On Building Entity Recommender Systems Using User Click Log and Freebase Knowledge

Xiao Yu[†*], Hao Ma[‡], Bo-June (Paul) Hsu[‡], Jiawei Han[†]
[†]University of Illinois at Urbana-Champaign [‡]Microsoft Research
[†]{xiaoyu1, hanj}@illinois.edu [‡]{haoma, paulhsu}@microsoft.com

ABSTRACT

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 importance of the scenarios at their intersection: providing an integrated search and information discovery user experience. In this paper, 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.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Search Process, Information Filtering

Keywords

Entity Recommendation; Entity Graph; Personalization; Search Click Log; User Behavior Analysis

1. INTRODUCTION

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 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 [3] and website recommendation [16] [28]. 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 this movie, as shown in Figure 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.

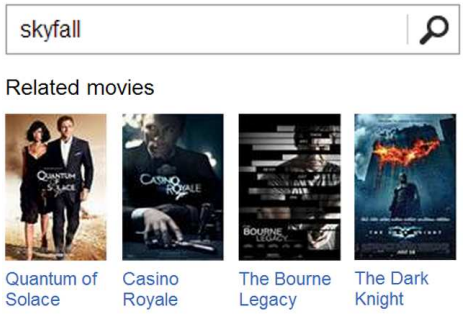


Figure 1: Related entity suggestion feature on major search engines

Figure 2: Personalized entity recommendation scenario studied in this paper. When a user types movie related queries in a search engine, like "movies", "skyfall", we show movie recommendations for this user based on his/her past user log.

*The research was performed during an internship of the first author at Microsoft Research.

WSDM'14, February 24–28, 2014, New York, New York, USA.

http://dx.doi.org/10.1145/2556195.2556233.

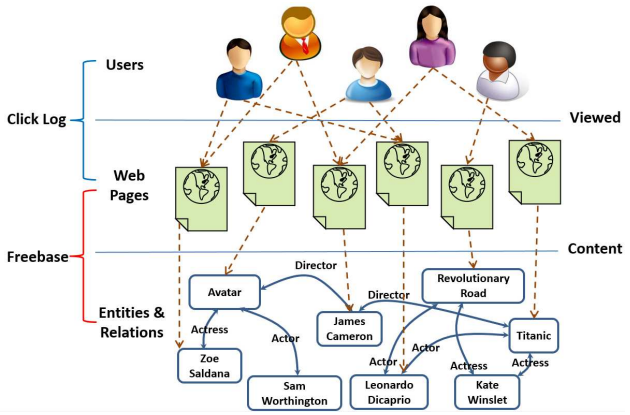


Figure 3: Heterogeneous relations between users, web pages and entities

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 paper.

By taking advantage of user click log and knowledge extracted from Freebase[1], 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 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.
- 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 5 demonstrate the effectiveness of the proposed framework over traditional entity recommendation techniques.

[1]http://www.freebase.com/

The major contributions of this paper 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 paper is organized as follows. Section 2 introduces the background and preliminaries of this paper. Section 3 discusses features and heuristics in user click log and the Freebase entity graph. Section 4 introduces the proposed entity recommender framework. Experiments and results are discussed in Section 5. Related work, conclusions and future works are presented at the end of the paper.

2. BACKGROUND AND PRELIMINARIES

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

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 5.

The top two layers ("Users" and "Web Pages") in Figure 3 illustrate the relationship between users and web pages in user click log.

2.2 Freebase Entity Graph

Freebase [1] 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 [14] [19].

Each entity in Freebase are associated with some URLs that are related to this entity. For example, for the movie entity "Jobs"[2], by utilizing the relationships "/common/topic/

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

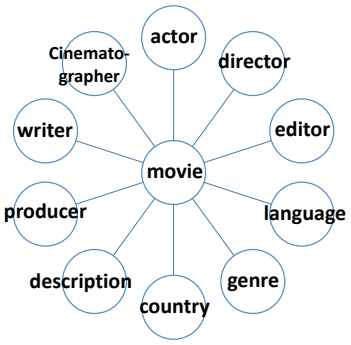


Figure 4: Partial movie related entity graph schema

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.*

Formally, a Freebase entity graph is a heterogeneous information network [23], which contains multi-typed entities, relationships and attributes collected from different domains. Similar to an entity-relation diagram in relational databases, we use an abstract graph to represent the entity and relationship type restrictions in Freebase, which is usually referred to as the *entity graph schema*, denoted as $G_T = (\mathcal{A}, \mathcal{R})$. An example of a partial entity graph schema for the movie domain can be found in Figure 4.

