Two Birds One Stone: On both Cold-Start and Long-Tail Recommendation

Jingjing Li
University of Electronic Science and Technology of China
School of ITEE, The University of Queensland
lijin117@yeah.net

Ke Lu
University of Electronic Science and Technology of China
kel@uestc.edu.cn

Zi Huang
School of ITEE, The University of Queensland
huang@itee.uq.edu.au

Heng Tao Shen
University of Electronic Science and Technology of China
shenhengtao@hotmail.com

ABSTRACT

The number of "hits" has been widely regarded as the lifeblood of many web systems, e.g., e-commerce systems, advertising systems and multimedia consumption systems. However, users would not hit an item if they cannot see it, or they are not interested in the item. Recommender system plays a critical role of discovering interested items from near-infinite inventory and exhibiting them to potential users. Yet, two issues are crippling the recommender systems. One is "how to handle new users", and the other is "how to surprise users". The former is well-known as cold-start recommendation, and the latter can be investigated as long-tail recommendation. This paper, for the first time, proposes a novel approach which can simultaneously handle both cold-start and long-tail recommendation in a unified objective.

For the cold-start problem, we learn from side information, e.g., user attributes, user social relationships, etc. Then, we transfer the learned knowledge to new users. For the long-tail recommendation, we decompose the overall interested items into two parts: a low-rank part for short-head items and a sparse part for long-tail items. The two parts are independently revealed in the training stage, and transfered into the final recommendation for new users. Furthermore, we effectively formulate the two problems into a unified objective and present an iterative optimization algorithm. Experiments of recommendation on various real-world datasets, such as images, blogs, videos and musics, verify the superiority of our approach compared with the state-of-the-art work.

CCS CONCEPTS

• **Information systems → Social recommendation**;

KEYWORDS

Recommender system; cold-start recommendation; long-tail recommendation

MM'17, October 23–27, 2017, Mountain View, CA, USA.
© 2017 ACM. ISBN 978-1-4503-4906-2/17/10…$15.00
DOI: http://dx.doi.org/10.1145/3123266.3123316

1 INTRODUCTION

Imagine, for a minute, some fantastic things among which one may experience in his life. What have you got in your mind? Love at first sight? Yes, that is fantastic! It is fantastic because it involves *NEW* and *SURPRISE*. Love at first sight happens. Happens a lot between human beings. Unfortunately, it rarely happens between users and on-line multimedia consumption systems as what they yearn for. Why? Check the recommender system [1, 5, 6, 22, 27]. It might have no idea about how to handle new users, or it is mediocre and cannot surprise any customer. Even worse, both of them. In fact, these two issues are the most challenging problems that cripple a fantastic recommender system.

Handling new users, or items, is well-known as the cold-start problem [10, 16, 30] in recommender systems. The user cold-start problem concerns recommendation for a new user who did not have any historical record in the target system. Currently, the state-of-the-art technology pervasively deployed in recommender systems is collaborative filtering (CF) [11, 12, 19, 26, 29]. CF, unfortunately, is built on past interactions between the user and the system. Thus, it obviously cannot handle the cold-start problem. Cold-start problem has been regarded as a quite challenging problem in the community [30]. With no direct information available, previous work advocate just recommending the most popular items to new users, or resort to learning from side-information to facilitate the cold-start [13, 28, 32, 38]. The side-information can be user attributes (e.g. gender, age, occupation and location) [10] or user's social connections (e.g., user's friend groups) [7, 31]. However, only recommending the most popular items may give the user a negative impression of not being taken seriously and further considers that the recommender system is mediocre. Studying side-information gives us an opportunity to learn more about the user and then recommend items based on the user's attributes. For instance, suppose that we found out married young males spend most of their money on beers and diapers, we should also recommend beers and diapers for a new user who is a married young male.

The essence of side-information based cold-start is to recommend the items to a new user where the recommended items are popular among old users who have common characteristics with the new user. Compared with the strategy of recommending the system-wide popular items, the side-information based recommendation makes a user feeling that the system knows something about

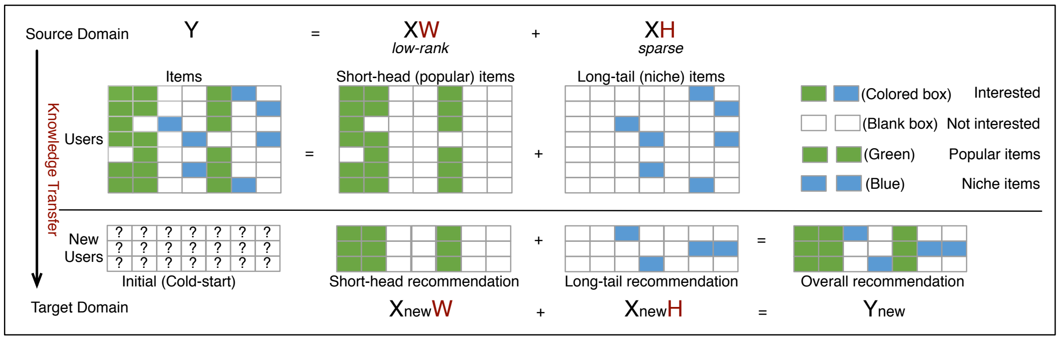


Figure 1: Illustration of the proposed method. We handle cold-start recommendation by transfer learning, and we decompose the recommendations into two part, a low-rank part to address short-head items and a sparse part to handle long-tail items.

him. People enjoy being taken seriously. However, only know something is not enough. Considering a real-world multimedia system, e.g., youtube.com, there would be massive users with each one has only single-digit of available attributes. In other words, hundred of thousands of users would be share the same combination of attributes [35]. The recommender system, therefore, would treat these users with no discrimination. As a result, the system can recommend some interested items to the new user, but there is much of a chance it would lose the *individuality*. In fact, losing the individuality is a pervasive problem because previous work [4, 14, 31] generally preserve only principal components [37]. Individuality recommendations surprises users, while commonness recommendations end up boring users. For example of music recommendation, Mick is a big fun of pop music, he joins several music sharing groups, e.g., the group of the Beatles and the group of Bob Dylan. He is also the creator, as well as a member, of the group of Leonard Cohen. Since the Beatles and Bob Dylan are more popular than Leonard Cohen, both the group of the Beatles and the group of Bob Dylan are much larger than the group of Leonard Cohen, let us say there are 500, 500 and 50 members in each group, respectively. The recommender system would recommend either *Hey Jude*[1] or *Like a Rolling Stone*[2] for Mick with great confidence because most of his social connections are in the first two groups, regardless of the fact that Mick's favorite singer is Leonard Cohen and his favorite song is *Bird On a Wire*[3]. Therefore, we should pay close attention to individuality recommendations, which can be achieved by long-tail recommendation [25, 37].

