Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Personalized Tag Recommendation through Nonlinear Tensor Factorization Using Gaussian Kernel

Xiaomin Fang and **Rong Pan***
Department of Computer Science
Sun Yat-sen University
Guangzhou, China
{fangxmin@mail2, panr@}.sysu.edu.cn

Guoxiang Cao, Xiuqiang He and **Wenyuan Dai**
Huawei Technologies Co., Ltd.
Bantian, Longgang District
Shenzhen, China
{benjamin.cao, hexiuqiang, dai.wenyuan}@huawei.com

Abstract

Personalized tag recommendation systems recommend a list of tags to a user when he is about to annotate an item. It exploits the individual preference and the characteristic of the items. Tensor factorization techniques have been applied to many applications, such as tag recommendation. Models based on Tucker Decomposition can achieve good performance but require a lot of computation power. On the other hand, models based on Canonical Decomposition can run in linear time and are more feasible for online recommendation. In this paper, we propose a novel method for personalized tag recommendation, which can be considered as a nonlinear extension of Canonical Decomposition. Different from linear tensor factorization, we exploit Gaussian radial basis function to increase the model's capacity. The experimental results show that our proposed method outperforms the state-of-the-art methods for tag recommendation on real datasets and perform well even with a small number of features, which verifies that our models can make better use of features.

Introduction

Nowadays, a great amount of information springs up in the Internet and users struggle to find the items they are really interested in. To tackle this problem, recommendation systems are designed to recommend appropriate items to the users according to their habits. Collaborative tagging systems like *Delicious*, *Last.fm* and *Movielens* allow users to upload resources and annotate them. The activity that users annotate items, such as movies, songs and pictures with some keywords, is called tagging. A tag can be viewed as an implicit rating and be used to identify not only the features of the items, but also the users' personality. Tag recommendation is a task to suggest tags to a user when he is about to annotate an item. Given an item, different users may use different tags to annotate it. Personalized tag recommender systems utilize users' past tagging behaviors to predict their future behaviors. For example, if two users annotated an item with the same tag before, it is likely that they will annotate another item with the same tag in the future.

Some tag recommender systems rank tags by means of tensor factorization techniques. Tensor factorization is a high-order extension of matrix factorization. Tensor factorization-based models decompose the user-item-tag tensor into three matrices to learn the latent features of the users, items and tags. Higher Order Singular Vector Decomposition (HOSVD) (Symeonidis, Nanopoulos, and Manolopoulos 2008), Multiverse (Karatzoglou et al. 2010) and RTF (Rendle et al. 2009) are based on Tucker Decomposition (TD), which can improve the state-of-the-art recommenders in prediction quality, but require too much computation power. In contrast, pairwise interaction tensor factorization (PITF) (Rendle and Schmidt-Thieme 2010) is based on another factorization approach, Canonical Decomposition (CD). It is linear both in the dataset size and the feature dimensionality. In this paper, we present a novel approach based on Canonical Decomposition and can be considered as a nonlinear generalization of CD. We use Gaussian radial basis function to capture the complex relations between the users, items and tags. We will analyze why exploiting Gaussian can make better use of the latent features and its relation to the traditional linear tensor factorization techniques. Our experimental results show that our proposed model outperforms other competitive methods and can achieve good performance with only a small number of latent features.

The contributions of our work are summarized as follows.

- We propose a novel nonlinear approach based on Canonical Decomposition, which employs Gaussian radial basis function. The time performance is comparable to PITF and runs in linear time.
- We compare our nonlinear approach to the traditional linear methods and analyze the reasons why the Gaussian-based method can make better use of the latent features.
- The experimental results demonstrate that our method outperforms other methods on real datasets and performs well even with only a small number of latent features.