The bottom two layers ("Web Pages" and "Entities & Relations") in Figure 3 present the relationship between web pages and Freebase entity graph. Moreover, in this paper, we use the phrases "Freebase", "knowledge base" and "entity graph" interchangeably.

2.3 Mapping URLs to Entities

With the definitions in Section 2.1 and Section 2.2, we can now map users' clicked URLs in user log to the corresponding entities in Freebase, as demonstrated in Figure 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.

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:

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 1.

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.

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.

Table 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

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:

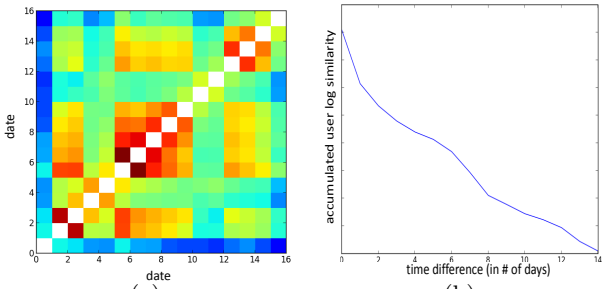
$$r(\hat{e}_T^u, L_T^u) \propto \sum_{e_t^u \in L_T^u} P(\hat{e}_T^u | e_t^u). \quad (1)$$

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 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 5(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 5(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 5: 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)

Table 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

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 4 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 [13] [23] 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.

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 2.

We first observe that the two ranking lists are substantially different. Comic Book Resources[3] users are mostly interested in comic book based sci-fi movies, including Spider-Man, Superman, Thor, *etc.* Rue La La[4] 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.

[3]comicbookresources.com, a comic book discussion website
[4]ruelala.com, a fashion boutique with mostly female customers

Table 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	*movie*1 is a prequel of *movie*2
is_sequel	*movie*1 is a sequel of *movie*2
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

3.4 Feature Calculation

Based on the above discussion, in Table 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 [23]; 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, following [23], we can calculate the similarity score between these two entities with Equation 2:

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

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. RECOMMENDATION FRAMEWORK

With the features and heuristics discussed in Section 1 and 3, we present the proposed entity recommendation framework in this section.

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 3):

$$r(\hat{e}_T^u; L_T^u, \theta) = \sum_{e_t^u \in L_T^u} w_t(\hat{e}_T^u, e_t^u) \sum_{k=1}^{K} \theta_k S_k(\hat{e}_T^u, e_t^u), \qquad (3)$$

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:

$$g(e_T^u; L_T^u, \theta) = r(e_T^u; L_T^u, \theta) - r(e_m^u; L_T^u, \theta), \qquad (4)$$

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

$$O_g(\theta) = -\sum_u g(e_T^u; L_T^u, \theta) + \frac{\lambda}{2}\|\theta\|_2^2, \qquad (5)$$

where λ controls L_2 regularization to prevent over-fitting.

Objective function defined in Equation 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 12) by defining the margin as follows:

$$g\prime(e_T^u; L_T^u, \theta) = RR(e_T^u; L_T^u, \theta) - RR(e_m^u; L_T^u, \theta), \qquad (6)$$

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 only requires the calculation of the recommendation scores of e_T^u and e_m^u. While to utilize Equation 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 can achieve similar effectiveness as Equation 6. Hence, in this paper, we use Equation 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.

4.2 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:

$$r(\hat{e}_T^u; L_T^u, \theta_T^u) = \sum_{e_t^u \in L_T^u} w_t(\hat{e}_T^u, e_t^u) \sum_{k=1}^{K} S_k(\hat{e}_T^u, e_t^u)\theta_k^u. \qquad (7)$$

Different from Equation 3, personalized recommendation model requires a set of parameters θ^u for each user.

