Relation-aware Blocking for Scalable Recommendation Systems

Huizhi Liang
Newcastle University
United Kingdom
huizhi.liang@newcastle.ac.uk

Zehao Liu
University of Reading
United Kingdom
marshallcnliu@gmail.com

Thanet Markchom
University of Reading
United Kingdom
t.markchom@pgr.reading.ac.uk

ABSTRACT

Recommender systems contain rich relation information. The multiple relations in a recommender system form a heterogeneous information network. How to efficiently find similar users and items based on hop-n relations in heterogeneous information networks is one significant challenge to develop scalable recommender systems in the era of big data. Hashing has been popularly used for dimensionality reduction and data size reduction. Current hashing techniques mainly focus on hashing for directly related (i.e. hop-1) features. This paper proposes to develop relation-aware hashing techniques to bridge this gap. The proposed approaches use locality sensitive hashing (LSH) and consider hop-n relations in an information network to construct user or item blocks. They help facilitate efficient neighborhood formation and recommendation making. The experiments conducted on a large-scale real-life dataset show that the proposed approaches are effective.

ACM Reference Format:
Huizhi Liang, Zehao Liu, and Thanet Markchom. 2022. Relation-aware Blocking for Scalable Recommendation Systems. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM '22), October 17–21, 2022, Atlanta, GA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3511808.3557682

1 INTRODUCTION

Recommender systems can help users deal with the information overload issue. In recommender systems, there are multiple entities including users, items, and affiliated entities such as tags and item categories. They contain strong social ties among users such as friendship relation, and weak social ties such as "following" and "mention" relations. Moreover, user behaviours such as tagging, rating, and review behaviours connect users and items and form multiple relations among users, items, user-generated content, meta-data of users (e.g., age, occupation, and gender), and meta-data of items (e.g., item images, description, categories, and brand). The multiple relations form a complex interconnected heterogeneous information network [14]

Collaborative filtering (CF) approaches such as user-based and item-based K-nearest neighbor methods are widely used to make recommendations in various areas [2]. They have high recommendation accuracy performance, high explainability, and are easy to implement. Neighborhood formation is the key component of CF based recommender systems. Typically, pair-wise comparisons such as cosine similarity calculation are commonly used to build the correlations (i.e. find the the nearest neighbors of each user or item). With exponential computational complexity, CF based recommender systems are known as having low scalability. How to decrease the number of pair-wise comparisons and find the nearest neighbor users and candidate items quickly is an important research question, especially in the era of big data.

Blocking or indexing techniques can help to significantly decrease the number of comparisons [19]. The objects in a database can be inserted into one or more blocks according to some blocking criteria, such that only objects within a block are compared with each other. Locality Sensitive Hashing (LSH) [16] is an approximate blocking approach that uses a set of hash functions to map data objects such as users or items within a certain distance range into the same block with a given probability. It can filter out those data objects with low similarities for a given data object, thus decreasing the number of comparisons [17]. LSH can generate blocks quickly and has advantages such as dimension reduction, noise tolerance, and similarity-preserving. It has been widely used in industries, such as personalized news recommendation in Google [13] and real-time recommender systems [3].

Current hashing techniques mainly focus on hashing for content features or directly related (i.e. hop-1) relation features [10]. For a heterogeneous information network, for a given node, meta-path based approaches can be used to find pre-defined hop-n relation features that are represented by a set of connected nodes. In this paper, we propose to use LSH and meta-path based approaches to construct blocks for nodes based on hop-n ($n > 1$) relations in the information network. Besides relation information, content information or attributes of nodes such as temporal features, text, and images play an important role to find similar nodes in various applications. For example, find users or items with similar semantic topic interests [15], visual preferences [4], and temporal patterns [12]. For a given node, jointly considering both content and hop-n relational features to find similar nodes is needed in many scenarios. In this paper, we propose a hierarchical hybrid hashing techniques to consider both content and hop-n relations. The proposed hashing techniques will be used to construct user and item blocks. Based on the constructed user and item blocks, we apply CF approach to form neighborhood and make recommendations efficiently. The main contributions are summarized as follow:

- We proposed relation-aware blocking methods for hop-n relations.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CIKM '22, October 17–21, 2022, Atlanta, GA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9236-5/22/10…$15.00
https://doi.org/10.1145/3511808.3557682

- We proposed a hierarchical hybrid hashing method to jointly consider content and hop-n relations.
- We applied the proposed blocking techniques to recommender systems and conducted experiments on a real-life dataset.