Related Work

Tag Recommendation Techniques

There is a lot of previous work on tag recommendation. Work by Krestel, Fankhauser, and Nejdl proposed an approach based on Latent Dirichlet Allocation(LDA). Given a

*Corresponding author

document, the top terms composing its latent topics are recommended to the users. By this way, the terms in the same topic but showed in the other documents have the opportunity to be presented. Lipczak et al. employed probabilistic latent topic model for tag recommendation as well. They added a variable of tag to LDA, so as to link the tags to the topics of the document. Liu et al. ranked the tags for images by using random walks. However, the systems mentioned above are impersonalized. Different users may prefer different tags for the same item. Work by Guan et al. ranked tags by a graph-based ranking algorithm. It takes personalized tag recommendation as a "query and ranking" problem. The query consists of two parts, the user and the item. The tags are related to the item and the user. Besides, Durao and Dolog introduced a hybrid approach. It makes use of several factors, including tag similarity, tag popularity, tag representativeness and affinity between user and tag to provide appropriate tags.

Tensor Factorization Techniques

Tucker Decomposition and Canonical Decomposition are the high-order extension of Singular Value Decomposition (SVD). Nickel, Tresp, and Kriegel applied tensor factorization to collective entity resolution. Some studies (Dunlavy, Kolda, and Acar 2011; Ermiş, Acar, and Cemgil 2012; Spiegel et al. 2012) proposed tensor-based models to solve the problem of link prediction by incorporating time information. Moreover, tensor factorization is applied to personalized recommendation, including personalized web search (Sun et al. 2005), hyperlink recommendation (Gao et al. 2012) and personalized tag recommendation (Rendle and Schmidt-Thieme 2010; Symeonidis, Nanopoulos, and Manolopoulos 2008; 2010; Rendle et al. 2009; Chi and Zhu 2010). With respect to personalized tag recommendation, Higher Order Singular Vector Decomposition (HOSVD) (Symeonidis, Nanopoulos, and Manolopoulos 2008) and RTF (Rendle et al. 2009) are based on Tucker Decomposition (TD), while PITF (Rendle and Schmidt-Thieme 2010) is based on Canonical Decomposition (CD). HOSVD can not deal with missing values and it fills the unknown entries with 0. RTF handles the missing values by adding pairwise ranking constraint. Although TD-based methods outperform other state-of-the-art tag recommendation approaches, they require a lot of computation power. Therefore, they are not feasible for online recommendation. Compared with TD-based methods, CD-based methods have a huge advantage in running time, because they can be trained in linear time. PITF, an extension of CD, splits the ternary relation of users, items and tags into two relations, user-tag and item-tag and its quality is comparable to RTF.

Notations and Preliminaries

Tensor Factorization

A tensor is a multidimensional or multi-way array and the order of a tensor is the number of the dimensions.

Tucker Decomposition (TD) (Tucker 1966) and Canonical Decomposition (CD) (Carroll and Chang 1970; Harshman 1970) can be seen as higher-order extension of the matrix factorization, Singular Value Decomposition (SVD).

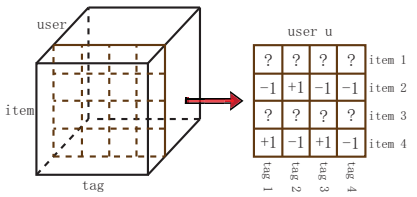

Figure 1: Tensor construction and post-based ranking interpretation for personalized tag recommendation

Tucker Decomposition (TD) decomposes a tensor into a core tensor multiplied by a matrix along each mode. Take a three-order tensor $\mathbf{A} \in \mathbb{R}^{I \times J \times K}$ for example and it can be written as

$$\mathbf{A}_{i,j,k} = \sum_{r_1=1}^{R_1} \sum_{r_2=1}^{R_2} \sum_{r_3=1}^{R_3} \mathbf{C}_{r_1,r_2,r_3} \cdot \mathbf{X}_{i,r_1} \cdot \mathbf{Y}_{j,r_2} \cdot \mathbf{Z}_{k,r_3}, \tag{1}$$

where $\mathbf{X} \in \mathbb{R}^{I \times R_1}$, $\mathbf{Y} \in \mathbb{R}^{J \times R_2}$, $\mathbf{Z} \in \mathbb{R}^{K \times R_3}$ and $\mathbf{C} \in \mathbb{R}^{R_1 \times R_2 \times R_3}$ is a core tensor. R_1, R_2 and R_3 are the number of latent features.