When learning personalized models, directly maximizing margins between e_T^u and e_m^u 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 5). To alleviate such problem, when learning personalized models, we use both the user log history L_T^u of the target user, as well as other "similar" user log sequences, namely the *neighbors* of L_T^u, denoted by $N(L_T^u)$. We provide the formal definition of *neighbors* as well as an efficient approach to generate neighbors in Section 4.3.

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

$$\begin{aligned} O_u(\theta^u) = \quad & - \sum_{t=1}^{T} w_t(e_T^u, e_t^u) g(e_t^u; L_t^u, \theta^u) + \\ & \lambda_1 \sum_{L_{T\prime}^{u\prime} \in N(L_T^u)} w_u(L_{T\prime}^{u\prime}, L_T^u) g(e_{T\prime}^{u\prime}; L_{T\prime}^{u\prime}, \theta^u) + \\ & \tfrac{1}{2}\lambda_2 \quad \|\theta^u - \theta_g\|_2^2, \end{aligned} \qquad (8)$$

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 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 8 models the consistency and drift of user interest (Section 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_t^u. 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 tar-

get 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 7 and propose a new neighborhood based personalized recommendation model in Equation 9:

$$r_n(\hat{e}_t^u; L_T^u, \theta^u) = r(\hat{e}_t^u; L_T^u, \theta^u) + \lambda_1 \sum_{L_{T\prime}^{u\prime} \in N(L_T^u)} w_u(L_{T\prime}^{u\prime}, L_T^u) r(\hat{e}_t^u; L_{T\prime}^{u\prime}, \theta^u). \quad (9)$$

In this neighborhood based recommendation function, besides personalized recommendation function defined in Equation 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 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 7 and Equation 9 are presented in Section 5.

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:

$$w_u(L_T^u, L_{t\prime}^{u\prime}) = \frac{r(E; L_T^u, \theta_g) \cdot r(E; L_{t\prime}^{u\prime}, \theta_g)}{\|r(E; L_T^u, \theta_g)\| \|r(E; L_{t\prime}^{u\prime}, \theta_g)\|}, \quad (10)$$

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 [26].

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 Parameter Estimation

In Equation 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 non-continuous and both objective functions (Equations 5 and 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 5 and 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 with a fixed e_m^u as follows. The gradients of the two objective functions can be obtained accordingly:

$$\frac{\partial g}{\partial \theta_k} = \sum_{e_t^u \in L_T^u} w_t(e_T^u, e_t^u)\Big(S_k(e_T^u, e_t^u) - S_k(e_m^u, e_t^u)\Big). \quad (11)$$

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 [5], 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.

5. EXPERIMENTS

In this section, we implement both global and personalized entity recommendation models proposed in Section 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.

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 be-

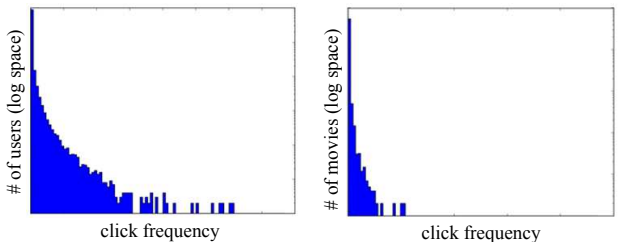


Figure 6: User Log Distribution

tween 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 6.

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.

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:

$$MRR = \frac{1}{|D_{test}|} \sum_{i=1}^{|D_{test}|} \frac{1}{rank(e_T^u)}, \tag{12}$$

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 paper are implemented as follows:

- Global recommendation model: estimate global recommendation model θ_g using Equation 5 and recommend entities with Equation 3;
- Personalized recommendation model without neighbor data: estimate personalized recommendation model θ_T^u with Equation 8 and recommend entities with Equation 7. We abbreviate this method as *PRM*;
- Personalized recommendation model with neighbor data: estimate personalized recommendation model θ^u with Equation 8 and recommend entities using Equation 9. We abbreviate this method as *PRM-KNN*.

Table 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

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;
- 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 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 [6], 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 6, 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 paper are the lower bounds of the actual performances of these methods. Performance results of all the recommendation approaches are presented in Table 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 [11], 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, 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 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 7 as well as neighbor similarity function (Equation 10). Neighborhood similarity threshold in Equation 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 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.

