summarized as follows:

- We study the similarity query ambiguity problem in heterogeneous information networks, and propose to use meta-path-based similarity feature space to interpret different similarity semantic meanings.
- We design entity ranking model ensemble and ranking model selection algorithm which can

choose semantically related ranking models for a given query.

- Empirical studies on DBLP and IMDb demonstrate the effectiveness of our unified search framework.

The rest of the chapter is organized as follows, we first provide the problem definition in Section 6.2. We next review meta-path-based feature space feature selection and retrieval model learning in Section 6.3 and Section 6.4. The ranking model selection method is introduced in Section 6.5. Experiments and results are presented at the end of this chapter with discussions and conclusions.

6.2 Problem Definition

With the definitions and examples of heterogeneous information networks defined in Chapter 2, we now define the user-guided entity similarity search problem in heterogeneous information networks as follows:

Definition 6.1 (User-Guided Entity Similarity Search) *Given a heterogeneous information network G, the entity similarity search problem is to find a list of entities similar to a query entity, q. To eliminate the query similarity ambiguity and better understand the query intent, users can also provide one or more example entities of the same type as user guidance. We denote such user provided similarity examples as $e_1, e_2, ..., e_m$.*

One can notice that, a similarity semantic aware search engine should first understand the hidden meaning of a query q and the examples $e_1, e_2, ..., e_m$, and then answer the query accordingly with the correct ranking models.

6.3 Meta-Path-Based Feature Space

In order to build a semantic meaning aware entity search framework, we first investigate the representation of different entity similarity semantic meanings.

With meta-paths and the meta-path compatible measurements defined in Chapter 2, a meta-path-based feature space $\mathbf{F}$ can be represented as a Cartesian product of the two sets as follows:

$$\mathbf{F} = \mathbf{P} \times \mathbf{M} \tag{6.1}$$

where $\mathbf{P}$ is the set of possible meta-paths and $\mathbf{M}$ is the set of possible meta-path-based measurements. Meta-paths represent and carry different similarity semantic meanings, while different measurements quantify the similarity using different methods. Combinations of meta-paths and meta-path-based measurements could be used to measure the similarity between two entities in a heterogeneous information network from different semantic perspectives.

In a heterogeneous network with a simple schema, it is possible to enumerate all possible meta-paths with a reasonable length constraint. However, when the network schema is more complicated, enumerating meta-paths could be very inefficient. In this work, we manually select a set of meta-paths when building the meta-path-based feature space. However, other feature selection method [18] or simply enumerate meta-paths can be used as well.

6.4 Similarity Semantic Meaning Aware Ranking Models

With the meta-path-based similarity feature space, we now have methods to represent different entity similarity semantic meanings, and with these features we can measure entity similarity from different semantic perspectives quantitatively.

6.4.1 Ranking Model Definition

In order to learn a high quality ranking model for each similarity semantic meaning, instead of using the entire meta-path-based similarity feature space, we first apply feature selection process to generate a feature subspace which contains only features relevant to the target similarity semantic. We then build linear ranking models using the selected meta-path-based features, and learn the parameters using training datasets. We define the meta-path-based ranking model for similarity semantic meaning i as follows:

$$S_i(q, r) = \sum_{f \in \mathbf{F}^*} f(q, r) \cdot \theta_f \tag{6.2}$$

where $\mathbf{F}^*$ is a selected subset of meta-path-based feature space, q is the query entity while r is a search result candidate, and θ_f is the coefficient associated with each feature f in $\mathbf{F}^*$. With this ranking model, similarity semantics hidden in the training dataset can be captured and represented by a set of meta-path-based features. Coefficients associated with features indicate the importance and expressiveness of each feature in terms of describing the target similarity semantics.

6.4.2 Training Dataset and Feature Selection

In order to conduct feature selection and ranking model training, we prepare the training datasets in this study following a *per-query* format, *i.e.*, for each entity, we collect a number of labeled entities (both positive and negative) under the target similarity semantic meaning assumption. The training dataset can be formalized as follows:

$$\mathcal{D} = (q_i, q_i^+, q_i^-), i = 1, \ldots, N \tag{6.3}$$

where q_i is the query entity, q_i^+ is a set of the positive examples representing the correct answers (relevant entities) to query q_i, while q_i^- is a set of the negative examples representing the incorrect answers (irrelevant entities) to q_i under the target similarity semantic meaning.

According to [17], information gain or entropy based feature selection metrics are not suitable for ranking problems. Based on this observation, we propose a feature selection method based on Kendall tau rank correlation coefficient [1].

Given a training dataset and a comprehensive meta-path-based feature space, for each query q_i in the dataset, we first enumerate all positive and negative pairs by calculating $q_i^+ \times q_i^-$. High quality features should be able to rank positives before negatives. In order to rank two instances correctly, a feature f should have $f(q_i, q_i^{+(m)}) > f(q_i, q_i^{-(n)})$ for a given query q_i and a positive negative pair $(q_i^{+(m)}, q_i^{-(n)})$. Correctly ranked positive and negative pairs are denoted as *concordant pairs* for feature f, while incorrectly ranked pairs are denoted as the *discordant pairs* for feature f. Notice that, pairs which have $f(q_i, q_i^{+(m)}) = f(q_i, q_i^{-(n)})$ are neither concordant nor discordant.

Rather than using the Kendall tau rank coefficient to measure the correlation between the feature and the training dataset, we propose to use concordant and discordant ratio to rank features. By calculating similarities between all possible entities with one meta-path-base feature, we can

generate a similarity matrix. The similarity matrices for different features may have different densities. In general, Kendall tau ranking coefficient favors features with denser similarity matrices, which does not necessarily indicate high feature quality. Concordant and discordant ratio does not have such bias. By adding concordant and discordant ratios together, we can define the correlation between a meta-path-based feature f and training dataset D as follows:

$$rank\_corr(f, D) = \sum_{q \in \mathcal{D}} \frac{concordant(q, f) + 1}{discordant(q, f) + 1} \tag{6.4}$$

After calculating the ranking correlation of each feature and the training dataset, we employ an entropy based histogram thresholding method to select features. More details can be found in [14].

6.4.3 Ranking Model Learning

In order to estimate the parameters in Equation 6.2, we use a similar objective function as proposed in [38]. The objective function is defined as follows:

$$O(\theta) = \sum_{i=1,\ldots,N} o_i(\theta) \tag{6.5}$$

where $o_i(\theta)$ is per-query objective function, measuring the performance of ranking model on each query q_i. Per-query objective function $o_i(\theta)$ is defined as log-likelihood function for all positive and negative instances related to q_i:

$$o_i(\theta) = \sum_{e^+ \in q_i^+} \frac{ln(p_i(e^+))}{|q_i^+|} + \sum_{e^- \in q_i^-} \frac{ln(1 - p_i(e^-))}{|q_i^-|} \tag{6.6}$$

where $p_i(e) = \sigma(S_i(q, e))$ and $\sigma(x)$ is the sigmoid function. Potential bias due to an unequal number of positive and negative instances is resolved by normalizing with the number of positive instances and negative instances so that their impacts on the objective function will be balanced. Equation 6.5, the objective function, is a sum over all queries in the training dataset. By maximizing this function, a linear combination ranking model can be learned with a standard optimization method. We use Broyden Fletcher Goldfarb Shanno (BFGS) method in our experiments for optimization.