Canonical Decomposition (CD) decomposes a tensor into a sum of component rank-one tensors. Let $\mathbf{A} \in \mathbb{R}^{I \times J \times K}$ be a tensor and it can be written as

$$\mathbf{A}_{i,j,k} = \sum_{r=1}^{R} \mathbf{X}_{i,r} \cdot \mathbf{Y}_{j,r} \cdot \mathbf{Z}_{k,r}, \tag{2}$$

where $\mathbf{X} \in \mathbb{R}^{I \times R}$, $\mathbf{Y} \in \mathbb{R}^{J \times R}$, $\mathbf{Z} \in \mathbb{R}^{K \times R}$ and R is the number of latent features.

$\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Z}$ in Eqs. (1) and (2) are in analogy with the factor matrices in Singular Value Decomposition (SVD) and values in core tensor $\mathbf{C}$ is in analogy with eigenvalues. Besides, if we compare the running time of TD in Eq. (1) and CD in Eq. (2), we can find that CD is linear in feature dimensionality but the time complexity of TD is much higher.

Personalized Tag Recommendation

Tagging is a process that a user associates an item with its keywords, which are called tags. We use the similar notation of (Rendle et al. 2009; Rendle and Schmidt-Thieme 2010) for the formulation of personalized tag recommendation. Let U be the set of users, I be the set of items, and T be the set of tags. The process that user u annotate item i with tag t is symbolized by a triple (u, i, t). Let $S \in U \times I \times T$ denote the set of tagging history. If a triple $(u, i, t) \in S$, it means a particular user u has annotated a specific item i with tag t. The relation between the users, items and tags for personalized tag recommendation can represented by a three-order tensor, which is depicted in Figure 1.

Rendle et al. introduced post-based ranking interpretation scheme for personalized tag recommendation and we employ the schema in this paper. Let P_S represent the set of all of the user-item pairs (u, i) in S and it can be written as

$$P_S = \{(u, i) | \exists (u, i, t) \in S\}.$$

Post-based schema generates positive examples and negative examples only from the observed posts $(u, i) \in P_S$. Given a post $(u, i) \in P_S$, all of the triples $(u, i, t) \in S$ are taken as positive examples and the triples $(u, i, t) \notin S$ are taken as negative examples. If $(u, i) \notin P_S$, we do not know that user's preference for that item. Thus, all the entries for the unobserved posts are taken as missing values. We formulate the relevance of (u, i, t) with $f(u, i, t)$. Given a post (u, i), if user u has annotated item i with tag t^{pos} rather than tag t^{neg}, it implies that user u prefers tag t^{pos} to tag t^{neg} for that post. In other words, $f(u, i, t^{pos}) > f(u, i, t^{neg})$, iff triple $(u, i, t^{pos}) \in S$ and triple $(u, i, t^{neg}) \notin S$. Thus, in this paper, we use notation t^{pos} for "positive tags", and notation t^{neg} for "negative tags". As a result, the pairwise ranking constraints D_S from S is defined as

$$D_S = \{(u, i, t^{pos}, t^{neg}) | (u, i, t^{pos}) \in S \wedge (u, i, t^{neg}) \notin S\}.$$

The elements in D_S are used as training examples for tag recommendation. Figure 1 shows the tensor construction and a toy example of post-based ranking interpretation for personalized tag recommendation. User u has labeled item 4 with tag 1 and tag 3. Thus, we assume user u prefers tag 1 and tag 3 to tag 2 and tag 4. Besides, since user u has not labeled item 1 and item 4, the values of all the entries in the first row and the third row are missing.

Nonlinear Tensor Factorization using Gaussian Kernal for Tag Recommendation

Nonlinear Tensor Factorization using Gaussian

In this paper, we introduce a nonlinear way for tensor decomposition, and we refer to it as Nonlinear Tensor Factorization (NLTF). The ternary relation between users, items and tags is represented by a three-order tensor with $|U|$ users, $|I|$ items and $|T|$ tags. The tensor can be decomposed into three components, a user matrix $\mathbf{U} \in \mathbb{R}^{|U| \times R}$, an item matrix $\mathbf{I} \in \mathbb{R}^{|I| \times R}$ and a tag matrix $\mathbf{T} \in \mathbb{R}^{|T| \times R}$, where R is the number of features. The entries in the i-th row of matrix $\mathbf{U}$, matrix $\mathbf{I}$ and matrix $\mathbf{T}$ indicate the latent features of i-th user, i-th item and i-th tag, respectively. Each column of matrices $\mathbf{U}$, $\mathbf{I}$ and $\mathbf{T}$ can be considered as a latent topic.