5.3 Personalized Recommendation Model Performance

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

5.3.1 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 7(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 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.

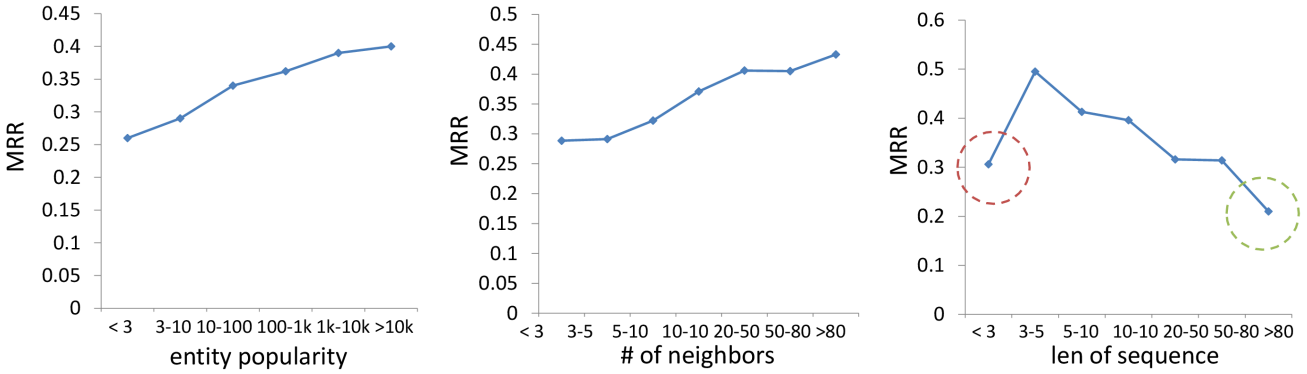
5.3.2 With Different Neighborhood Sizes

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 7(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 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.

5.3.3 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 7(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



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

Figure 7: Personalized Framework Performance Analysis

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 7(c) is due to lack of data, and as shown in Figure 6, 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.

6. RELATED WORKS

In this section, we review two related research areas, which are (1) the existing recommender systems, and (2) the information network analysis methods.

6.1 Recommender Systems

Recommender systems have received increasing attention and have been actively studied in recent years due to the various applications in E-commerce and other Web applications. Collaborative filtering techniques, which includes neighbor-based approaches and model-based approaches, are popular and widely applied. Neighbor-based methods usually study similarity calculation methods among users (user-based) [2] [20] or items (item-based) [15] [22]. User-based collaborative filtering methods provide recommendation to target users by first finding similar users and then collecting and analyzing neighbor 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 try to fit user-item rating data into different models (Bayesian Network [27], matrix factorization [12] [21], or clustering models [25] *etc*) and use the learned models to recommend items to users in unseen scenarios. 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 nicely. Many studies have been done using this technique to interpret different heuristics in user-item rating datasets [10]. In [6], Hu et al. propose to model implicit feedback as positive and negative preferences with different confidence levels with a matrix factorization based approach. Koren et al. incorporate temporal information in matrix factorization models to further improve the recommendation quality [11].

Besides traditional recommender systems, researchers proposed hybrid approaches to incorporate both user-item rating dataset as well as other contextual information in different scenarios, including social network information, time information, *etc*. For instance, social trust or friend aware recommender approaches model trustworthiness or similarities of users [7] [17] [27]. Koren in [11] proposed a time-aware collaborative filtering method that can effectively incorporate time information into the matrix factorization framework. Middleton et al. propose to use ontological user profiling technique to build recommendation system [18]. Notice that ontology is substantially different from the entity graph we employed in this study. An ontology usually summarizes the concept levels of certain domains, *e.g.*, "text classification" is a subtopic of "machine learning", and the size of which is usually small. Entity graph is built with real world entities. The size of entity graphs can be of very large-scale. Yu et al. [30] [31] introduced meta-path concept into hybrid recommender systems. However, these works usually focus on the heterogeneous entity relationships, and can not fully take advantage of the rich features in user click log and Freebase.

The proposed framework in this paper belongs to hybrid recommender system category. Different from previous works, our work models various features and heuristics generated from user click log and Freebase, which suits the application better than transitional methods, thus can achieve more promising performances.