6.5 Ranking Models Selection

With the method introduced in last section, for each similarity semantic meaning, with sufficient training data, we can learn a ranking model. Each ranking model can answer similarity queries independently under the assumption of certain similarity semantic meaning. Given a query, we need to first select the related ranking models. With the related models, search results can be calculated accordingly. To select related ranking models given a query, we need to first understand the hidden semantic meaning of the given query. In this section, we introduce the model selection method given query q and similarity examples $e_1, e_2, ..., e_m$.

Given a query q and similarity examples $e_1, e_2, ..., e_m$, we need to find the related models from a set of ranking models learned in Section 6.4. In this study, we assume all ranking models are independent from each other, and one query can be matched to 0, 1 or more than 1 ranking models. If 0 models can be found, we inform users that the current system can not understand the query. If 1 or more models can be found, we use these models to answer the query independently and return the search results from each model to the users.

Based on the above discussion, we can model the ranking model selection problem as a set of two-class classification problem. For each ranking model, we use a classifier to determine whether a given query with similarity examples is a match. We employ a method similar to logistic regression for such classification task as follows:

$$Pr(match = 1|q, S_i; \boldsymbol{\theta}) = \frac{e^z}{e^z + 1} \tag{6.7}$$

where $z = \Sigma_{e_i}\Sigma_{f \in F\prime}\theta_f \cdot f(q, e_i) + b$. $Pr(match = 1|q, S_i; \boldsymbol{\theta})$ is the probability that query q matches similarity semantic meaning S_i. f is a meta-path-based similarity feature.

In order to learn the above classifier for each ranking model, we need to prepare training dataset for each semantic meaning. Intuitively, similar entity pairs with the target semantic meaning can be used as positive training examples, while dissimilar pairs can be used as negatives. To generate such dataset, we re-organize the dataset that used in ranking model training by result. For each result r in $\mathcal{D}$, positively related queries are collected and denoted as t_r^+. "Positively related" means that under the target semantic meaning, r would be retrieved and ranked as a relevant entity for a

user input query q, *i.e.*, $r \in q^+$. Similarly, we denote t_r^- as queries which are negatively related to entity r. The model selection classifier should be able to distinguish t_r^+ and t_r^- for a given r. Based on the re-organization, positive training examples are queries which can be used to retrieve the same entity as relevant search result, while negatives can not. Following this idea, for each result entity r, we sample positive training examples from $t_r^+ \times t_r^+$ and negatives from $t_r^+ \times t_r^-$. Formally, we define the training dataset as $\mathcal{D}^{\mathcal{T}}$ (Equation 6.8):

$$\mathcal{D}^{\mathcal{T}} = r_i^+, r_i^-, i = 1, \ldots, N \tag{6.8}$$

where $r^+ = t_r^+ \times t_r^+$ and $r^- = t_r^+ \times t_r^-$.

With the training dataset for each semantic meaning, we learn the model selecting classifier as proposed in Equation 6.7 with MLE (Maximum Likelihood Estimation) in our experiments. With the ranking model selecting classifiers, given a query with examples from users, we can identify which ranking models are related to the query, and thus we can generate search results with the related ranking models. We call the collection of ranking model selecting classifiers query dispatcher as in Figure 6.5.1 because the purpose of the classifiers is to match the query to the correct ranking models.

6.5.1 Unified User-Guided Entity Similarity Search Framework

During the on-line query answering process, users provide similarity queries along with examples, aiming to find similar entities from a heterogeneous information network. As illustrated in Figure 6.5.1, a query with user guidance will first go through the query dispatcher, which contains a set of ranking model selecting classifiers. The classifiers will decide whether the represented ranking models are related to the given queries. After choosing related ranking models, we use each model to answer the query independently, and return the results as the final answer to the query.

6.6 Experiments

We test the proposed semantic meaning aware user-guided entity similarity search approach in this section. In order to demonstrate the existence of query semantic ambiguity problem, and the

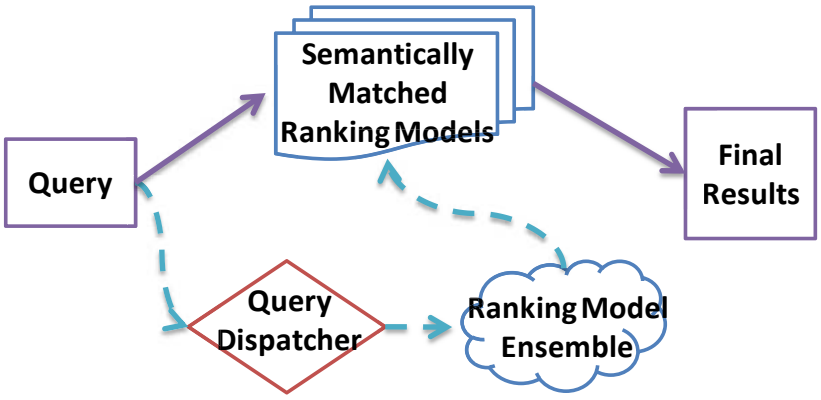


Figure 6.2: Entity similarity search framework

effectiveness of the proposed approach in terms of understanding similarity semantic meanings and retrieve higher relevance entities, we apply our method along with several competitor methods on both DBLP network and IMDb network. The schema of the two networks can be found in Figure 2.1. Both datasets are collected from web-based applications. DBLP dataset that we used contains 5,000 authors, 464 publication venues (both conferences and journals), 382,519 papers, 16,798 terms, and IMDb network has 29,360 movies and TV Shows, 53,088 actors, 5,408 directors, 22,470 keywords and 28 different genres. We choose four entity types, which are author, publication venue, movie and actor, to test the user-guided entity similarity search performance against different methods, mostly because the search results of these entity types are easier to evaluate compared to entity types like paper or director. We define 18 different similarity search semantic meanings on these two networks. Besides the examples we introduced in Section 6.1, more similarity search semantic meaning examples can be found in Figure 6.3. From these examples, one can notice that entity similarity ambiguity problem is ubiquitous in heterogeneous information networks, and entity similarity queries can be hard to understand even for human beings without sufficient domain knowledge. However, by utilizing additional similar examples provided by user (user guidance), our approach can first infer entity similarity semantic meanings associated with given queries, and then choose corresponding ranking model(s) from ranking model ensemble to answer the queries accordingly.

Type	Semantic Meanings	Query Examples
Actor	TV/movie co-stars	"Jennifer Aniston" taking "Lisa Kudrow" and "Courteney Cox" as examples
Actor	similar genre and popularity	"Jennifer Aniston" taking "Sarah J. Parker" and "Christina Applegate" as examples
Venue	same domain	"KDD" taking "SDM" and "PAKDD" as examples
Venue	same reputation	"KDD" taking "VLDB" and "WWW" as examples

Figure 6.3: Entity similarity semantic meaning example

6.6.1 Training Dataset and Feature Space

In order to build ranking models for similarity semantic meaning, we need to first collect training datasets in *per-query* format as described in Section 6.4.2, *i.e.*,, for each query, relevant and irrelevant results are collected as the positive and negative examples. User log is the major source for collecting training datasets in large-scale search systems for ranking model learning. Given our system is newly built, it is difficult for us to collect sufficient user logs. Instead, in the following experiments, we generate training datasets from previous published data mining results, by using well-known ground truth from public web services including DBLP and arnetminer *etc*, or by manually labeling. The quality and quantity of our training datasets are arguably sufficient for all the experiments, and we make sure the fairness of comparison by using the same training datasets when learning raking models for different methods.