The rest of the paper is organised as follows. In Section 2, we will briefly review the related work. The proposed relation-aware blocking approaches will be discussed in Section 3. The application of the proposed approaches to recomemnder systems will be discussed in Section 4. The experiments and conclusions will be discussed in Section 5 and 6.

2 RELATED WORK

Hashing is a process that maps original high-dimension data into lower dimensional data blocks, in which a hash function can be a mathematical function, machine learning model or neural networks [19]. In some studies, the original data is also called reference data or input data, and the output of hash function can be referred to as hash codes, compact codes, hash address, signature or fingerprint [19]. Some hashing methods try to avoid collision, which can be used to detect errors and verify data integrity, as small changes in original data will consequence in significantly different hash code. Others used mathematical methods to map data into blocks using collision, but it cannot preserve similarity relationship in the original data [11]. Locality Sensitive Hashing (LSH) has been proposed to maximize the similarity of data in the same buckets, and it minimizes the similarity between buckets [16]. Wang et. al. [20] used domain knowledge to build a taxonomy tree of concepts to provide semantic similarity reference for LSH, and they combined textual and semantic similarity in LSH for entity blocking. Learning to hash is a data-related method, which learns hash functions from training sets and aims to keep the same similarity between input space and encode space. PreHash [18] built a novel user preference representation by learning hashing buckets from user's historical interactions. Current techniques mainly focus on hashing for content features or directly related (i.e. hop-1) relation features [10]. How to design hashing techniques to consider hop-n relation and jointly consider both hop-n relation and content information still need to be explored.

3 THE PROPOSED BLOCKING APPROACH

3.1 Hop-n Relation hashing

For a heterogeneous information network, meta-path based approaches can be used to find pre-defined relation features (represented by a set of connected nodes or weighted nodes) for each node [10]. Let $\mathcal{N}_i$ denote a node type, $\mathcal{R}_{ij}$ denote a relation type between node $\mathcal{N}_i$ and $\mathcal{N}_j$. A meta-path $\mathcal{M}$ is a sequence of node types and relation types.A meta-path defined as $\mathcal{M} = (ij...kl) = \mathcal{N}_i \overset{\mathcal{R}_{ij}}{\rightarrow} \mathcal{N}_j ... \mathcal{N}_k \overset{\mathcal{R}_{kl}}{\rightarrow} \mathcal{N}_l$. A **path** is a sequence of nodes and relations in an information network G. The **length** of $\mathcal{M}$ (denoted as $|\mathcal{M}|$) is the number of relations (i.e., **hops**) in $\mathcal{M}$. For example, let U denote Users, P denote Items, and A denote Actors, meta-path $\mathcal{M} = (UPA)$ connects users with actors via users' rated or tagged items.

For a given node v, let $\mathbf{F}_v$ denote the hop-n relation feature following a given hop-n meta-path $\mathcal{M} = (ij...x)$, we can extract hop-n relation features which is the set of connected destination nodes $\mathbf{F}_v = \mathcal{V}_l = \{v | v \in \mathcal{N}_l\}$, where $\mathcal{V}_l$ denote the set of nodes of type $\mathcal{N}_l$ connected with v via a path following $\mathcal{M}$. Usually, we use a binary adjacency matrix to represent a relation $\mathcal{R}_{ij}$ and use matrix multiplication to calculate context-free hop-n relation features following meta-path $\mathcal{M}$ and aggregate all the middle nodes and destination nodes for context-sensitive hop-n relation features. For any given two nodes v and v', their hop-n relation similarity can be measured by the Jaccard similarity or cosine similarity of their hop-n relation features. As Jaccard similarity does not require the two sets to have the same dimension, thus it is more suitable for graph with different number of neighbour nodes. We used Jaccard similarity to measure the relation similarity of two nodes in this paper as follows: $sim(v, v') = J(\mathbf{F}_v, \mathbf{F}_{v'}) = \frac{|\mathbf{F}_v \cap \mathbf{F}_{v'}|}{|\mathbf{F}_v \cup \mathbf{F}_{v'}|}$ where J denotes the Jaccard similarity function. It represents the ratio of the size of the intersection of the elements of the two feature sets to that of the union of the elements of the two feature sets.