From another perspective, the perspective of items, the most popular items are well-known by users, they are willing to consume them with or without recommendation. However, a niche market needs more recommendations and it also more rewarding for both users and systems [2]. The niche items are also known as long-tail items [2]. Compared with popular item recommendation, long-tail recommendation has at least three benefits: 1) The market of popular items are highly competitive, thus its profit is very limited. Long-tail items, however, embrace relatively large marginal profit [37]. 2) Long-tail items can further boost the sales for the reason of "one-stop shopping convenience" effect [2]. 3) As we stated before, recommending long-tail items will surprise the users, and increase customer loyalty and satisfaction.

[1]A popular song by the Beatles.
[2]A popular song by Bob Dylan.
[3]A popular song by Leonard Cohen.

Most of previous work, e.g., CF models [32], matrix factorization models [14] and probabilistic topic models [4], would fail either cold-start recommendation or long-tail recommendation for the reason that those models either depend on past interactions or preserve only principal components (short-head items) and ignore the niche factors (long-tail items). This paper proposes a novel approach to challenge both of them. Technically, our approach is motivated by the following two insights:

1) Since there is no direct information available for cold-start problem, we learn from side information. At first, mapping functions which map side information to user-item relationship are learned on the training dataset. Then, we transfer the learned knowledge to new users.
2) Given a system of near-infinite inventory, popular items are only a small fraction of the total items. These items, however, have positive relationship with a majority of users. In a user-item matrix, as illustrated in Fig. 2, the popular items can be revealed by a low-rank function [20]. At the same time, long-tail items are a major portion of the inventory, but they have positive relationship with only a handful of users. Thus, their relationship can be represented by a sparse function. We, therefore, deploy two complementary parts, a low-rank part and a sparse part, in our formulation to handle both the principal components and the niche factors, respectively.

At last, we formulate the cold-start recommendation and long-tail recommendation into a unified objective, and propose an effective optimization strategy. The main contribution of this work is that we, for the first time, challenge both cold-start and long-tail recommendation in a unified objective. The main ideas of this paper are illustrated in Fig. 1.

The remainder of this paper is organized as follows, section 2 gives a brief review of related work and highlight the difference of our model. Section 3 is the problem definition, formulation and optimization. Experiments are presented in section 4. At last, section 5 is the conclusion.

2 RELATED WORK

Since there is no previous work challenging both cold-start recommendation and long-tail recommendation in a unified framework.

Here we review some related work which focus on either cold-start commendation or long-tail recommendation.

2.1 Cold-Start Recommendation

Among the models which address cold-start recommendation, we focus on the ones which exploit side-information to facilitate the cold-start problem. Those models can be roughly grouped in three categories, e.g., the similarity based models [28, 32], matrix factorization models [15, 23] and feature mapping models [10].

The similarity based models are straightforward. Their formulations can be expressed as

$$Y_t = WY_s, \tag{1}$$

where W is the similarity matrix. Generally, W is the similarity between the relationships Y_s and Y_t, the subscripts s and t denote "source" and "target", respectively. However, Y_t is unavailable in cold-start problems, so W represents the similarity between user side-information. W can be calculated by several ways, typically based on the cosine similarity.

Matrix factorization models generally factorize the relationship matrix into two latent representations by optimizing the following objective

$$\min_{U,V} \|Y - UV\|_F^2 + \Omega(U, V), \tag{2}$$

where Ω is the regularization used to avoiding over-fitting. For cold-start problems, one can learn a shared U or V from the side-information, and then use it to predict Y.

Feature mapping models normally learn a feature mapping between the side-information and one of the latent feature representations. The differences between the matrix factorization models and the feature mapping models is that in matrix factorization models the shared U is jointly learned from Y and the the side-information, while in feature mapping models, one needs to learn an additional feature mapping, and further learn different U_s and U_t by sharing the feature mapping. More details can be found in [10, 31].

2.2 Long-Tail Recommendation

For the long-tail recommendation problems, Yin et al. [37] propose a novel suite of graph-based algorithms. They represent user-item information with undirected edge-weighted graph and apply hitting time algorithm for long-tail item recommendation. Park and Tuzhilin [25] handle the long-tail recommendation by clustering algorithm. They split the whole itemset into the head and the tail parts and clusters only the tail items. Then recommendations for the tail items are based on the ratings in these clusters and for the head items on the ratings of individual items. Szpektor et al. [34] focus on the long-tail problem in query. They extend the reach of query assistance techniques to long-tail queries by reasoning about rules between query templates rather than individual query transitions. Shi [33] proposes a graph-based recommendation approach which trade-off among multiple criteria of measurements, such as accuracy, similarity, diversity, and long-tail. Domingues et al. [9] study the long-tail recommendation in the application of one-line music recommender systems.

The existing long-tail recommendation approaches, however, still have some problems. Firstly, some of them, e.g., [37] and [9], are overdoing a bit by only recommending long-tail items and ignoring the popular items. Can you imagine a cellphone recommendation list without iPhone on it? I would not risk that. Both popular items and niche items should be considered in an "one-stop shopping" website. Popular items help bring users, and niche items increase customer loyalty. Secondly, none of them considers long-tail recommendation in cold-start problem. As we afore stated, long-tail items in cold-start recommendation will surprise the new users and attract them for staying.

To challenging these issues, this paper proposes a novel approach which considers both cold-start and long-tail recommendation in an integrated optimization problem.

3 THE PROPOSED METHOD

3.1 Notations

In this paper, we use lowercase and uppercase letters to represent vectors and matrices, respectively. A sample is denoted as a vector, e.g., x. Specifically, we use U to denote the user matrix, and I to denote the item matrix. $|U|$ and $|I|$ are the total number of users and items. Let $Y \in \{0, 1\}^{|U|*|I|}$ be the relationship matrix between U and I, where $Y_{ui} = 1$ means there is a positive interaction between user u and item i. The positive interaction can be defined in several ways depending on specific application, e.g., for a e-commercial system, the interaction can be purchase, for a video system, the interaction can be watching and for a text system, the interaction can be reading.