In order to interpret various entity search semantic meanings, as discussed in Section 6.3, a comprehensive meta-path-based feature space needs to be calculated before model learning. Based on Equation 6.1, feature space is a combination of meta-path set $\mathbf{P}$ and meta-path compatible similarity / proximity measurements $\mathbf{M}$. Meta-paths convey diverse similarity semantic meanings, while similarity measurements utilize different similarity heuristics. We enumerate valid meta-paths within length 6 and use this set as $\mathbf{P}$. We implement PathSim, Personalized PageRank (denoted as p-PageRank), Random Walk, as well as SimRank as meta-path compatible measurements ($\mathbf{M}$ as in Equation 6.1). One should notice that, due to the differences in measurement definitions, similarity measurements have different meta-path compatibility. For example, PathSim requires symmetric meta-paths, and p-PageRank can only be calculate along infinite paths (or paths which are long

enough to make p-PageRank converge). Moreover, meta-paths can be semantically insignificant, and similarity measurements calculated along such paths can hardly contribute anything to the model learning process. For example, meta-path *movie-year-movie-year-movie* carries the exact same amount of information compared with meta-path *movie-year-movie*. Considering such cases, when we calculate meta-path-based feature space $\mathbf{F}$, we ignore invalid meta-path and measurement combinations and skip semantically insignificant meta-paths for efficiency consideration.

In the rest of this section, we study and present our experiments on similarity semantic meaning awareness, the expressiveness of a comprehensive meta-path-based feature space, as well as case studies on similarity ambiguity analysis.

6.6.2 Semantic Meaning Awareness Experiment

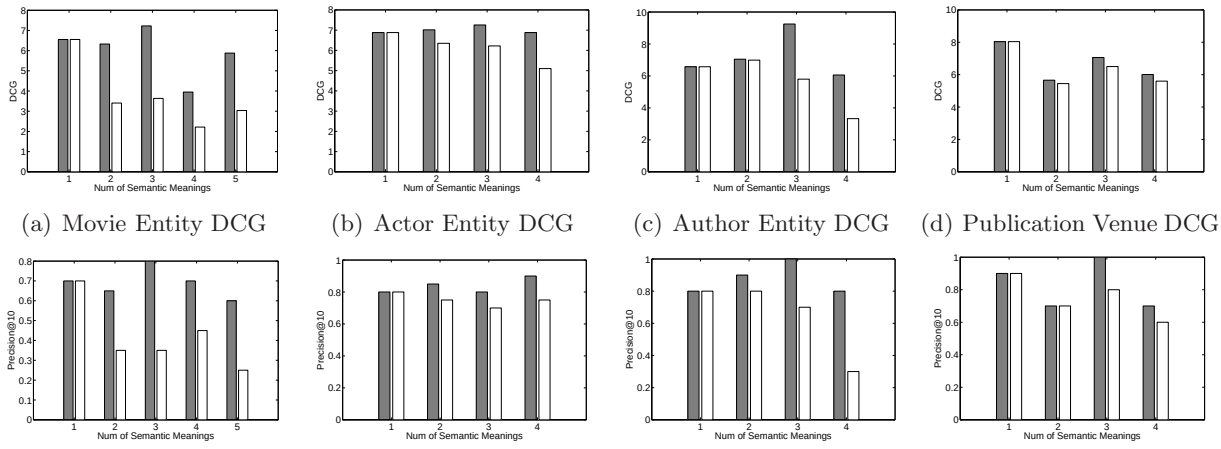
We first compare our similarity semantic meaning aware approach to regular information network learning-to-rank method. We define 18 different similarity semantic meanings over the entire dataset, and collect training datasets accordingly. In order to demonstrate the similarity semantic ambiguity problem, we incrementally add training datasets associated with different similarity semantic meanings into our system and the competitor system. Both systems utilize the entire meta-path-based feature space, use the same optimization method and objective function during ranking model learning, and take in the same amount of training data at each stage. The difference is at each stage, a new ranking model is trained and a model selecting classifier is learned accordingly in our system, while in the competitor system, the semantic meaning unaware learning-to-rank approach, combines new data and old data and learns one updated ranking model. Retrieval results evaluation is not trivial, since returned entities possess similarity semantic meanings and these entities belong to different domains. It requires a decent understanding of the domain to be able to judge the results. We invite people with such background to test our system. For each query, they assess the top 30 results with their judgment and evidence. To ensure the quality of the assessment, we allow the invited human judges to use the Internet to verify their labels.

We use Discounted Cumulative Gain (DCG score), which is a widely used information retrieval evaluation metric considering both precision and recall, and precision-at-10 to evaluate the performances of the two systems. The results can be found in Figure 6.4.

During the experiments, to make a fair competition, we make sure that queries and related entities do not appear in any training datasets. Based on the experiments results, one can notice that, when there is only one semantic meaning in the training dataset, the performance of both systems are exactly the same. However, the newly proposed semantic meaning aware framework can retrieve more relevant results than the competitor system when the number of semantic meanings increases. For example, when all similarity semantic meanings added, our system can out perform competitor system significantly on movie queries and author queries. Even with only two semantic meanings, our system can increase the performance of regular learning-to-rank model by at least 50% measured with precision-at-10. Significant improvements can be observed in other entity types as well.

However, the improvement of our system is not linear with the number of semantic meanings increase, and it gives a very similar performance on publication venue and author entity when two semantic meanings are introduced into the systems. The explanation of such observation is that although we assume the semantic meanings are not correlated, in real-world, semantic meanings often do correlate. For example, one semantic meaning for author entities can be collaboration (as discussed in Section 6.1), with which users can search for collaborators of the target author, while another semantic meaning can be similar research interests. Clearly these two semantic meanings are correlated, since one major factor that collaborators decide to work together is due to their common research interests. When the correlation of semantic meanings in the system is high, improvement of our system will be less obvious.

The reason that our method defeats traditional learning-to-rank method is because our system is capable of capturing entity similarity semantic differences carried by the query with user guidance, while similarity semantic meaning unaware method can not utilize such information and treats all queries the same. This experiment proves the existence of the similarity semantic ambiguity problem in real-world heterogeneous information network datasets, showcases the ability of entity similarity query semantic understanding of the proposed system, and also demonstrates the importance of semantic meaning understanding in terms of improving the quality of the entity retrieval results.