LSH can help find approximate results for a query node for high dimensional data [6]. Minwise hashing (minHash) is a popular approximation approach that estimates the Jaccard similarity [5]. This hashing method applies a random permutation $\prod$ on the elements of any two hop-n relation feature sets $\mathbf{F}_v$ and $\mathbf{F}'_v$ and utilizes

$$\mathscr{P}(min(\prod \mathbf{F}_v)) = \mathscr{P}(min(\prod \mathbf{F}_{v'})) = J(\mathbf{F}_v, \mathbf{F}_{v'}) \quad (1)$$

to estimate their Jaccard similarity, where $min(\prod \mathbf{F}_v)$ denotes the minimum value of the random permutation of the elements of hop-n feature set $\mathbf{F}_v$. $\mathscr{P}$ denotes the hash collision probability. A minHash function can generate a basic signature for a given node. The basic signature is called a length-1 signature and the hash function is called a length-1 hash function. In order to control the similarity threshold to allocate nodes that have similarity above the pre-defined threshold into the same block, k length-1 hash functions can be combined to form a length $k(k > 1)$ compound LSH scheme [6].

Obtaining multi-hop relation features via matrix multiplication is usually computationally expensive, especially for long meta-paths. For a length l meta-path, let n be the average number of adjacent nodes per node, the time cost for obtaining multi-hop relation is approximately n^l for each given starting node. Thus, blocking based on hop-n relations may result in exponential time complexity and causing the scalability issue [9]. We discuss the proposed scalable hop-n relation hashing approach in the following sub-section.

3.2 Scalable Hop-n Relation Hashing

Graph sampling is a common way to sample a sub set of nodes of a big graph. There are various kinds of graph sampling approaches for homogeneous information networks, for example, traverse based, random walk, snowball, forest fires. Meta-path guided random walk is usually used for heterogeneous information networks [8]. In this paper, we propose to combine meta-path guided graph sampling approach with LSH approach for long hop-n relations in large graphs. For a given meta-path $\mathcal{M} = (ij...kx)$, starting from source node v with type $\mathcal{N}_i$, it can walk along the path and reach target nodes with type $\mathcal{N}_j$. The walk transition probability from node v to another node v' with type $\mathcal{N}_j$ following $\mathcal{R}_{ij}$ can be calculated based on the ratio of the weight of the edge $l_{vv'}$ over the total number of weights of all outgoing edges of v with relation $\mathcal{R}_{ij}$. This paper assumes each connection (i.e., edge) between any two nodes with the same type is equally important. Let O_v be the set of outgoing

edges of v with relation $\mathcal{R}_{ij}$ of meta-path $\mathcal{M}$. The walk transition probability from node v to another v' following $\mathcal{R}_{ij}$ can be calculated as: $Pr\,(v, v') = \left(\frac{w_{vv'}}{\sum_{v'' \in O_v} w_{vv''}}\right)^{\rho}$.

Where $\rho \in [0,1]$ is a parameter to tune the impact of the out-degree in the random walk propagation process. For simplicity, the weight of each edge is set to 1, $w_{vv'} = w_{vv''} = 1$ in this paper.

For meta-path $\mathcal{M} = (ij...kx)$, the transition probability from starting node type $\mathcal{N}_i$ to target node type $\mathcal{N}_x$ can be calculated by the multiplication of the probability of one node to the neighbor nodes of each relation following $\mathcal{M}$, as show by the following formula:

$$Pr\,(v_i, v_x|\mathcal{M}) = Pr\left(v_i, v_j\right) * ... * Pr\,(v_k, v_x) \tag{2}$$