Analogous to the previous work (Rendle and Schmidt-Thieme 2010), we model three pairwise relations user-item, user-tag and item-tag rather than directly model ternary relation use-item-tag. As we have mentioned, an entry in a feature vector stands for a latent topic. Without loss of generality, we take relation item-tag for example. We first make an assumption that the latent topics are independent to each other. Fox example, if the items are movies, the first latent topic might be comedy and the second latent topic might be action. If tag t is intimately related to item i with respect to the first topic, the value of the first latent feature of tag t should be close to the value of the first latent feature of item i. Taking consideration of it, we assume if item i and tag t are relevant with the k-th topic, then $\mathbf{T}_{t,k}$ obeys the Gaussian distribution $N(\mathbf{I}_{i,k}, \sigma^2)$ and vice versa, where $\mathbf{T}_{t,k}$ represents the k-th feature of tag t and $\mathbf{I}_{i,k}$ represents the k-th feature of item i. In other words, the distance between the value of the k-th feature of tag t and the k-th feature of item i obeys Gaussian distribution. The smaller the distance is, the more relevant are item i and tag t. It can be written as

$$||\mathbf{I}_{i,k} - \mathbf{T}_{t,k}|| \sim N(0, \sigma^2),$$

where σ is the standard deviation of the Gaussian distribution. The situations of user-tag and user-item relations are in analogy to that of the item-tag relation. However, the distributions of the latent topics of user-item, user-tag and item-tag relationships might be different. Thus, we define two sets of the features for the users, items and tags, respectively, and we have

$$\begin{aligned} ||\mathbf{U}^I_{u,k} - \mathbf{I}^U_{i,k}|| &\sim N(0, \sigma^2), \\ ||\mathbf{U}^T_{u,k} - \mathbf{T}^U_{t,k}|| &\sim N(0, \sigma^2), \\ \text{and } ||\mathbf{I}^T_{i,k} - \mathbf{T}^I_{t,k}|| &\sim N(0, \sigma^2), \end{aligned}$$

where $\mathbf{U}^T, \mathbf{U}^I \in \mathbb{R}^{|U| \times R}$, $\mathbf{I}^U, \mathbf{I}^T \in \mathbb{R}^{|I| \times R}$, $\mathbf{T}^U$, and $\mathbf{T}^I \in \mathbb{R}^{|T| \times R}$.

With the assumptions made, given a particular triple (u, i, t), we define its relevant score $f(u, i, t)$ as

$$\begin{aligned} f(u, i, t) = &\sum_k \zeta^{UI}_k \cdot exp(-\frac{1}{2}\beta||\mathbf{U}^I_{u,k} - \mathbf{I}^U_{i,k}||^2) \\ &+ \sum_k \zeta^{UT}_k \cdot exp(-\frac{1}{2}\beta||\mathbf{U}^T_{u,k} - \mathbf{T}^U_{t,k}||^2) \\ &+ \sum_k \zeta^{IT}_k \cdot exp(-\frac{1}{2}\beta||\mathbf{I}^T_{i,k} - \mathbf{T}^I_{t,k}||^2), \end{aligned} \quad (3)$$

where $\beta = 1/\sigma^2$. ζ^{UI}_k is the weight or the importance of the k-th topic for relation user-item. Similarly, ζ^{UT}_k and ζ^{IT}_k are the importance of the k-th topic for relations user-tag and item-tag, respectively.

In fact, the NLTF model can be represented by a neural network with a specific activation function and a three-layer networks structure. The framework of the neural network is depicted in Figure 2. The neural network contains three layers, input layer, hidden layer and output layer. The input layer of the neural network is composed of six blocks and the hidden layer is composed of three blocks. The first two block of the input layer and the first block of hidden layer corresponds to relation user-item. The representation of relations user-tag and item-tag are in analogy with relation user-item. For the input representation, one-of-n encoding schema is used. Given a triple (u, i, t), the u-th unit in the first block is set to be 1 and all the other units in the first block are set to be 0. The settings for the other input blocks are similar. For the hidden layer, we use $exp(-\frac{1}{2}\beta||p - q||^2)$ as the activation function, where p and q are the weights of the nonzero input units. Take relation user-item for example. p and q correspond to $\mathbf{U}^I_{u,k}$ and $\mathbf{I}^U_{i,k}$ in Eq. (3), respectively. ζ^{UI}_k in Eq. (3) corresponds to the weight of the k-th unit in the first block of the hidden layer.