(a) Movie Entity DCG (b) Actor Entity DCG (c) Author Entity DCG (d) Publication Venue DCG

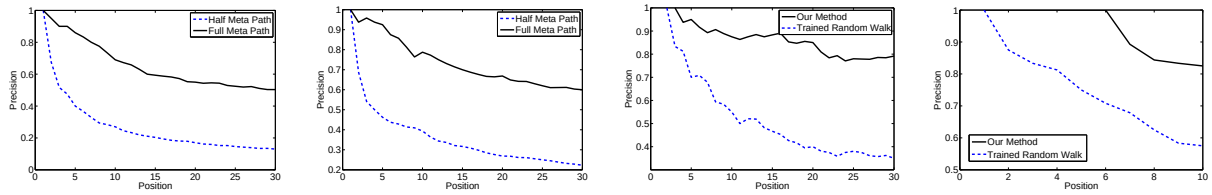
(e) Movie Precision@10 (f) Actor Precision@10 (g) Author Precision@10 (h) Publication Venue Precision@10

Figure 6.4: Performance Plots with DCG and Precision@10

6.6.3 Expressiveness of Meta-Path-based Feature Space

In order to learn semantic meaning aware ranking models and model selecting classifiers, we need an entity similarity semantic meaning representation. We claim the proposed meta-path-based feature space **F** is such a representation because meta-paths are capable of interpreting a number of entity similarity semantic meanings, while meta-path compatible measurements can quantify the similarity between entities with different structure heuristics. A relatively comprehensive meta-path-based feature space should be more expressive in terms of representing different similarity semantic meanings than an incomplete or sampled similarity feature space. In this experiment, we will demonstrate the power of comprehensive meta-path-based feature space. In the previous experiment, entity similarity semantic meanings are fixed and known to all comparison methods. We will change either **P** or **M** in Equation 6.1 to reduce the completeness of feature space **F**, and test the quality of the retrieval results and the overall performance.

We first sample 50% meta-path set **P** several times, and use the sampled meta-path sets to build similarity feature space, and apply the exact same learning-to-rank technique with the same amount of training datasets. We apply both the complete meta-path-based feature space and the sampled feature space on the proposed approaches, and evaluate the precision at each position in top 30 search results with IMDb dataset. The results can be found in Figure 6.5(a) and 6.5(b). From the results, one can notice that our proposed ranking system which utilizes the entire meta-



(a) Movie Entity Precision (b) Actor Entity Precision (c) Author Entity Precision (d) Publication Venue Precision

Figure 6.5: Performance Plots of Different Meta-Path-Based Models

path set **P** outperforms the ranking model with sampled meta-path set, by around 40% with movie queries and 50% with actor queries. The performance of top 3 to top 4 results of the ranking model with sampled meta-paths are reasonably good, but precision begins to drop dramatically for the rest of the results. The reason that a more comprehensive meta-path set can lead to better performance is very intuitive. Comprehensive meta-path set can interpret more similarity semantic meanings than sampled meta-path set.

We then sample meta-path compatible measurement set **M**, and we only use random walk as entity similarity measurement when building similarity feature space. We apply both the approach using the entire measurement set and the ranking model with only random walk similarity features on DBLP dataset, and similarly measure the precision at each position in the top 30 results returned by both methods. The results can be found in Figure 6.5(c) and 6.5(d). Very similar to the previous experiments, by using more meta-path compatible similarity measurements, our method outperforms the ranking model which only uses random walk as the similarity measurement. This means the meta-path-based similarity feature space is more expressive and can convey more information than traditional supervised random walk method. By aggregating different meta-paths and a number of similarity measurements, meta-path-based feature space makes representing entity similarity from different semantic aspects possible.

6.6.4 Similarity Ambiguity Case Study

After evaluating the semantic meaning aware entity search framework and demonstrate the power of meta-path-based similarity feature space, we apply meta-path-based model selecting classifiers on DBLP subset in order to demonstrate the significance of the similarity semantic understanding problem as well as the effectiveness of our proposed approach. To our best knowledge, similarity

semantic meaning understanding in heterogeneous information networks has not been studied previously, which makes it difficult to do quantitative evaluation. Therefore, we present two similarity semantic meaning case studies as follows:

Case Study 6.1 Query: find authors similar to "ChengXiang Zhai[a]", taking "Jing Gao[b]" as example. *Both authors work in Computer Science Department, University of Illinois, at Urbana-Champaign. And they collaborated before. So given this query, user would be interested to find out other authors who also work in the same department and may have collaborations with the author in the query. Our method manages to capture this semantic meaning and returns other authors who collaborated with ChengXiang Zhai from the same institute as search results.*

Case Study 6.2 Query: find authors similar to "Stefano Lonardi[c]", taking "Spiros Papadimitriou[d]" as example. *Lonardi is an associate Professor in University of California, Riverside while Papadimitriou was a PhD student in Carnegie Mellon University. Both of them studied and published papers on "time-series" analysis which makes them share similar research topic. Our ranking model selecting method successfully matches this query to research topic similarity ranking model during query dispatching step.*

Based on these experiments results, we can conclude that, meta-path-based ranking models give a significantly better performance than competitor methods. Examples that we show in this study on semantic meaning predicting demonstrate the common existence of this problem and the difficulty to solve this problem completely. With meta-path-based feature space and semantic meaning related training datasets, we manage to learn high quality ranking models and ranking model selecting classifiers which makes query processing and semantic meaning understanding on heterogeneous information network possible.

[a] http://www.cs.uiuc.edu/~czhai/
[b] http://www.ews.uiuc.edu/~jinggao3/
[c] http://www.cs.ucr.edu/~stelo/
[d] http://www.cs.cmu.edu/~spapadim/

6.7 Conclusion

In this chapter, we address the problem of entity similarity query semantic meaning understanding and query processing in heterogeneous information networks. Meta-path-based feature space helps measure similarity from different aspects, which could benefit information network retrieval as well as other information network related applications. By learning ranking models for different semantic meanings and training a query dispatcher which can select correct ranking models based on the queries, our search framework first tries to understand the hidden user intents and the semantic meaning in a given query with user guidance, and then uses related ranking models to further answer the query by returning lists of similar entities.

Empirical study shows that our approach describes and interprets semantic meanings better than other meta-path-based methods. Examples and case studies in this work prove the significance of the semantic meaning understanding problem and the effectiveness of our method in terms of understanding user intents. Interesting future work includes, retrieval and semantic meaning prediction on multi-sourced heterogeneous information network, meta-path-based feature space scalability study, as well as, semantic meaning prediction on multi-typed queries.

Chapter 7

Related Work

This chapter reviews the studies related to entity recommendation and search in heterogeneous information networks.

7.1 Information Network Analysis

Heterogeneous information networks which contain multi-typed entities and links become ubiquitous with the rapid growth of web-based applications and services. Information network analysis and mining have gained wide attention in both academia and industry. Many researchers believe that the heterogeneity and rich-relation nature of such datasets make information networks better data representations in many domains and scenarios. A lot of information network mining and learning tasks have been done in the past couple of years, including clustering [60] [61], classification [32], and link prediction [69], *etc.* Studies regarding entity similarity measurements, as a fundamental technique, have been actively engaged in many research works as well [13] [30] [59]. Researchers also discovered that certain similarity measurements could be defined along paths in information network, and such path compatible measurements could capture different similarity semantic meanings and can be used in different applications [38] [59]. These works also motivated user-guided or personalized data mining and analysis with information networks [60] [75].

7.2 Recommender Systems

Recommender systems have received increasing attentions and have been actively studied in recent years due to the wide applications in e-commerce and other web services. Collaborative filtering techniques, which include neighbor-based approaches and model-based approaches, are popular and widely applied. Neighbor-based methods usually study similarity calculation functions among users

(user-based) [10] [53] or items (item-based) [42] [56]. User-based collaborative filtering methods provide recommendation to target users by first finding similar users and then collecting and analyzing their ratings to further predict items that target users would be interested in. Similarly, item-based collaborative filtering methods take advantage of rating information of similar items.