With this approach, for a given node v and meta-path $\mathcal{M}$, we can get a sample of hop-n relation features denoted as $\tilde{F}_v$ with random walk based sampling. To increase the probability of having the same features (i.e., connected nodes), we conducted random walk that starts from node v and walk along $\mathcal{M}$ for r times to conduct OR-construction (disjunction) of the obtained feature nodes sets of each random walk $\tilde{\mathbf{F}}_v = \bigcup_{i=1}^{r} \mathcal{W}_i$, where $\mathcal{W}_i$ denotes the obtained feature node sets of the i-th random walk. Let s be the probability of one target node appearing in the obtained relation feature set $\tilde{\mathbf{F}}_v$. s can be considered as the prior of a node of $\tilde{\mathbf{F}}_v$. It is amplified to

$$s = 1 - (1 - Pr(v_i, v_x|\mathcal{M}))^r \tag{3}$$

After we get the relation feature set, we can randomly permute the elements of the feature set $\tilde{\mathbf{F}}_v$ and use minHash to get the approximation of the Jaccard similarity of two feature sets. Thus, we can combine meta-path based random walk and LSH hashing to conduct hop-n relations efficiently. For a given node v and hop-n meta-path M, let $\mathcal{H}_{r,k,l}(v, \mathcal{M})$ denote a LSH family that has r times of random walk following a hop-n meta-path $\mathcal{M}$ and l length-k hash functions. When the probability of one node appears in the approximate node feature set $\tilde{F}_v$ is s (as defined in formula 3), the collision probability of $\mathcal{H}_{r,k,l}$ can be estimated with $\mathscr{P} = 1 - (1 - t^k)^l$, where t denotes the collision probability of a length-1 minHash function.

3.3 Hierarchical hybrid hashing Scheme

We can first use a hash function to put similar nodes that have content similarity above a threshold into the same block. In this way, we will aggregate similar users' connected relations together. For easy description, we call this kind of block nodes level-0 block nodes. Let $\mathcal{B}_v^0$ denote level-0 block nodes of v, $\mathcal{H}_0$ denote a LSH hash scheme at this level $\mathcal{B}_v^0 = \mathcal{H}^0(v)$. After that, we can connect the level-0 block nodes with the entity nodes of original heterogeneous information network following a given hop-n meta-path. The proposed hop-n relation hashing scheme will be used to construct block nodes. We call this kind of block nodes level-1 block nodes. Let $\mathcal{B}_v^1$ denote level-1 block nodes of v, $\mathcal{H}_1$ denote a LSH hash scheme at this level, $\mathcal{B}_v^1 = \mathcal{H}^1(v, \mathcal{M})$ following meta-path $\mathcal{M}$. With the same process, we can construct level-m block nodes ($\mathcal{B}_v^m = \mathcal{H}^m(v)$) based on the meta-path that connecting level-(m-1) block nodes and entity nodes. To be general, we use hierarchical hashing schemes $\mathbb{H} = \{\mathcal{H}^0, \mathcal{H}^1, ..., \mathcal{H}^m\}$ to find approximately similar nodes considering both content and hop-n relation similarity of nodes, following meta-paths at different levels.

For example, for a given user u, we can construct user blocks with a two level hierarchical hashing scheme $\mathbb{H} = \{\mathcal{H}^0, \mathcal{H}^1\}$. With this scheme, we first get the blocks of item nodes $\mathcal{B}_p = \mathcal{H}^0(p)$. Then for a given user u, following meta-path $\mathcal{M}_1 = (U\mathcal{B}_pA)$, we find actors that are collected from all the items that have been allocated to the same item block $\mathcal{B}_p$ with user u's tagged/rated items. When we construct user block $\mathcal{B}_u = \mathcal{H}^1(u, \mathcal{M}_1)$ following meta-path $\mathcal{M}_1$. We can see that the block nodes that formed with meta-path (UPA) only considers the relations between users, items and actors, while $(U\mathcal{B}_pA)$ considers both the content similarity of items and the relations between users, items and actors.

4 SCALABLE RECOMMENDER SYSTEMS

This section discusses how to make Top-N recommendations. The *Neighbourhood* based CF approaches are popularly-used recommendation approaches, which are based on user-item hop-1 relation. Comparing with matrix factorisation and deep learning models, they are simple, explainable, and easy to implement. Neighbourhood formation is the process of generating like-minded peers (i.e., the K-nearest neighbours) for a target user $u \in U$. Typically, we can calculate the pair-wise cosine similarity of the neighbor user and $u \in U$ to select K-nearest neighbor users. To find the neighbourhood of each target user $u \in U$ quickly, we construct user blocks for users and items respectively. With the LSH blocking scheme $\mathbb{H}$, we can get signatures for user u. The users that have the same signature will be allocated into the same block. Parameters decides the similarity threshold of a block. The users in the same blocks with user u are the neighbor users of u. Thus, we can form neighbourhood quickly via hashing. In this way, instead of considering all users in the dataset, this approach can filter out users that have low similarities with the target user u, thus decreasing the number of pairwise comparisons. With hashing, we also can update the neighborhood of users quickly after users update their item preferences.