For a single training example $(u, i, t^{pos}, t^{neg}) \in D_S$, we define the objective function for personalized tag recommendation with respect to that single example as follows.

$$J(u, i, t^{pos}, t^{neg}) = sigmoid(f(u, i, t^{pos}) - f(u, i, t^{neg})), \quad (4)$$

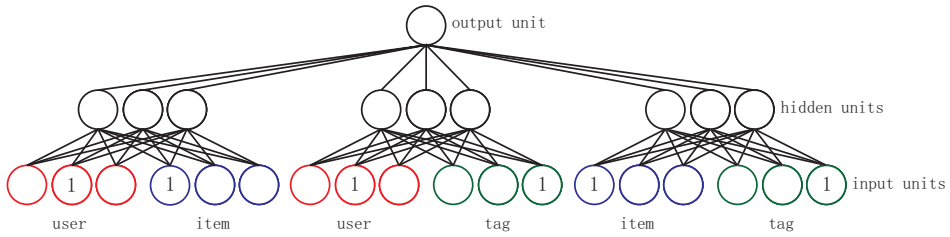

Figure 2: Neural network representation for NLTF

where $sigmoid(x) = \frac{1}{1+exp(-x)}$. It is clear that the sigmoid function, $sigmoid(x)$, is a monotonically increasing function. If a particular user u has annotated item i with tag t^{pos}, but not tag t^{neg}, it indicates that he prefers tag t^{pos} to tag t^{neg}. Thus, the difference of the relevant scores between (u, i, t^{pos}) and (u, i, t^{neg}) should be large and we use sigmoid function to model it.

Given a training set D_S, the overall objective function is defined as

$$J(D_S) = \sum_{(u,i,t^{pos},t^{neg}) \in D_S} J(u, i, t^{pos}, t^{neg}). \quad (5)$$

The parameters estimation for the tag recommendation task can be formulated as the optimization problem of maximizing $J(D_S)$.

Note that, given an example (u, i, t^{pos}, t^{neg}), the first component in $f(u, i, t)$ is eliminated when calculate $f(u, i, t^{pos}) - f(u, i, t^{neg})$ in Eq. (4). Thus, we drop it from $f(u, i, t)$. Besides, when maximizing the objective function, the scale of ζ_k^{UI}, ζ_k^{UT} and ζ_k^{IT} will get large, which leads to overfitting. To address this issue, we simply set the values of ζ_k^{UI}, ζ_k^{UT} and ζ_k^{IT} to 1. Therefore, given a triple (u, i, t), the relevant score $f(u, i, t)$ is simplified as

$$f(u,i,t) = \sum_k exp(-\frac{1}{2}\beta||\mathbf{U}_{u,k}^T - \mathbf{T}_{t,k}^U||^2) + \sum_k exp(-\frac{1}{2}\beta||\mathbf{I}_{i,k}^T - \mathbf{T}_{t,k}^I||^2). \quad (6)$$

Parameters Learning

We use stochastic gradient descent, a widely used technique, to optimize the objective function $J(D_S)$. We randomly select a training example (u, i, t^{pos}, t^{neg}) from D_S and perform an update for that example. This process repeats until convergence. The gradients for the nonlinear tensor factorization model are

$$\frac{\partial f_{u,i,t}}{\partial \mathbf{U}_{u,k}^T} = -\beta \cdot exp(-\frac{1}{2}\beta||\mathbf{U}_{u,k}^T - \mathbf{T}_{t,k}^I||^2)(\mathbf{U}_{u,k}^T - \mathbf{T}_{t,k}^U),$$