Model-based collaborative filtering methods aim to fit user-item rating data into different models (Bayesian network [66], matrix factorization [36] [55], or clustering models [64] *etc.*), and use the learned models to recommend items to users. Matrix factorization techniques gain rising attention in both explicit and implicit feedback applications. Using low-rank approximation to represent user-item rating matrix suits the need of handling large-scale datasets. Many studies have been done using this technique to interpret different heuristics in user-item rating datasets [34]. In [27], Hu *et al.* propose to model implicit feedback as positive and negative preferences with different confidence levels with a matrix factorization based-approach. Koren incorporates temporal information in matrix factorization models to further improve the recommendation quality [35].

Besides traditional recommender systems, researchers proposed hybrid approaches to incorporate both user-item rating dataset as well as other contextual information in different applications, including social network information, time information, *etc.* For example, researchers model trustworthiness or similarities of users when building recommendation models [28] [47] [66]. As mentioned above, in [35], Koren proposed a time-aware collaborative filtering method that can effectively incorporate time information into collaborative filtering. Middleton *et al.* proposed to use ontological user profiling technique to build recommendation system [50]. Notice that ontology is substantially different from the entity graph we employed in Chapter 4. An ontology usually summarizes the concept levels of certain domains, *e.g.*, "text classification" is a subtopic of "machine learning", and the sizes of the ontologies are usually small. Entity graph is built with real world entities. The size of entity graphs can be large-scale. Yu *et al.* [72] [74] introduced meta-path concept into hybrid recommender systems.

The proposed framework in Chapter 3 and 4 both belong to hybrid recommender system category. Different from previous works, the framework proposed in Chapter 3 utilizes relationship heterogeneity. We introduced a heterogeneous network view of the recommendation problem and discussed the connection between user preference propagation and measuring relevance between

users and items. We also proposed to build recommendation models at different granularity levels. The study introduced in Chapter 4 utilizes various features and heuristics generated from user click log from a commercial search engine and the Freebase entity graph. The proposed solution is designed to build recommender system for search engine users, and thus it can achieve more promising results compared with traditional recommendation methods.

7.3 CF-Based Hybrid Recommendation Models

As mentioned above, both studies in Chapter 3 and Chapter 4 are collaborative filtering (CF) based hybrid recommendation models. As we know, collaborative filtering methods are the most popular recommendation techniques. These methods have seen wide applications in different systems and domains. Collaborative filtering techniques have been extensively studied from different perspectives as well [56] [25]. Among different techniques, matrix factorization based models [55, 34] are widely employed in many systems due to its good performance [36].

Recently studies on collaborative filtering turn to leveraging different types of external information so that the well-known data sparsity problem can be addressed and better recommendation performance can be achieved. Early studies utilized user and item information in collaborative filtering process. Such studies are called content-based collaborative filtering. For example, [49, 6] incorporated item or user profiles (*e.g.*, user demographic information and item attributes) into collaborative filtering framework. These additional information complement the user rating data and such systems can usually generate high quality results.

In contrast to content-based information, utilizing knowledge extracted from relational data when building recommendation models are attracting increasingly interest. In particular, several works leveraged different social relationships in social network such as trust relationship [29, 44], friend relationship [23, 46], and user membership [76] to boost collaborative filtering results since these methods not only utilize target users' ratings but also borrow information from their like-minded "neighbors". Gu *et al.* [21] proposed to utilize graph Lapalacian regularization to exploit both internal and external relational information from general item graphs and user graphs. Singh *et al.* [57] introduced a collective matrix factorization framework for learning multiple types of relational information simultaneously for collaborative filtering.

To our knowledge, most existing relationship-based collaborative filtering focus on extracting knowledge from single or multiple homogeneous network, *i.e.*, networks with single type of nodes and links. Our works aim to leverage external knowledge extracted from heterogeneous information network for collaborative filtering boosting.

7.4 Citation Prediction

Early works about citation prediction [48] focuses on predicting the citation numbers of a given paper. They estimates the number of citation for each paper in each time epoch, and uses time series analysis technique for citation count prediction. Recently, [68] studied similar problem using machine learning technique. They trained a classifier based on several features of fundamental characteristics for the papers which are highly cited and predicted the popularity degree of each paper. In contrast, the problem we studied in Chapter 5 is more challenging than citation number prediction. Rather than estimating the count of citations for each paper, we aim to predict the citation relationships. In other words, we answer queries like which papers should cite a target paper, or which papers should be cited by a target paper. In fact, citation number prediction can be seen as a by-product of the proposed method in this paper, because as long as we can predict the citation relationship, it is straightforward to count the number of citations in each time epoch.

For search engines in networks, the similarity function defined on networks is the essential component to provide high quality answers. SimRank [30], P-PageRank (personalized PageRank) [31, 63, 13] and other extension works have been proposing new similarity functions in graphs and information network. Most of these functions are defined at a global level. Recently, [59] proposed a meta-path-based framework in defining the similarity measurements between two entities of the same type. However, they did not address the learning issue with this framework and how to incorporate topic / term similarity into the aforementioned framework.

[58] studied coauthor prediction, which attempts to predict the coauthor relationships in the future. However, the problem setting of coauthor prediction is simper than citation prediction. Coauthor prediction is a short term prediction problem. An author who published paper in 2011 is unlikely to become a coauthor of another author who published paper in 1960. In contrast, citation relationship does not have obvious time constraint. It is reasonable for a paper published in 2011

to cite another paper which was published in 1960. On the other hand, coauthor relationship has strong propagation property. For example, if author A and author B are coauthors, author B and author C are coauthors, then the probability of author A and author C being coauthors is high. However, the propagation perpetuity might not hold in citation prediction problem. In a word, citation prediction is more challenging than coauthor prediction or other similar link prediction problems. Our proposed framework in Chapter 5 considers both topic / term similarity and structural similarity and properties of paper entities. The proposed method achieves good performance in terms of effectiveness and efficiency.

7.5 Entity Search in Information Networks

In order to build an entity similarity search engine for heterogeneous information network data, it is critical to study the existing entity similarity functions. SimRank [30] is a well-known entity similarity function defined on networks. This method measures similarity between entities based on the intuition that entities are similar if they are connected to other similar entities. Due to the fact that SimRank is a global similarity function, the high computational cost makes it difficult to be implemented in large-scale. P-PageRank (personalized PageRank) [31, 63, 13] evaluates the probability of a given entity to other entities in the network with a stochastic process (random walk with restart), and thus it can be used to measure entity similarities in network structures. Other extensions such as ObjectRank [5] try to assign different weights to different relation types in a heterogeneous network. However, these similarity ranking functions are defined with specific similarity semantic meanings, which makes it difficult to incorporate user guidance and preferences into the search process.

On the other line, learning-to-rank approaches in network or graph datasets are frequently studied in the machine learning community [3, 2]. Different from the fixed ranking functions listed above, users can provide training samples as ordered pairs or lists during model training, and these methods will return ranking functions which has a low inconsistent rate with the labeled data and a high consistency with the graph structures. These studies can be applied to both global ranking problems and query-based ranking problems in information networks.

Recently, path-constrained random walk-based algorithms are proposed in [38, 37], which allows

us to calculate entity similarities at path level. Such works often incorporate user participation during training and feature construction. The idea behind these algorithms is that different paths can be used to represent different semantics. For a given ranking task, paths utilized in the framework may have different importances for the specific task. Authors usually use single similarity measurement in such studies, *e.g.*, random walk, and the search tasks are often pre-defined, *e.g.*, reference recommendation, or co-authorship prediction.