3.2 Definitions

Definition 1: Side information $X \in \mathbb{R}^{|U|*|S_s|}$ is defined as users' attributes, preferences or social relationships, e.g., X_{us} can be used to measure how much user u prefers item s in the related, or auxiliary, domain S_s.

Definition 2: Cold-start recommendation addresses the problem of recommendation for a new user u where u has no prior interactions in the target system, i.e., $Y[u, :]$ is unknown.

Definition 3: Long-tail recommendation addresses the problem of recommendation item i for user u where (1) i is related to u and (2) i lies in the long tail.

Problem 1: Given a target system S_t and a related source system S_s, user u is new for S_t, but it has interactions in S_s, recommend top k items, i.e., $i_1, i_2, \cdots, i_k$ for u in S_t, where i_j ($j \in [1, k]$) are close enough to u and some of i_j lie in the long tail.

3.3 Problem Formulation

It is worth noting that most previous recommenders preserve only principal components (short-head) and ignore the niche factors (long-tail). They would fail to surprise the users. Yin et al. [37] proposed a suite of graph-based algorithms for the long-tail recommendation. However, Yin et al. [37] focus on recommendations of only long-tail items. It will surprise the users, but it is overdoing a bit to make users confused. As we stated before, can you imagine a cellphone recommendation list without iPhone on it? We advocate the best situation is to recommend items which are a mixture of popular items and niche items. Popular items to make users feel reasonable and niche items to make them feel surprised. In other words, an optimal recommender system should be out of expectation but within understanding. To achieve this, we propose to decompose the relationship matrix Y into two parts, one for short-head, and the other for long-tail. As a result, we have

$$Y = Y_{SH} + Y_{LT} \tag{3}$$

where Y_{SH} is used to formulate the relationship between users and short-head items, while Y_{LT} is used to specify the relationship between users and long-tail items.

Sedhain et al. [31] found out that various cold-start recommenders can be boiled down to a linear model represented as

$$Y = XW \tag{4}$$

where W is used to map the user attributes or social relationship to the user-item matrix. For specific models, W can either be directly learned (W is explicit in the objective) or be derived (W is implicit in the objective), e.g., the objective function of [10] can be written as the form of Eq. (4) with $W = (X^\top X)^{-1}X^\top Y$.

As shown in Eq. (4), let $Y_{SH} = XW$ and $Y_{LT} = XH$ respectively, our objective can be formulated as follows

$$\min_{W,H} L(Y, XW, XH) + \Omega(XW, XH), \tag{5}$$

where L is loss function, and Ω is regularization.

For a real system with millions of users and near-infinite inventory, the popular items are only a tiny fraction (short-head) of total items. To capture this, we propose that XW should be low-rank. On the other hand, The frequency of long-tail items is essentially low. As a result, XH should be sparse. So, our formulation can be written as

$$\min_{W,H} \|Y - XW - XH\|_F^2 + \alpha \text{rank}(XW) + \beta \text{sparse}(XH), \tag{6}$$

where $\|\cdot\|_F$ is the Frobenius norm, $\alpha > 0$ and $\beta > 0$ are two balancing parameters. Constraints rank$(\cdot)$ and sparse$(\cdot)$ enable a matrix to be low-rank and sparse, respectively. It is worth noting that this formulation is not trivial, it is sound and can be verified. For a better understanding, we randomly choose 200 users from the Flickr dataset [35], and illustrate their preferences in Fig. 2. It is obvious that popular items can be captured by a low-rank [8, 17, 20] matrix, and the long-tail items can be represented by a sparse [18, 39] matrix.

Eq. (6) encompasses several approaches with different choices of rank$(\cdot)$ and sparse$(\cdot)$. Since rank minimization in Eq. (6) is a NP-hard problem, previous work generally use the trace norm $\|\cdot\|_*$ as a surrogate of rank$(\cdot)$. As a result, a straightforward approach of Eq. (6) can be written as:

$$\min_{W,H} \|Y - XW - XH\|_F^2 + \alpha\|XW\|_* + \beta\|XH\|_1, \tag{7}$$

where the trace norm $\|XW\|_*$ is used to encourage low-rankness on XW, and $\|XH\|_1$ is used to encourage sparsity on XH. Although $\|\cdot\|_*$ has been widely used as a surrogate of rank$(\cdot)$ in existing work, $\|\cdot\|_*$ is an implicit form of the low-rank constraint. It controls the single values of XW, but the changes of the single values do not always lead to a change of the rank. Thus, we propose to use an explicit form of low-rank constraint as follows:

$$\min_{W,H} \|Y - XW - XH\|_F^2 + \alpha \sum_{i=r+1}^{d} (\sigma_i(XW))^2 + \beta\|XH\|_1, \tag{8}$$

where $\sigma_i(XW)$ is the i–th singular value of XW, d is the total number of singular values of XW. $\sum_{i=r+1}^{d}(\sigma_i(XW))^2$ explicitly solves the problem of minimizing the square sum of r-smallest singular value of XW.

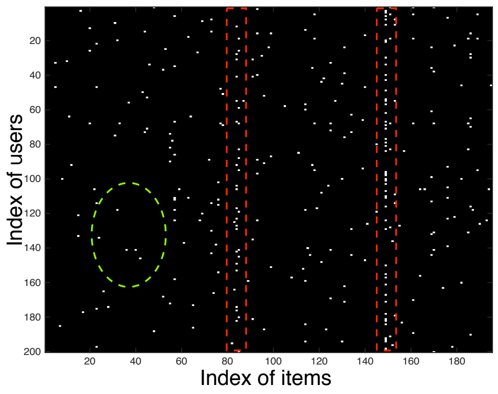


Figure 2: Illustration of the user/item matrix. White dot means there is a positive relationship between corresponding user and item. Obviously, items in the red boxes are popular items, while others, e.g., the ones marked in the green circle, are long-tail items.