6.2 Information Network Analysis

Heterogeneous information networks which contain multi-typed entities and links are the general data format of entity graphs. Information network analysis and mining have gained wide attention in both academia and industry. Many researchers believe that the heterogeneity and rich-relation nature make information network a good data representation in many scenarios. A lot of information network mining and learning tasks have been done in the past couple of years, including clustering [24], classification [9], and link prediction [29] *etc*. Studies regarding entity similarity measurements, as a fundamental technique, have been actively engaged in many research works as well [4] [8] [23]. Researchers also discover 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 [13] [23]. These works also motivated user guided data mining and analysis with information networks [24].

7. CONCLUSION AND FUTURE WORKS

In this paper, 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 paper 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.

Acknowledgments

The work was supported in part by the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA) and U.S. National Science Foundation grants CNS-0931975, IIS-1017362, IIS-1320617, IIS-1354329.

8. REFERENCES

[1] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, SIGMOD '08, pages 1247–1250, Vancouver, Canada, 2008.

[2] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, UAI'98, pages 43–52, San Francisco, CA, USA, 1998.

[3] H. Cao, D. Jiang, J. Pei, Q. He, Z. Liao, E. Chen, and H. 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.

[4] S. 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.

[5] Q. Gu and J. 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.

[6] Y. Hu, Y. Koren, and C. Volinsky. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, ICDM '08, pages 263–272, 2008.

[7] M. Jamali and M. 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.

[8] G. Jeh and J. 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.

[9] M. Ji, J. Han, and M. 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.

[10] Y. 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.

[11] Y. 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.

[12] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

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

[14] T. Lin, P. Pantel, M. Gamon, A. Kannan, and A. 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.

[15] G. Linden, B. Smith, and J. York. Amazon.com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE*, 7(1):76–80, 2003.

[16] H. Ma, C. Liu, I. King, and M. R. Lyu. Probabilistic factor models for web site recommendation. In *In Proceedings of SIGIR*, 2011.

[17] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. 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.

[18] S. E. Middleton, N. R. Shadbolt, and D. C. De Roure. Ontological user profiling in recommender systems. *ACM Trans. Inf. Syst.*, 22(1):54–88, Jan. 2004.

[19] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. In *In Proceedings of ACL*, 2009.

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

[21] J. D. M. Rennie and N. 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.

[22] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, WWW '01, pages 285–295, Hong Kong, Hong Kong, 2001.

[23] Y. Sun, J. Han, X. Yan, S. P. Yu, and T. Wu. PathSim: Meta Path-Based Top-K Similarity Search in Heterogeneous Information Networks. In *Proceedings of the 37th international conference on Very large data bases*, VLDB '11, 2011.

[24] Y. Sun, B. Norick, J. Han, X. Yan, P. S. Yu, and X. Yu. Integrating meta-path selection with user-guided object clustering in heterogeneous information networks. In *Proceedings of SIGKDD'12*.

[25] L. Ungar and D. Foster. Clustering methods for collaborative filtering. In *AAAI Workshop on Recommendation Systems*, number 1, 1998.

[26] S. Vempala. *The random projection method*, volume 65. Amer Mathematical Society, 2005.

[27] Y. Wang and J. Vassileva. Bayesian network-based trust model. In *Proceedings of the 2003 IEEE/WIC International Conference on Web Intelligence*, WI '03, 2003.

[28] R. W. White, P. Bailey, and L. Chen. Predicting user interests from contextual information. In *Proceedings of SIGIR*, 2009.

[29] X. Yu, Q. Gu, M. Zhou, and J. Han. Citation prediction in heterogeneous bibliographic networks. In *SDM*, 2012.

[30] X. Yu, X. Ren, Q. Gu, Y. Sun, and J. Han. Collaborative filtering with entity similarity regularization in heterogeneous information networks. In *IJCAI HINA*, 2013.

[31] X. Yu, X. Ren, Y. Sun, B. Sturt, U. Khandelwal, Q. Gu, B. Norick, and J. Han. Entity recommendation in heterogeneous information networks with implicit user feedback. In *RecSys*, 2013.