In [59, 58], the authors proposed a meta-path-based framework to define entity similarity measurements between entities of the same type. However, they did not address the learning problems for different query tasks, and they assume all queries are well-explained with no ambiguity.

Several studies in this thesis utilize a similar meta-path-based feature space as introduced above. Features are used differently in different studies. In recommendation, we use the meta-path or path-constraint definition to study user preference prorogation, while in search studies, we build meta-path-based feature space by combining meta-path set with different similarity measurements to further define ranking scores for different tasks.

Recently, query intent understanding and classification studies have drawn more attentions in both academia and industry. As discussed before, if a search engine can not correctly interpret the semantic meaning hidden in the queries, the retrieval results from this search engine can be unsatisfying. In [39], the authors propose to use click graphs to improve the quality of the query intent classifier. By combining query features along with click information, the proposed approach generates a click graph and further optimizes the click-through performance of the search engines. Hu *et al.* studied how to understand users' query intents with knowledge from Wikipedia in [26]. Ashkan *et al.* also demonstrated the performance improvement in traditional text search engines measured by click-through rate after adding a query intent classification feature in [4]. Based on our study in Chapter 6, we demonstrated that query ambiguity issues exist not only in text search engines but also in heterogeneous information network related search problems. Without understanding the meaning of the queries, it is difficult for any system to provide high quality search results. Our study addressed the ambiguity issue within a entity similarity search framework. More studies along the line of understanding user intents in heterogeneous network search problems are needed to generate high quality search results for different search tasks.

Chapter 8

Conclusions and Research Frontiers

In this thesis, I have studied entity recommendation and search problems in heterogeneous information networks. The proposed techniques can help users explore heterogeneous network datasets and facilitate their information discovery process. Previous studies on entity recommendation and search problems mostly treat the available data as two dimensional matrices or homogeneous graphs. The heterogeneous network view proposed in this thesis can be used to better model real-world information retrieval scenarios. Incorporating heterogeneous information networks into recommender systems and search engines not only improves the quality of the results, but also introduces new research topics into the data mining and information retrieval communities.

8.1 Conclusion

The major contributions of this thesis are summarized in the following aspects:

- **Heterogeneous Network View of Entity Recommendation and Search.** This thesis re-examined entity recommendation and search from the heterogeneous network perspective. Chapter 3 introduced to use user preference propagation technique to represent users and items under different semantics. Chapter 4 utilized an entity graph to bridge the gap of search and recommendation. Chapter 5 and 6 studied different search problems with the help of relationship heterogeneity. The heterogeneous network perspective motivated new methodologies in these studies.

- **Network Mining at a Fine Granularity.** Data mining in networks has been evolving in the past decade. From homogeneous networks to multi-homogeneous network projections, to heterogeneous networks, each evolution iteration introduces new problems, research principles

and methodologies. Sun *et al.* distinguished entity relationship types at a global level in heterogeneous networks in several previous studies [59, 58]. Chapter 3 and 4 propose to examine heterogeneous relationships at personalized level. Personalized heterogeneous relationship-based recommendation models illustrate user preferences at a finer granularity, which leads to good understanding of user behaviors and provides better user experiences.

- **Connection to Real-World Applications.** Entity recommendation and search problems have been widely applied in different domains. The studies included in this thesis not only address new research issues and propose effective methodologies but also examine the methods with real-world data and applications. Chapter 4 utilizes heterogeneous network mining principles with large-scale search engine data and proposes innovating features for commercial search engines. Chapter 5 applies topic and network techniques in citation prediction problem with DBLP dataset, which can benefits researchers from different disciplines. The connection between this thesis and real-world applications demonstrate the utility of the studies.

8.2 Research Frontiers

There are many interesting directions of future work along the line of entity recommendation and search in heterogeneous information networks. Based on the studies included in this thesis, I present several possible future work from the following aspects.

8.2.1 Serendipity-Oriented Recommendation

A few previous studies pointed out that serendipity is a desired feature in recommender systems [16, 52]. McNee *et al.* pointed out that sometimes the most accurate recommendation according to the standard metrics are not the most useful results to users. Recent studies begin to utilize different metrics when evaluating the quality of the recommender systems [7, 77]. In this thesis, the proposed models are optimized and evaluated based on accuracy. However, it is important to study serendipity-oriented recommendation methods in heterogeneous information network environment.

Compared with single source recommendation techniques, *e.g.*, collaborative filtering or social network-based methods, heterogeneous network-based techniques have the potential to find more

recommendation candidates under different semantics due to the relationship heterogeneity. One possible way to promote serendipity in recommendation results is to assign serendipity scores to items in both training and evaluation. The serendipity score can be defined by collaborative filtering techniques, *e.g.*, the higher support the recommendation results have, the lower serendipity score the items are assigned. When we are learning recommendation models with features from heterogeneous information networks, rather than optimizing accuracy directly, we can optimize the aforementioned serendipity score or a combination of serendipity score and accuracy. In this way, we can take advantage of the rich features in information networks and build serendipity-oriented recommendation models accordingly.

8.2.2 Unified Recommendation and Search in Heterogeneous Networks

The problem of entity recommendation and search in heterogeneous networks resemble each other. From the use case perspective, recommender systems measure similarity between user interests and potential entities, while entity search engines measure similarity between user queries and potential entities. From the optimization / evaluation perspective, both systems aim to generate a list of entities to satisfy users' information needs. Based on the above observations, it is possible for us to unify the entity recommendation and search solutions into one system. The advantage of such studies is that the core techniques in both systems can be interchanged. The new findings in both disciplines can be used to improve the quality of the unified system.

In order to build a unified system for both entity recommendation and search, several possible solutions need to be explored. One possible solution is to convert the hidden user interests to queries and rely on the entity search engine to provide recommendation results. Another direction is to convert both recommendation and search problems to link prediction problem. By studying a generalized link prediction problem with constraints in heterogeneous information networks, we can provide results for both recommender systems and search engines accordingly.

8.2.3 Handling Various Networks during Recommendation and Search

The network data we used in this thesis are reliable knowledge bases in different domains with high precision (definitions can be found in Chapter 2). However, real-world information networks

may not follow the definition of the information networks in this thesis. Special considerations are needed when handling information network variations.

In our studies, we do not consider weights or probabilities on entity relationships. However, in certain real-world scenarios, *e.g.*, newly constructed networks from text collections, networks may have weights associated with entity links to represent confidence or importance of the relationships. The studies included in this thesis treat all entity links the same, *i.e.*, $weight = 1$, which makes these methods unsuitable when handling networks with different link weights. We need to study relationship weight compatible feature space or look for entity similarity or proximity measurements which can utilize link weights when building recommender systems or search engines.

Another problem with network data is privacy preservation. In heterogeneous networks, certain relationships could be privacy sensitive, *e.g.*, social relationships. When handling such networks, privacy preservation becomes an very important issue. Including more entity relationships can potentially boost the quality of recommendation or search, but meanwhile, it makes the systems vulnerable for privacy attack. Good entity search engines and recommender systems should be able to identify such vulnerability and adjust the personalized recommendation results to preserve privacy when needed.

References

[1] Hervé Abdi. The kendall rank correlation coefficient. *Encyclopedia of Measurement and Statistics*, pages 508–510, 2007.