Note that

$$\textstyle\sum_{i=r+1}^{d}(\sigma_i(XW))^2 = \text{tr}(V^\top(XW)(XW)^\top V), \tag{9}$$

where tr is the trace operator of a matrix, and V are the singular vectors which corresponds to the $(d-r)$-smallest singular values of $(XW)(XW)^\top$. Thus, our final objective function can be written as:

$$\begin{aligned}\min_{W,H} &\|Y - XW - XH\|_F^2 + \alpha \text{tr}(V^\top XWW^\top X^\top V) + \\ &\beta\|XH\|_1 + \gamma_1\|W\|_F^2 + \gamma_2\|H\|_F^2\end{aligned} \tag{10}$$

where $\gamma_1 > 0$ and $\gamma_2 > 0$ are two penalty parameters. The F-norm on W and H are introduced to avoid over-fitting. In this paper, we set $\gamma_1 = 1$ and $\gamma_2 = 1$ for simplicity.

Now, we can learn a low-rank function W and a sparse function H by solving Eq. (10). Function W captures the commonality of different users sharing the same interests, and function H captures the personality of specific users. Thus, for a new user with its attribute matrix (or social relationship matrix) X_{new}, he will be mapped to the user-item matrix

$$Y_{new} = X_{new}W + X_{new}H. \tag{11}$$

Note that it is a transfer strategy. We learn W and H from Eq. (10), and transfer the learned knowledge to new users in Eq. (11). It is also worth noting that both short-head items and long-tail items are included in Y_{new}. Thus, our approach not only can handle cold-start problem, but also can handle long-tail recommendation.

3.4 Problem Optimization

Since Eq. (10) is not convex over all variables, we resort to an alternative optimization strategy. Specifically, we optimize only one variable at a time and keep the others fixed. Thus, Eq. (10) can be solved by alternatively solving the two subproblems: (1) solving the short-head recommendation W and (2) solving the long-tail recommendation H.

Solving the short-head recommendation W: When H is fixed, we can optimize W via:

$$\min_{W} \|Y - XW - XH\|_F^2 + \alpha \mathrm{tr}(V^\top XWW^\top X^\top V) + \gamma_1 \|W\|_F^2. \quad (12)$$

Note that the problem involves both V and W. We alternatively update them. At first, by treating V as a constant, we calculate the deviation to W and set it to zero:

$$X^\top (Y - XW - XH) = \alpha X^\top VV^\top XW + \gamma_1 W. \quad (13)$$

Then, W has the following closed solution:

$$W^* = (\alpha X^\top VV^\top X + \gamma_1 I + X^\top X)^{-1} X^\top (Y - XH), \quad (14)$$

where I is the identity matrix with appropriate size. After getting W, we can update V by Eq. (9).

Solving the long-tail recommendation H: When W is fixed, we can optimize H via:

$$\min_{H} \|Y - XW - XH\|_F^2 + \beta \|XH\|_1 + \gamma_2 \|H\|_F^2. \quad (15)$$

Since the ℓ_1−norm involves both X and H, and the constraint is not smooth. To handle this, we introduce an auxiliary variable $Z = XH$. Then, Eq. (15) can be rewritten as the following equivalent problem:

$$\min_{H,Z,E} \|Y - XW - Z\|_F^2 + \beta \|Z\|_1 + \gamma_2 \|H\|_F^2 + \mu \|XH - Z + E\|_F^2, \quad (16)$$

where $\mu > 0$ is a penalty parameter, and E is the scaled dual variable.

For Z, we optimize it by fixing H and E. Then, Eq. (16) can be reduced as:

$$\min_{Z_j} \|XH_j - Z_j + E\|_F^2 + \frac{\beta}{\mu} \|Z_j\|_1. \quad (17)$$

Then, Z can be optimized via soft thresholding operation as:

$$Z_j^* = \mathrm{soft}(XH_j + E, \frac{\beta}{\mu}), \quad (18)$$

where

$$\mathrm{soft}(a, b) = \mathrm{sign}(a) \max(|a| - b, 0). \quad (19)$$

For H, by fixing Z and E, we have:

$$\min_{H_j} \sum_{i=1}^{|U|} \|Y_{ij} - x_i W_j - x_i H_j\|_F^2 + \gamma_2 \|H\|_F^2 + \mu \|XH_j - Z_j + E\|_F^2. \quad (20)$$

Then, the gradient w.r.t. H_j can be calculated as:

$$\begin{aligned} \Delta_{H_j} \mathcal{J} &= 2(\sum_{i=1}^{|U|} x_i x_i^\top + \gamma_2 I + \mu X^\top X) H_j \\ &-2(\sum_{i=1}^{|U|} x_i^\top (Y_{ij} - x_i W_j) + \mu X^\top (Z_j - E)) \end{aligned} \quad (21)$$

By setting Eq. (21) to zero, we have the closed form of H_j as:

$$\begin{aligned} H_j^* = (\sum_{i=1}^{|U|} x_i x_i^\top + \gamma_2 I + \mu X^\top X)^{-1} (\sum_{i=1}^{|U|} x_i^\top (Y_{ij} - x_i W_j) + \\ \mu X^\top (Z_j - E)) \end{aligned} \quad (22)$$

At last, the scaled dual variable E can be updated as:

$$E^* = E + HX - Z. \quad (23)$$

Thus, the optimal H can be achieved by alternatively updating Z, E and H. For clarity, we show the key steps of our algorithm in Algorithm 1.

Algorithm 1. *On both cold-start and long-tail recommendation*

```
Input: User-item matrix $Y$, user attributes $X$, parameters $\alpha$, $\beta$, and $\mu$.
Output: Recommended items for new users.
Warm-up:
  Repeat
    1. Optimize the short-head recommendation function $W$:
       $W^* = (\alpha X^\top VV^\top X + \gamma_1 I + X^\top X)^{-1} X^\top (Y - XH)$.
    2. Optimize the long-tail recommendation function $H$:
       $H_j^* = (\sum_{i=1}^{|U|} x_i x_i^\top + \gamma_2 I + \mu X^\top X)^{-1} * \Gamma$,
       where $\Gamma = (\sum_{i=1}^{|U|} x_i^\top (Y_{ij} - x_i W_j) + \mu X^\top (Z_j - E))$.
    3. Update the dual variable $E$:
       $E^* = E + HX - Z$.
  Until Convergence
Cold-start:
  $Y_{new} = X_{new} W + X_{new} H$.
Recommendation:
  Using the logistic regression function to predict the recommendation
  probability of items, and recommend the top-$k$ items.
```