$$\frac{\partial f_{u,i,t}}{\partial \mathbf{I}_{i,k}^T} = -\beta \cdot exp(-\frac{1}{2}\beta||\mathbf{I}_{i,k}^T - \mathbf{T}_{t,k}^I||^2)(\mathbf{I}_{i,k}^T - \mathbf{T}_{t,k}^I),$$

$$\frac{\partial f_{u,i,t}}{\partial \mathbf{T}_{t,k}^U} = \beta \cdot exp(-\frac{1}{2}\beta||\mathbf{U}_{u,k}^T - \mathbf{T}_{t,k}^I||^2)(\mathbf{U}_{u,k}^T - \mathbf{T}_{t,k}^U),$$

and $$\frac{\partial f_{u,i,t}}{\partial \mathbf{T}_{t,k}^I} = \beta \cdot exp(-\frac{1}{2}\beta||\mathbf{I}_{i,k}^T - \mathbf{T}_{t,k}^I||^2)(\mathbf{I}_{i,k}^T - \mathbf{T}_{t,k}^I).$$

With the gradients, we can update the parameters of our model. We first initialize all of the parameters and then learn the parameters until convergence.

Comparison with Other Tensor Factorization Models

HOSVD (Karatzoglou et al. 2010), RTF (Rendle et al. 2009) and PITF (Rendle and Schmidt-Thieme 2010) are models based on linear tensor factorization. In particular, HOSVD and RTF are TD-based and PITF is CD-based. Our proposed model NLTF can be seen as a nonlinear extension of the CD-based models. We use Gaussian radial basis function rather than employing dot product used in PITF.

Let's assume the number of features of users, items and tags are the same for TD-based models, which is R. If we exploit stochastic gradient descent (SGD) to learn the parameters for all models, given a training example, TD-based models take $O(R^3)$ to update the parameters, while CD-based models take $O(R)$, where R is the number of features. For prediction, given a post $(user, item)$, TD-based models take $O(|T| \cdot R + R^3)$ to rank the tags, while CD-based models take $O(|T| \cdot R)$. Therefore, CD-based models outperform TD-based models in terms of the running time and are more feasible for online tag recommendation.

On the other hand, our model that exploits Gaussian radial basis function to calculate the similarity is better than models exploiting dot product. Models employing dot product can be viewed as two-class classifiers, while models employing Gaussian radial basis function can be seen as multi-class classifiers. For example, we want to provide a set of tags to user u who has just watched movie i. Let's assume there is only one latent feature $\mathbf{I_{i,1}}$. For dot product, if the value of $\mathbf{I}_{i,1}$ is positive, it may indicate the movie is a comedy; otherwise, the movie is not. Thus, if the value of $\mathbf{I}_{i,1}$ is positive and tag t is related to comedies, the value of $\mathbf{T}_{t,1}$ should be positive so as to make the dot product of $\mathbf{I}_{i,1}$ and $\mathbf{T}_{t,1}$ large, so that tag t can rank high given post (u, i). For Gaussian-based models, a toy example is depicted in Figure 3. The horizontal axis represents the value of $\mathbf{I_{i,1}}$.If $\mathbf{I}_{i,1} \in (-3, 3)$, the movie is probably an action; if $\mathbf{I}_{i,1} \in (-9, -5)$, the movie is probably an romantic; if $\mathbf{I}_{i,1} \in (5, 9)$, the movie is probably an drama; if $\mathbf{I}_{i,1} \in (-\infty, 20) \bigcup (20, \infty)$, it does not belong to any of the categories in Figure 3. For instance, tag t is "comedy" and movie i_1 and movie i_2 are annotated with tag t. As we assume, if an item and a tag are related, the distance between the feature of that item and the feature of that tag obeys

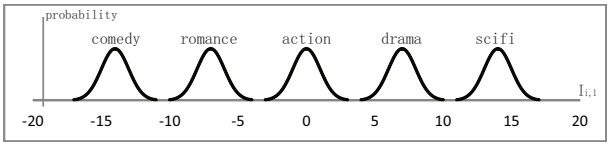

Figure 3: A toy example of nonlinear tensor factorization using Gaussian radial basis function

Gaussian distribution. That means the distance between the feature of tag t and the feature of movie i_1 is likely to be small and the distance between the feature of tag t and the feature of movie i_2 is likely to be small as well. Thus, the feature of i_1 is likely to be close to the feature of i_2. In other words, the features of the movies that belong to the same categories are likely to be close to each other and obey Gaussian distribution. We can distinguish several categories by using only one feature, because different categories fall into different part of the coordinate axis. In conclusion, Gaussian-based models can deal with more than two classes with only one feature, but dot product can handle only two classes. Thus, exploiting Gaussian function on radial basis distance can make better use of the latent features than dot product. The experimental results on real datasets in the next section demonstrate this point.