[2] Alekh Agarwal, Soumen Chakrabarti, and Sunny Aggarwal. Learning to rank networked entities. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'06)*, pages 14–23, 2006.

[3] Shivani Agarwal. Ranking on graph data. In *Proceedings of the 23rd international conference on Machine learning (ICML'06)*, pages 25–32, 2006.

[4] Azin Ashkan, Charles LA Clarke, Eugene Agichtein, and Qi Guo. Classifying and characterizing query intent. In *Advances in Information Retrieval*, pages 578–586. Springer, 2009.

[5] Andrey Balmin, Vagelis Hristidis, and Yannis Papakonstantinou. Objectrank: Authority-based keyword search in databases. In *Proceedings of the 30th international conference on Very large data bases (VLDB'04)*, pages 564–575, 2004.

[6] Justin Basilico and Thomas Hofmann. Unifying collaborative and content-based filtering. In *Proceedings of the 21st international conference on Machine learning (ICML'04)*, page 9, 2004.

[7] Alejandro Bellogín, Iván Cantador, and Pablo Castells. A comparative study of heterogeneous item recommendations in social systems. *Information Sciences*, 221:142–169, 2013.

[8] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data (SIGMOD'08)*, pages 1247–1250, Vancouver, Canada, 2008.

[9] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of the 19th International Conference on Computational Statistics (CompStat'10)*, pages 177–186. Springer, 2010.

[10] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th conference on Uncertainty in artificial intelligence (UAI'98)*, pages 43–52, San Francisco, CA, USA, 1998.

[11] Richard H Byrd, Peihuang Lu, Jorge Nocedal, and Ciyou Zhu. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16(5):1190–1208, 1995.

[12] Huanhuan Cao, Daxin Jiang, Jian Pei, Qi He, Zhen Liao, Enhong Chen, and Hang Li. Context-aware query suggestion by mining click-through and session data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'08)*, pages 875–883, Las Vegas, Nevada, USA, 2008.

[13] Soumen Chakrabarti. Dynamic personalized pagerank in entity-relation graphs. In *Proceedings of the 16th international conference on World wide web (WWW'07)*, pages 571–580, Banff, Alberta, Canada, 2007.

[14] C-I Chang, Yingzi Du, J Wang, S-M Guo, and PD Thouin. Survey and comparative analysis of entropy and relative entropy thresholding techniques. In *Proceedings of the 2006 Vision, Image and Signal Processing (VISP'06)*, pages 837–850, 2006.

[15] Chris Ding, Tao Li, and Michael I Jordan. Convex and semi-nonnegative matrix factorizations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(1):45–55, 2010.

[16] Mouzhi Ge, Carla Delgado-Battenfeld, and Dietmar Jannach. Beyond accuracy: evaluating recommender systems by coverage and serendipity. In *Proceedings of the fourth ACM conference on Recommender systems (RecSys'10)*, pages 257–260, 2010.

[17] Xiubo Geng, Tie-Yan Liu, Tao Qin, and Hang Li. Feature selection for ranking. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'07)*, pages 407–414, 2007.

[18] Quanquan Gu and Jiawei Han. Towards feature selection in network. In *Proceedings of the 20th ACM international conference on Information and knowledge management (CIKM'11)*, pages 1175–1184, 2011.

[19] Quanquan Gu, Zhenhui Li, and Jiawei Han. Generalized fisher score for feature selection. In *Proceedings of the 27th conference on Uncertainty in artificial intelligence (UAI'11)*, pages 266–273, 2011.

[20] Quanquan Gu and Jie Zhou. Subspace maximum margin clustering. In *Proceedings of the 18th ACM conference on Information and knowledge management (CIKM'09)*, pages 1337–1346, Hong Kong, China, 2009.

[21] Quanquan Gu, Jie Zhou, and Chris HQ Ding. Collaborative filtering: Weighted nonnegative matrix factorization incorporating user and item graphs. In *Proceedings of the 2010 SIAM International Conference on Data Mining (SDM'10)*, pages 199–210, 2010.

[22] Siyu Gu, Jun Yan, Lei Ji, Shuicheng Yan, Junshi Huang, Ning Liu, Ying Chen, and Zheng Chen. Cross domain random walk for query intent pattern mining from search engine log. In *Proceedings of the 2011 IEEE international conference on data mining (ICDM'11)*, pages 221–230, 2011.

[23] Ido Guy, Naama Zwerdling, David Carmel, Inbal Ronen, Erel Uziel, Sivan Yogev, and Shila Ofek-Koifman. Personalized recommendation of social software items based on social relations. In *Proceedings of the third ACM conference on Recommender systems (RecSys'09)*, pages 53–60, 2009.

[24] Jorge E Hirsch. An index to quantify an individual's scientific research output. *Proceedings of the National academy of Sciences of the United States of America (PNAS'05)*, 102:16569–16572, 2005.

[25] Thomas Hofmann. Collaborative filtering via gaussian probabilistic latent semantic analysis. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'03)*, pages 259–266, 2003.

[26] Jian Hu, Gang Wang, Fred Lochovsky, Jian-tao Sun, and Zheng Chen. Understanding user's query intent with wikipedia. In *Proceedings of the 18th international conference on World wide web (WWW'09)*, pages 471–480, 2009.

[27] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 IEEE international conference on data mining (ICDM'08)*, pages 263–272, 2008.

[28] Mohsen Jamali and Martin Ester. Trustwalker: a random walk model for combining trust-based and item-based recommendation. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'09)*, pages 397–406, Paris, France, 2009.

[29] Mohsen Jamali and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems (RecSys'10)*, pages 135–142, 2010.

[30] Glen Jeh and Jennifer Widom. Simrank: a measure of structural-context similarity. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'02)*, pages 538–543, Edmonton, Alberta, Canada, 2002.

[31] Glen Jeh and Jennifer Widom. Scaling personalized web search. In *Proceedings of the 12th international conference on World wide web (WWW'03)*, pages 271–279, 2003.

[32] Ming Ji, Jiawei Han, and Marina Danilevsky. Ranking-based classification of heterogeneous information networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'11)*, pages 1298–1306, San Diego, California, USA, 2011.

[33] Thorsten Joachims. Training linear svms in linear time. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'06)*, pages 217–226, 2006.

[34] Yehuda Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'08)*, pages 426–434, Las Vegas, Nevada, USA, 2008.

[35] Yehuda Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD '09)*, pages 447–456, Paris, France, 2009.

[36] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[37] Ni Lao and William W Cohen. Fast query execution for retrieval models based on path-constrained random walks. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'10)*, pages 881–888, 2010.

[38] Ni Lao and William W Cohen. Relational retrieval using a combination of path-constrained random walks. *Machine learning*, 81(1):53–67, 2010.

[39] Xiao Li, Ye-Yi Wang, and Alex Acero. Learning query intent from regularized click graphs. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR'08)*, pages 339–346, 2008.

[40] Ryan N Lichtenwalter, Jake T Lussier, and Nitesh V Chawla. New perspectives and methods in link prediction. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'10)*, pages 243–252, 2010.

[41] Thomas Lin, Patrick Pantel, Michael Gamon, Anitha Kannan, and Ariel Fuxman. Active objects: actions for entity-centric search. In *Proceedings of the 21st international conference on World wide web (WWW'12)*, pages 589–598, Lyon, France, 2012.