For each target user u, let $\mathcal{P}$ denote the items set and O_u denote the set of u's observed items. The prediction score of how much u will be interested in an unobserved candidate item $p \in \mathcal{P} - O_u$ is calculated by considering the similarities between user u and those users who are neighbours of u and have rated item p [1]:

$$\mathcal{L}\,(u, p) = \sum_{u' \in \left(\tilde{N}_u \cap L_j\right)} sim\left(u, u'\right) \tag{4}$$

Where L_j denotes the user nodes that item node p has linked based on **hop**-1 relation $\mathcal{R}_{UP}{}^{-1}$ (i.e., those users that have rated item p). The Top-N items with high prediction scores will be recommended to the target user u.

5 EXPERIMENTS

5.1 Dataset

We conducted experiments on the following real-world dataset:

- **HetRec2011-MovieLens Dataset**: This is an extension version based on MovieLens 10M dataset. It has rich information of movies and users review. There were 10197 movies, 2113 users and 47,957 tagging records. Each tagging behaviour has user ID, Tag ID, movie ID and timestamp information. On

Table 1: Statistics of filtered dataset

Dataset	Node	Count	Relation	Count
MovieLens	User (U)	1132	R_{UP}	24530
	Movie (P)	3767	R_{PT}	19230
	Tag (T)	4693	R_{UT}	11173

average, each user created 22.696 tags, and each movie was assigned 8.117 tags.

In order to reduce sparsity in dataset, we filtered out those users and items that have less than 2 records. Also, we split 80% of samples for training and 20% for testing. The statistics of the dataset after pre-processing are shown in Table 1.

In this paper, we use the temporal features as content information for a node. In our experiments, we used 17 manual temporal features: *year, month, day, hour, minute, second, week_of_year, day_of_week, day_in_month, quarter, is_month_start, is_month_end, is_quarter_start, is_year_start, is_leap_year, is_quarter_end, is_year_end*. In order to obtain the temporal features of a given node, we first collected all the timestamps of the edges of a given node (e.g., tagging behaviour). Then we get all the temporal features. For each selected temporal feature, we first scaled the values to 0 and 1, and then we calculated the mean, median, maximum, minimum values of all temporal features of a given node as statistical values of temporal features. The temporal features are used as content information. All of our experiments were conducted on a machine using NVIDIA V100 16GB GPU and 128GB RAM.

5.2 Recommendation Results

Top-N recommendation task is popularly used for implicit or binary ratings [7]. As the task is to recommend items, we only keep User-Item $\mathcal{R}_{UP}$ relation in the test set. For a target user, we randomly selected 20% of User-Item relation as test set, the rest User-Item relation and the affiliated information forms the training set. The Mean Average Precision@N (MAP@N) and Mean Recall@N (Recall@N) of all test users are used to measure the overall **Accuracy** performances of recommendation approaches. This work conducted Top-N (N = [1,5,10,50,100]) recommendation tasks on the dataset. Moreover, to evaluate the efficiency, the total recommendation time over all test users is used. We also measured the peak Memory Usage of each method.

To evaluate the effectiveness of the proposed approaches, we compared the performances of the following approaches.

- ***CF***: is the typical user based collaborative filtering (CF) approach. It is based on the user-item hop-1 relation.
- ***RH***: is the user based CF approach based on the proposed relation-aware blocking scheme.
- ***SRH***: is the user based CF approach based on the proposed scalable relation-aware blocking scheme.