3.5 Computational Complexity

Here we analysis the computational complexity of Algorithm 1 by big O notation. Suppose there are n users and m items, the optimization of W costs $O(dn^2 + d^3 + dnm)$. The optimization of H costs $O(dn^2 + d^3 + d^2 m + dnm)$. Thus, the overall computation costs is $O(n^2)$. It is worth noting that the calculation of V involves eigen-decomposition of $XWW^\top X$, which would be time-consuming with large-scale dataset. As a surrogate, we can use the trace norm $\|\cdot\|_*$ instead of rank$(\cdot)$, as shown in Eq. (7). Although it is an implicit constraint of low-rank, we can use it to reduce the time complexity with the cost of some effectiveness. Specifically, from previous work [21], we know that

$$\|W\|_* = \min_{U,V: W=UV^\top} \frac{1}{2}(\|U\|_F^2 + \|V\|_F^2) \quad (24)$$

where if $W \in \mathbb{R}^{m*n}$, then $U \in \mathbb{R}^{m*r}$ and $V \in \mathbb{R}^{n*r}$. As a result, we can handle U and V instead of W [36]. The time complexity can be significantly reduced for the reason $r \ll \min(m, n)$.

4 EXPERIMENTS

4.1 Data Description

In this section, we evaluate the proposed approach on four real-world datasets which include images, texts, videos and musics recommendation.

- **Flickr** [35] is a dataset collected from *flickr.com*[4], which is a popular personal photos managing and sharing website. Users in flickr can tag photos and subscribe photos in terms of tags with which he is interested. For instance, a user can subscribe photos with tag "baseball". The evaluated dataset consists of 80,513 users, 195 interest groups as the items, and a social network with 5,899,882 links.
- **BlogCatalog** [35] is a dataset collected from *blogcatalog.com*[5], which is a popular blog collaboration system. Any article published by a blogger in blogcatalog can be cataloged into some groups according to the topics, e.g.,

[4]http://www.flickr.com

[5]http://www.blogcatalog.com

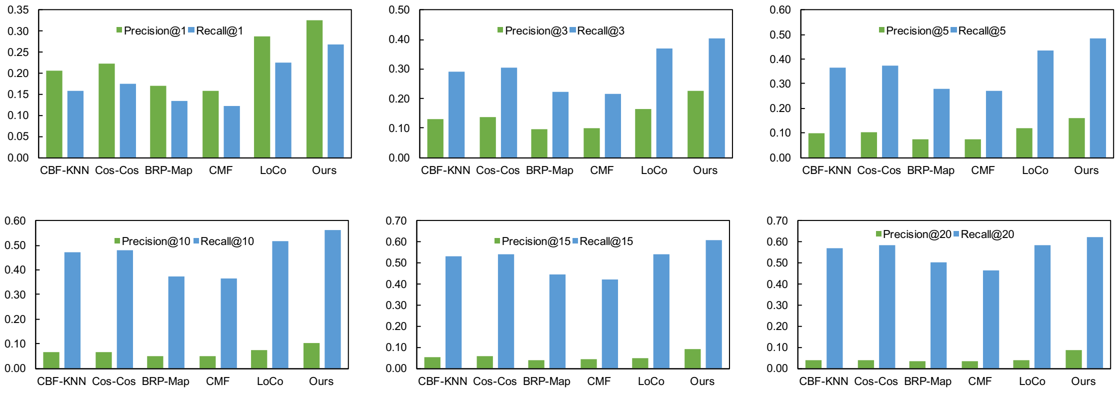


Figure 3: Experimental results on Flickr dataset. The vertical axis denotes accuracy rate.

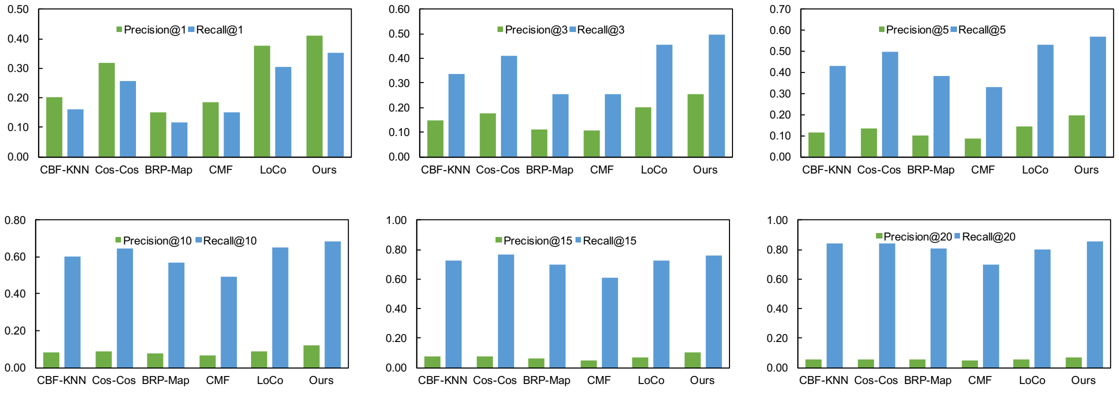


Figure 4: Experimental results on BlogCatalog dataset. The vertical axis denotes accuracy rate.

"sports", "business" and "technology". The tested dataset consists of 10,312 users, 39 topics as items, and a social network with 333,983 links.

- **YouTube** [35] is a dataset collected from *youtube.com*[6], which is a popular video watching and sharing website. Users in YouTube can also subscribe interested topics. The evaluated dataset consists of 1,138,499 users, 47 categories as items, and a social network with 2,990,443 links.
- **Hetrec11-LastFM** [3] is a dataset collected from *last.fm*[7], which is a online music system. Hetrec11-LastFM contains social networking, tagging, and music artist listening information. The tested dataset consists of 1,892 users, 17,632 artists as items, and 186,479 tag assignments.

4.2 Compared Methods

We compare our method with five previous work detailed as follows:

- **CBF-KNN** [10] is a straightforward recommender system based on user similarity. The user similarity is calculated from user-attributes.
- **Cos-Cos** [32] is a neighborhood-based methods for cold-start CF in a generalized matrix algebra framework.
- **BPR-Map** [10] is a method that maps entity (e.g. user or item) attributes to the latent features of a matrix (or higher dimensional) factorization model.
- **CMF** [15] is a multi-relational factorization framework using Bayesian personalized ranking.
- **LoCo** [31] is a low-rank linear regression method for cold-start recommendation.