[42] Greg Linden, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[43] Chao Liu, Hung-chih Yang, Jinliang Fan, Li-Wei He, and Yi-Min Wang. Distributed nonnegative matrix factorization for web-scale dyadic data analysis on mapreduce. In *Proceedings of the 19th international conference on World wide web (WWW'10)*, pages 681–690, 2010.

[44] Hao Ma, Irwin King, and Michael R Lyu. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval (SIGIR'09)*, pages 203–210, 2009.

[45] Hao Ma, Chao Liu, Irwin King, and Michael R Lyu. Probabilistic factor models for web site recommendation. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval (SIGIR'11)*, pages 265–274, 2011.

[46] Hao Ma, Haixuan Yang, Michael R Lyu, and Irwin King. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management (CIKM'08)*, pages 931–940, 2008.

[47] Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining (WSDM'11)*, pages 287–296, Hong Kong, China, 2011.

[48] JN Manjunatha, KR Sivaramakrishnan, Raghavendra Kumar Pandey, and M Narasimha Murthy. Citation prediction using time series approach kdd cup 2003 (task 1). *ACM SIGKDD Explorations Newsletter*, 5(2):152–153, 2003.

[49] Prem Melville, Raymod J Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the 2002 AAAI Conference on Artificial Intelligence (AAAI'02)*, pages 187–192, 2002.

[50] Stuart E. Middleton, Nigel R. Shadbolt, and David C. De Roure. Ontological user profiling in recommender systems. *ACM Transactions Information System*, 22(1):54–88, January 2004.

[51] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP (ACL/IJCNLP'09)*, pages 1003–1011, 2009.

[52] Tomoko Murakami, Koichiro Mori, and Ryohei Orihara. Metrics for evaluating the serendipity of recommendation lists. *New frontiers in artificial intelligence*, pages 40–46, 2008.

[53] Michael J Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, 13(5-6):393–408, 1999.

[54] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI'09)*, pages 452–461, 2009.

[55] Jasson D. M. Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd international conference on Machine learning (ICML'05)*, pages 713–719, Bonn, Germany, 2005.

[56] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World wide web (WWW'01)*, pages 285–295, 2001.

[57] Ajit P Singh and Geoffrey J Gordon. Relational learning via collective matrix factorization. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'08)*, pages 650–658, 2008.

[58] Yizhou Sun, Rick Barber, Manish Gupta, Charu C Aggarwal, and Jiawei Han. Co-author relationship prediction in heterogeneous bibliographic networks. In *Proceedings of the 2011 international conference on Advances in social networks analysis and mining (ASONAM'11)*, pages 121–128, 2011.

[59] Yizhou Sun, Jiawei Han, Xifeng Yan, Philip S Yu, and Tianyi Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. In *Proceedings of the 2011 Conference on Very large data bases (VLDB'11)*, 2011.

[60] Yizhou Sun, Brandon Norick, Jiawei Han, Xifeng Yan, Philip S Yu, and Xiao Yu. Integrating meta-path selection with user-guided object clustering in heterogeneous information networks. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'12)*, pages 1348–1356, 2012.

[61] Yizhou Sun, Yintao Yu, and Jiawei Han. Ranking-based clustering of heterogeneous information networks with star network schema. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'09)*, pages 797–806, 2009.

[62] Jie Tang, Jing Zhang, Limin Yao, Juanzi Li, Li Zhang, and Zhong Su. Arnetminer: extraction and mining of academic social networks. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining (KDD'08)*, pages 990–998, 2008.

[63] Hanghang Tong, Christos Faloutsos, and Jia-Yu Pan. Fast random walk with restart and its applications. In *Proceedings of the 6th IEEE International Conference on Data Mining (ICDM'06)*, pages 613–622, 2006.

[64] Lyle H Ungar and Dean P Foster. Clustering methods for collaborative filtering. In *Proceedings of AAAI Conference on Artificial Intelligence Workshop on Recommendation Systems*, 1998.

[65] Santosh S. Vempala. Random projection. In *The random projection method*, volume 65. American Mathematical Soc., 2004.

[66] Yao Wang and Julita Vassileva. Bayesian network-based trust model. In *Proceedings of WIC International Conference on Web Intelligence (WI'03)*, pages 372–378, 2003.

[67] Ryen W White, Peter Bailey, and Liwei Chen. Predicting user interests from contextual information. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval (SIGIR'09)*, pages 363–370, 2009.

[68] Rui Yan, Jie Tang, Xiaobing Liu, Dongdong Shan, and Xiaoming Li. Citation count prediction: Learning to estimate future citations for literature. In *Proceedings of the 20th ACM international conference on Information and knowledge management (CIKM'11)*, pages 1247–1252, 2011.

[69] Xiao Yu, Quanquan Gu, Mianwei Zhou, and Jiawei Han. Citation prediction in heterogeneous bibliographic networks. In *Proceedings of the 2012 SIAM International Conference on Data Mining (SDM'12)*, pages 1119–1130, 2012.

[70] Xiao Yu, Hao Ma, Bo-June (Paul) Hsu, and Jiawei Han. On building entity recommender systems using user click log and freebase knowledge. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM'14)*, pages 263–272, New York, NY, USA, 2014.

[71] Xiao Yu, Ang Pan, Lu-An Tang, Zhenhui Li, and Jiawei Han. Geo-friends recommendation in gps-based cyber-physical social network. In *Proceedings of the 2011 international conference on Advances in social networks analysis and mining (ASONAM'11)*, pages 361–368, 2011.

[72] Xiao Yu, Xiang Ren, Quanquan Gu, Yizhou Sun, and Jiawei Han. Collaborative filtering with entity similarity regularization in heterogeneous information networks. In *Proceedings of the 2013 Joint Conference of Artificial Intelligence, Heterogeneous Information Network Analysis Workshop (IJCAI-HINA'13)*, 2013.

[73] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. Personalized entity recommendation: A heterogeneous information network approach. In *Proceedings of the 7th ACM International Conference on Web Search and Data Mining (WSDM'14)*, pages 283–292, New York, NY, USA, 2014.

[74] Xiao Yu, Xiang Ren, Yizhou Sun, Bradley Sturt, Urvashi Khandelwal, Quanquan Gu, Brandon Norick, and Jiawei Han. Entity recommendation in heterogeneous information networks with implicit user feedback. In *Proceedings of the seventh ACM conference on Recommender systems (RecSys'13)*, pages 347–350, 2013.

[75] Xiao Yu, Yizhou Sun, Brandon Norick, Tiancheng Mao, and Jiawei Han. User guided entity similarity search using meta-path selection in heterogeneous information networks. In *Proceedings of the 21st ACM international conference on Information and knowledge management (CIKM'12)*, pages 2025–2029, 2012.

[76] Quan Yuan, Li Chen, and Shiwan Zhao. Factorization vs. regularization: fusing heterogeneous social relationships in top-n recommendation. In *Proceedings of the fifth ACM conference on Recommender systems (RecSys'11)*, pages 245–252, 2011.

[77] Yuan Cao Zhang, Diarmuid Ó Séaghdha, Daniele Quercia, and Tamas Jambor. Auralist: introducing serendipity into music recommendation. In *Proceedings of the fifth ACM international conference on Web search and data mining (WSDM'12)*, pages 13–22, 2012.