For ***RH*** approach, we used the following 2 meta-paths: $\mathcal{M}_1 = (UPT)$ and $\mathcal{M}_2 = (UPU)$. For fair comparisions, we used the same set of meta-paths of ***RH*** for the scalable hop-n relation hashing approach ***SRH***. To jointly consider both content and hop-n relation, we used ***RH*** approach to construct a two-level hierarchical hybrid hashing scheme. It first constructed tag blocks and user blocks based on their content information (i.e. temporal features). For tags T, the tag block $\mathcal{B}_T$ is generated by level-0 LSH hashing $\mathcal{H}^{00}$, $\mathcal{B}_T = \mathcal{H}^{00}(T)$. For users, the user block $\mathcal{B}_U$ is generated by level-0 LSH hashing $\mathcal{H}^{01}$, $\mathcal{B}_U = \mathcal{H}^{01}(U)$. For level-1 LSH hashing, the following three meta-paths are used: $\mathcal{M}_3 = (UPT\mathcal{B}_T)$, $\mathcal{M}_4 = (UPU\mathcal{B}_U)$), where $\mathcal{B}_T$ and $\mathcal{B}_U$ denote the tag block node type and user block node type respectively. In the experiments, $\rho = 1$. We tested different parameter settings for the proposed approaches. For ***RH*** and ***SRH***, the best parameter setting is $r = 5$ and $l = 16$, $k = 1$. The graph sampling rate was set to 0.6 for ***SRH***.

As shown in figure 1, ***SRH*** has better MAP@N and Recall@N than ***RH*** for all the selected meta-paths. For the same meta-path, ***SRH*** consumed less run time and memory than ***RH***. It indicates that the scalable hop-n hashing method is effective. ***RH*** based on $\mathcal{M}_3$ and $\mathcal{M}_4$ are hierarchical hybrid hashing schemes. We can see that they have better accuracy and run time than those non-hybrid hashing schemes that only consider hop-n relations (i.e., ***RH*** based on $\mathcal{M}_1$ and $\mathcal{M}_2$). However, they consumed slightly more memory usage than the relation only approach. It indicates that the proposed hierarchical hybrid hashing schemes can effectively leverage content information to improve Top-N recommendation performance. Compared with the baseline recommendation models ***CF***, both proposed methods, ***RH*** and ***SRH***, achieved better accuracy, had less run time, but consumed similar memory usage. Overall, the proposed approaches ***SRH*** and ***RH*** outperformed the baseline model ***CF***.

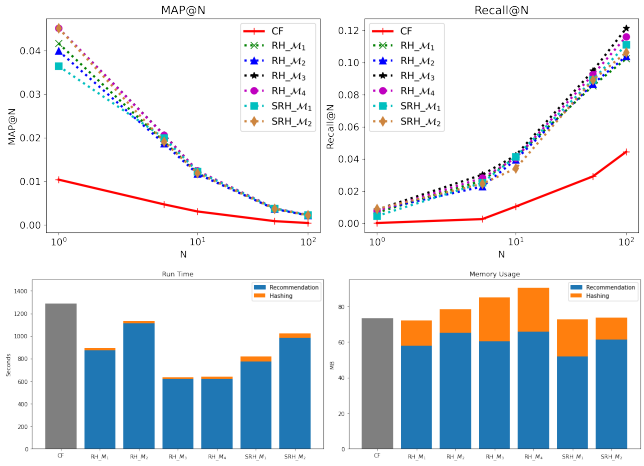


Figure 1: Accuracy, Run Time, and Memory Usage for MovieLens Dataset

6 CONCLUSIONS

This paper proposed relation-aware hashing techniques to consider hop-n relations in information network framework for scalable recommender systems. We also proposed a hierarchical hybrid hashing scheme to jointly consider hop-n relation and content information to construct node blocks. Temporal features of a node is used as content information in the experiments. The experiments conducted on the popular large-scale real-world dataset demonstrates the effectiveness of the proposed approaches.

REFERENCES

[1] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering* 17, 6 (June 2005), 734–749.

[2] Charu C. Aggarwal. 2016. An Introduction to Recommender Systems. In *Recommender Systems*. Springer International Publishing, Cham, 1–28. https://doi.org/10.1007/978-3-319-29659-3_1

[3] Badrish Chandramouli, Justin J Levandoski, Ahmed Eldawy, and Mohamed F Mokbel. 2011. Streamrec: a real-time recommender system. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*. 1243–1246.