4.3 Experimental Protocols

For the evaluated datasets, we split each of them into two subsets, one includes 10% of the users as new users (test dataset) for cold-start, and the remainder of 90% users are collected as training data for warm-up. The new users are randomly selected, so we build 10 training-test folds and report the average results.

Following previous work [31], we deploy the widely used precision@k, recall@k and mean average precision (mAP@100) as the measurements. All the hyper-parameters in the objective are tuned by cross-validation. One can also directly set $\lambda = 1$, $\beta = 0.1$ and $\mu = 1$ for simplicity. The parameters sensitivity is reported in section 4.5.

4.4 Experimental Results

Fig. 3 shows the experimental results on Flickr dataset. We report both precision and recall (@1, @3, @5, @10, @15 and @20). It can

[6]http://www.youtube.com
[7]http://www.last.fm

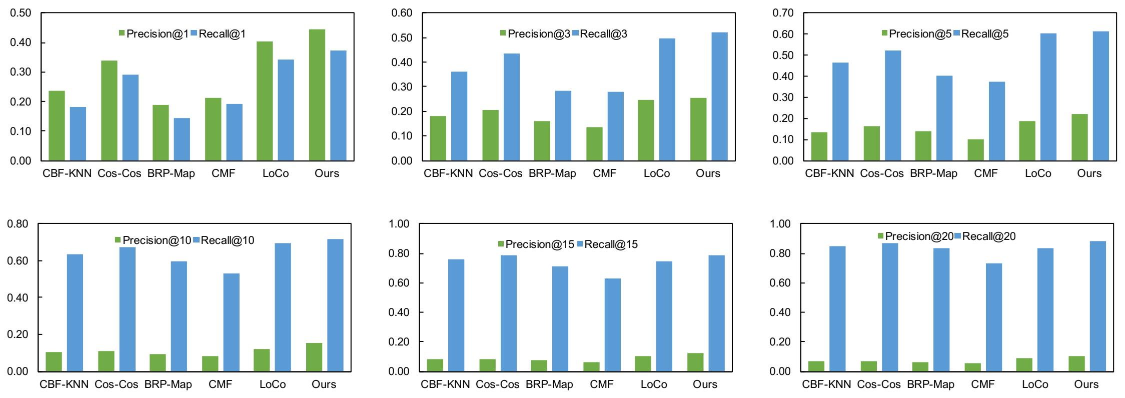


Figure 5: Experimental results on YouTube dataset. The vertical axis denotes accuracy rate.

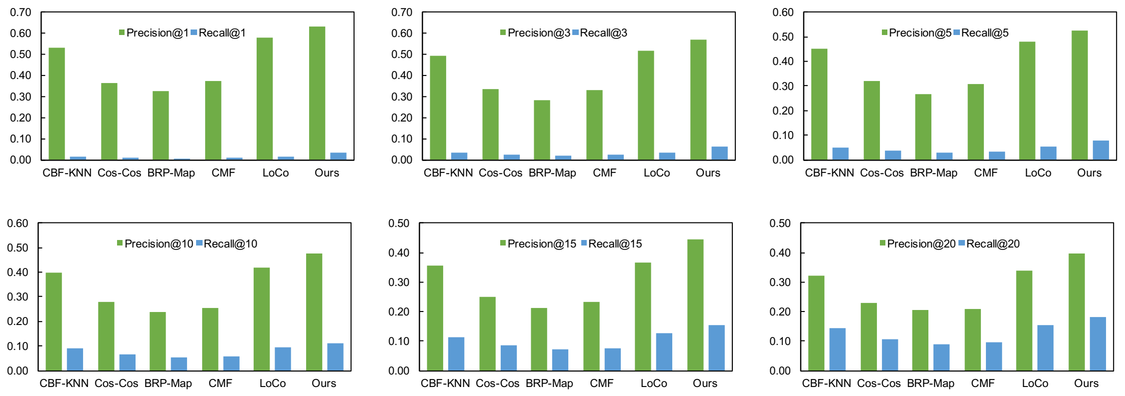


Figure 6: Experimental results on Hetrec11-LastFM dataset. The vertical axis denotes accuracy rate.

be seen that our approach always performs the best with respect to either precision or recall. The neighbor relationship based methods, e.g., CBP-KNN and Cos-Cos, generally perform better than the learning based approaches, e.g., BPR-Map and CMF. A possible explanation is that using social information is effective when we know nothing about the user's preference. Users have similar attributes tend to have similar tastes. Although the learning based approaches usually perform better on regular recommendation, the compared methods are not sophisticated enough in cold-start recommendation.

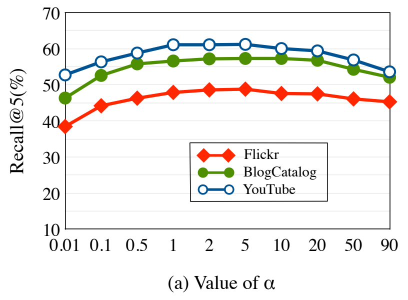
Our approach adapts the ideas of transfer learning [24]. We first learn from the side information, and then transfer the learned knowledge to the new users. The user relationship are embedded in the learned knowledge. Thus, our approach takes the advantage of neighbor relationship based approaches. On the other hand, the compared baselines commonly preserve the principal components (short-head items) and ignore the long-tail items, which can degenerate the personal recommendation. Our approach independently handles the short-head items and long-tail items, and combine them for overall recommendation.

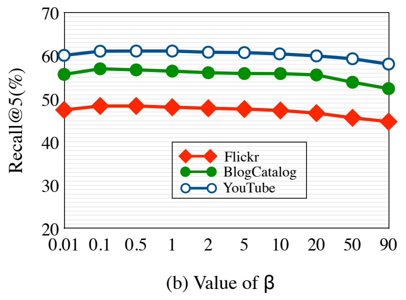
The same observations can also be drawn from the experimental results on BolgCatalog dataset reported in Fig. 4. The precision is relatively high when k is small, whist the recall is relatively high when k is large. The latter is easy to be explained for the reason that more relevant items will be included with larger k. The former is, however, not straightforward. It is relevant with the ground truth. Since the relationship matrices of both Flickr and BlogCatalog are very sparse, e.g., the sparsity of Flickr is 99.31%, and the average observations per user is around 1.4, the numerator of the precision@k tends to be fixed with the increase of the denominator.

Furthermore, we show the experimental results on two additional multimedia systems, e.g., YouTube and LastFM, in Fig. 5 and Fig. 6, respectively. Fig. 5 shows the similar observations with Fig. 4. However, the performance on YouTube is generally better than on BlogCatalog. A possible explanation is that YouTube dataset has more information than BlogCatalog dataset. Both the average observations per user and per label in YouTube dataset are larger than in BlogCatalog dataset. In other words, the side information is richer in YouTube dataset.

Fig. 6 shows different patterns with the former ones. Experimental results reported in Fig. 6 commonly have higher precision than recall. This is related with the ground truth of the dataset. Different from the other three datasets, the interested items (ground truth) of each user are much more in Hetrec11-LastFM. As a result, the recall@k is small with a small k and a big relevant number.

All in all, experiments on different real-world datasets verify the superiority of our algorithm, no matter what measurement is used.





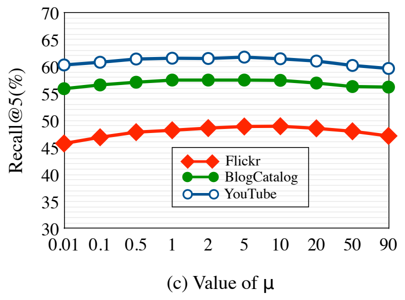


Figure 7: Parameters sensitivity of the proposed method.

Table 1: Results of mAP@100 on different dataset.

	Flickr	BlogCatalog	YouTube	LastFM
CBF-KNN	0.2805	0.3271	0.3421	0.1712
Cos-Cos	0.3142	0.4106	0.4667	0.1226
BPR-Map	0.2159	0.2822	0.3035	0.0885
CMF	0.2141	0.2776	0.2816	0.0813
LoCo	0.3357	0.4535	0.4879	0.1809
Ours	**0.3711**	**0.4962**	**0.5306**	**0.2113**

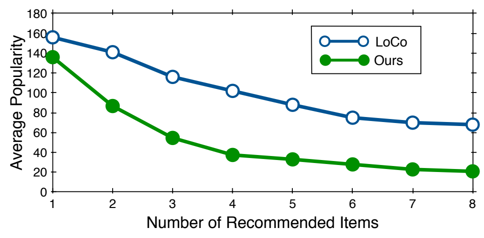


Figure 8: Average popularity of recommendations.

Our formulation, therefore, is effective for cold-start recommendation. In addition, we report the mAP@100 on different datasets in Table 1. It can be seen that our approach achieves a significant improvements against the compared methods, which further verifies the effectiveness of our formulation.

4.5 Parameters Sensitivity

To show the parameters sensitivity of our model, we report the experimental results with different values of λ and β and μ in Fig. 7(a), Fig. 7(b) and Fig. 7(c), respectively. Each of the parameters are tested from a wide range from 0 to 100.

It can be seen that our approach is not sensitive to β and μ, but the value of α should not be too small. In practice, we suggest setting α from 1 to 10, and finding the optimal value by cross-validation.

4.6 Effectiveness of Long-tail Recommendation

To evaluate the effectiveness of long-tail recommendation in the cold-start process, we build two serials of experiments. In the first serials, we set $H = 0$ to verify the contribution of long-tail recommendations w.r.t overall performance. The experimental results shows that the overall performance degenerate 2%-5% on different evaluations. In the second serials, we study the popularity of the recommended items to verify whether long-tail items are included in the final recommendation. The measurement of popularity is defined as the rating/clicking/liking frequency of items [37]. We randomly select 1,000 users from BlogCatalog and report the average popularity of recommended items in Fig. 8. It can be seen that our approach generally recommend more niche items than LoCo [31]. Since our approach considers both popular items and niche items, it tends to recommend the popular item when the number of recommendation is only 1 (If it can recommend only 1 cellphone to users, surely it would be iPhone). LoCo continues recommending popular items with increasing number of recommendations. Our approach, however, prefers recommending more niche items.

5 CONCLUSION

Both cold-start recommendation and long-tail recommendation are challenging problems in the community. To the best of our knowledge, this work is the first one which challenges both cold-start recommendation and long-tail recommendation in a unified optimization problem. Extensive experiments on four real-world datasets verify the effectiveness of the proposed method. This paper shows that one can use side information to warm-up the recommender system when there is no available historical recorders. We also find that considering long-tail items in the process of cold-start can be beneficial. In fact, our ideas of independently handling the short-head items and long-tail items can also be used in regular recommendations (relative to cold-start recommendation). It is one of the work we will study in the future.

In many real-world applications, the number of users and items can be quite large. Thus, how to reduce the time complexity of the proposed method is very important. We have mentioned in our paper, around Eq. (24), that matrix decomposition would be helpful. In our future work, we will also study how to optimize the formulation and deploy the system in a distributed environment.

6 ACKNOWLEDGEMENTS

This work was supported in part by the National Natural Science Foundation of China under Grant 61371183, ARC under Grant FT130101530 and DP170103954, the Applied Basic Research Program of Sichuan Province under Grant 2015JY0124, the China Scholarship Council and the China Postdoctoral Innovative Talent Program.

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 TKDE* 17, 6 (2005), 734–749.

[2] Chris Anderson. 2006. *The long tail: Why the future of business is selling less of more.* Hachette Books.

[3] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2011. 2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011). In *ACM RecSys.* ACM, New York, NY, USA.

[4] Wen-Yen Chen, Jon-Chyuan Chu, Junyi Luan, Hongjie Bai, Yi Wang, and Edward Y Chang. 2009. Collaborative filtering for orkut communities: discovery of user latent behavior. In *WWW.* ACM, 681–690.

[5] Zhiyong Cheng and Jialie Shen. 2014. Just-for-Me: An Adaptive Personalization System for Location-Aware Social Music Recommendation. In *International Conference on Multimedia Retrieval ICMR.* 185.