[4] Wei-Ta Chu and Ya-Lun Tsai. 2017. A hybrid recommendation system considering visual information for predicting favorite restaurants. *World Wide Web* 20, 6 (2017), 1313–1331.

[5] Ondrej Chum, James Philbin, Andrew Zisserman, et al. 2008. Near duplicate image detection: Min-hash and TF-IDF weighting.. In *Bmvc*, Vol. 810. 812–815.

[6] Mayur Datar, Nicole Immorlica, Piotr Indyk, and Vahab S Mirrokni. 2004. Locality-sensitive hashing scheme based on p-stable distributions. In *Proceedings of the twentieth annual symposium on Computational geometry*. 253–262.

[7] Mukund Deshpande, George Karypis, G Karypis, and • M Deshpande. 2004. Item-Based Top-N Recommendation Algorithms. , 143-177 pages. Issue 1.

[8] Wei Feng and Jianyong Wang. 2012. Incorporating heterogeneous information for personalized tag recommendation in social tagging systems. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2012), 1276–1284. https://doi.org/10.1145/2339530.2339729

[9] Li Gao, Hong Yang, Jia Wu, Chuan Zhou, Weixue Lu, and Yue Hu. 2018. Recommendation with Multi-Source Heterogeneous Information. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*. 3378–3384.

[10] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S. Yu. 2018. Leveraging meta-path based context for top-n recommendation with a neural co-attention model. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 1 (2018), 1531–1540. https://doi.org/10.1145/3219819.3219965

[11] Ming Hu and Yan Wang. 2009. MD5-based error detection. In *Proceedings of the 2009 Pacific-Asia Conference on Circuits, Communications and System, PACCS 2009*. 187–190. https://doi.org/10.1109/PACCS.2009.147

[12] Yehuda Koren. 2009. Collaborative filtering with temporal dynamics. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2009), 447–455. https://doi.org/10.1145/1557019.1557072

[13] Jiahui Liu, Elin Pedersen, and Peter Dolan. 2010. Personalized News Recommendation Based on Click Behavior. In *2010 International Conference on Intelligent User Interfaces*.

[14] Jian Liu, Chuan Shi, Binbin Hu, Shenghua Liu, and Philip S. Yu. 2017. Personalized ranking recommendation via integrating multiple feedbacks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 10235 LNAI (2017), 131–143. https://doi.org/10.1007/978-3-319-57529-2_11

[15] Qi Liu, Enhong Chen, Hui Xiong, Chris HQ Ding, and Jian Chen. 2011. Enhancing collaborative filtering by user interest expansion via personalized ranking. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 42, 1 (2011), 218–233.

[16] Anand Rajaraman, Jeffrey David Ullman, Anand Rajaraman, and Jeffrey David Ullman. 2012. Chapter 3: Finding Similar Items. In *Mining of Massive Datasets*. 53–107. https://doi.org/10.1017/cbo9781139058452.004

[17] Kexin Rong, Clara E. Yoon, Karianne J. Bergen, Hashem Elezabi, Peter Bailis, Philip Levis, and Gregory C. Beroza. 2018. Locality-sensitive hashing for earthquake detection: A case study of scaling data driven science. *Proceedings of the VLDB Endowment* 11, 11 (2018), 1674–1687. https://doi.org/10.14778/3236187.3236214 arXiv:1803.09835

[18] Shaoyun Shi, Weizhi Ma, Min Zhang, Yongfeng Zhang, Xinxing Yu, Houzhi Shan, Yiqun Liu, and Shaoping Ma. 2020. Beyond User Embedding Matrix: Learning to Hash for Modeling Large-Scale Users in Recommendation. *SIGIR 2020 - Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* 61672311 (2020), 319–328. https://doi.org/10.1145/3397271.3401119

[19] Jingdong Wang, Ting Zhang, Jingkuan Song, Nicu Sebe, and Heng Tao Shen. 2018. A Survey on Learning to Hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 4 (apr 2018), 769–790. https://doi.org/10.1109/TPAMI.2017.2699960 arXiv:1606.00185

[20] Qing Wang, Mingyuan Cui, and Huizhi Liang. 2016. Semantic-aware blocking for entity resolution. In *2016 IEEE 32nd International Conference on Data Engineering, ICDE 2016*. 1468–1469. https://doi.org/10.1109/ICDE.2016.7498378