[6] Zhiyong Cheng and Jialie Shen. 2016. On Effective Location-Aware Music Recommendation. *ACM Trans. Inf. Syst.* 34, 2 (2016), 13:1–13:32.

[7] Peng Cui, Zhiyu Wang, and Zhou Su. 2014. What videos are similar with you?: Learning a common attributed representation for video recommendation. In *ACM Multimedia.* ACM, 597–606.

[8] Zhengming Ding, Ming Shao, and Yun Fu. 2015. Missing modality transfer learning via latent low-rank constraint. *IEEE transactions on Image Processing* 24, 11 (2015), 4322–4334.

[9] Marcos Aurélio Domingues, Fabien Gouyon, Alípio Mário Jorge, José Paulo Leal, João Vinagre, Luís Lemos, and Mohamed Sordo. 2013. Combining usage and content in an online recommendation system for music in the long tail. *IJMIR* 2, 1 (2013), 3–13.

[10] Zeno Gantner, Lucas Drumond, Christoph Freudenthaler, Steffen Rendle, and Lars Schmidt-Thieme. 2010. Learning attribute-to-feature mappings for cold-start recommendations. In *ICDM.* IEEE, 176–185.

[11] David Goldberg, David Nichols, Brian M Oki, and Douglas Terry. 1992. Using collaborative filtering to weave an information tapestry. *Commun. ACM* 35, 12 (1992), 61–70.

[12] Yang Hu, Xi Yi, and Larry S Davis. 2015. Collaborative fashion recommendation: a functional tensor factorization approach. In *ACM Multimedia.* ACM, 129–138.

[13] Mohsen Jamali and Martin Ester. 2010. A matrix factorization technique with trust propagation for recommendation in social networks. In *ACM RecSys.* ACM, 135–142.

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

[15] Artus Krohn-Grimberghe, Lucas Drumond, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2012. Multi-relational matrix factorization using bayesian personalized ranking for social network data. In *ACM WSDM.* ACM, 173–182.

[16] Xuan Nhat Lam, Thuc Vu, Trong Duc Le, and Anh Duc Duong. 2008. Addressing cold-start problem in recommendation systems. In *ACM IMCOM.* ACM, 208–211.

[17] Jingjing Li, Yue Wu, Jidong Zhao, and Ke Lu. 2016. Low-rank discriminant embedding for multiview learning. *IEEE transactions on cybernetics* (2016).

[18] Jingjing Li, Jidong Zhao, and Ke Lu. 2016. Joint Feature Selection and Structure Preservation for Domain Adaptation.. In *IJCAI.* 1697–1703.

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

[20] Guangcan Liu, Zhouchen Lin, Shuicheng Yan, Ju Sun, Yong Yu, and Yi Ma. 2013. Robust recovery of subspace structures by low-rank representation. *IEEE TPAMI* 35, 1 (2013), 171–184.

[21] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. 2010. Spectral regularization algorithms for learning large incomplete matrices. *JMLR* 11, Aug (2010), 2287–2322.

[22] Sean M McNee, John Riedl, and Joseph A Konstan. 2006. Being accurate is not enough: how accuracy metrics have hurt recommender systems. In *CHI'06 extended abstracts on Human factors in computing systems.* ACM, 1097–1101.

[23] Joseph Noel, Scott Sanner, Khoi-Nguyen Tran, Peter Christen, Lexing Xie, Edwin V Bonilla, Ehsan Abbasnejad, and Nicolas Della Penna. 2012. New objective functions for social collaborative filtering. In *WWW.* ACM, 859–868.

[24] Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *IEEE TKDE* 22, 10 (2010), 1345–1359.

[25] Yoon-Joo Park and Alexander Tuzhilin. 2008. The long tail of recommender systems and how to leverage it. In *ACM RecSys.* ACM, 11–18.

[26] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: an open architecture for collaborative filtering of netnews. In *CSCW.* ACM, 175–186.

[27] Francesco Ricci, Lior Rokach, and Bracha Shapira. 2011. *Introduction to recommender systems handbook.* Springer.

[28] Vala Ali Rohani, Zarinah Mohd Kasirun, Sameer Kumar, and Shahaboddin Shamshirband. 2014. An effective recommender algorithm for cold-start problem in academic social networks. *Mathematical Problems in Engineering* 2014 (2014).

[29] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *WWW.* ACM, 285–295.

[30] Andrew I Schein, Alexandrin Popescul, Lyle H Ungar, and David M Pennock. 2002. Methods and metrics for cold-start recommendations. In *ACM SIGIR.* ACM, 253–260.

[31] Suvash Sedhain, Aditya Krishna Menon, Scott Sanner, Lexing Xie, and Darius BraziunasS. 2017. Low-rank Linear Cold-Start Recommendation from Social Data. In *AAAI.*

[32] Suvash Sedhain, Scott Sanner, Darius Braziunas, Lexing Xie, and Jordan Christensen. 2014. Social collaborative filtering for cold-start recommendations. In *ACM RecSys.* ACM, 345–348.

[33] Lei Shi. 2013. Trading-off among accuracy, similarity, diversity, and long-tail: A graph-based recommendation approach. In *ACM RecSys.* ACM, 57–64.

[34] Idan Szpektor, Aristides Gionis, and Yoelle Maarek. 2011. Improving recommendation for long-tail queries via templates. In *WWW.* ACM, 47–56.

[35] Lei Tang and Huan Liu. 2009. Relational learning via latent social dimensions. In *ACM SIGKDD.* ACM, 817–826.

[36] Chang Xu, Dacheng Tao, and Chao Xu. 2016. Robust extreme multi-label learning. In *ACM SIGKDD.* 13–17.

[37] Hongzhi Yin, Bin Cui, Jing Li, Junjie Yao, and Chen Chen. 2012. Challenging the long tail recommendation. *VLDB* 5, 9 (2012), 896–907.

[38] Zi-Ke Zhang, Chuang Liu, Yi-Cheng Zhang, and Tao Zhou. 2010. Solving the cold-start problem in recommender systems with social tags. *EPL (Europhysics Letters)* 92, 2 (2010), 28002.

[39] Lei Zhu, Jialie Shen, Liang Xie, and Zhiyong Cheng. 2017. Unsupervised visual hashing with semantic assistant for content-based image retrieval. *IEEE Transactions on Knowledge and Data Engineering* 29, 2 (2017), 472–486.