Experimental Results

Experimental Setup

We use four datasets to evaluate the performance of our proposed model, including *Delicoius-small*[1], *Last.fm*[2], *Movielens*[3] and *Delicious-large*[4] (Wetzker, Zimmermann, and Bauckhage 2008). Three smaller datasets *Delicious-small*, *Last.fm* and *Movielens* come from the 2-nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011). The larger dataset *Delicious-large* (Wetzker, Zimmermann, and Bauckhage 2008) contains the complete bookmarking activity for almost 2 million users from the launch of the social bookmarking website in 2003 to the end of March 2011. For each dataset, we removed the infrequent users, items occurred in less than 5 posts and tags occurred in less than 5 triplets. The statistics of datasets after removal are described in Table 2. To avoid randomness of the results, we perform 10-fold cross-validation for the three smaller datasets, *Delicious-small*, *Last.fm* and *Movielens*. For larger dataset *Delicious-large*, we randomly sample about 40 thousand posts for test and the remaining data is used for training.

To evaluate our method, we introduce two baseline functions $N_{UT}(u,t)$ and $N_{IT}(i,t)$. Given user u and tag t, we define $N_{UT}(u,t)$ as

$$N_{UT}(u,t) = |\{i|(u,i,t) \in S\}|.$$

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

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

[3]http://www.grouplens.org

[4]http://www.zubiaga.org/resources/socialbm0311

Table 2: Data statistics

Dataset	Users	Items	Tags	Posts	Triples
Delicious-small	1.8K	35.9K	9.3K	66.7K	312.0K
Last.fm	1.6K	7.3K	2.3K	61.1K	164.5K
Movielens	0.7K	2.6K	1.6K	17.8K	30.8K
Delicious-large	1.5M	7.5M	2.6M	198.9M	675.0M

Besides, given item i and tag t, we define $N_{IT}(i,t)$ as

$$N_{IT}(i,t) = |\{u|(u,i,t) \in S\}|.$$

We compare our proposed model NLTF with several baselines, including PITF, SVD, Top-UT, Top-IT and Top-UT+IT, where SVD only employs two-dimension relation item-tag, Top-UT ranks tags based on $N_{UT}(u,t)$, Top-IT ranks tags based on $N_{IT}(i,t)$ and Top-UT+IT ranks tags based on $N_{UT}(u,t) + N_{IT}(i,t)$. Here, Top-UT represents the affinity between users and tags, Top-IT represents the tag popularity and Top-UT+IT can be seen as a simple ensemble of Top-UT and Top-IT. We do not compare NLTF with TD-based models like HOSVD and RTF, because the computation time is unacceptable in practice, especially for larger dataset *Delicious-large*. Besides, the experimental results in (Rendle and Schmidt-Thieme 2010) show that the performance of PITF is comparable to TD-based models. For each dataset. we tuned the hyper-parameters of all models by grid search to achieve the best results.

Evaluation Methodology

We treat personalized tag recommendation as a information retrieval problem and employ Normalized Discounted Cumulative Gain (NDCG) as our metrics to evaluate the performance of the models. NDCG is widely used in information retrieval and it makes the assumption that relevant tags are more valuable if it appears earlier in the recommendation list.

Discounted Cumulative Gain (DCG) evaluates the gain of a tag based on its position in the ranked sequence returned by the model. The DCG accumulated at a particular position p is defined as

$$DCG@p = rel(1) + \sum_{k=1}^{p} \frac{rel(k)}{\log_2(k)}, \tag{7}$$

where $rel(k)$ is 1 if the tag at position k is relevant or 0 otherwise. In order to normalize the metric, Ideal Discounted Cumulative Gain (IDCG) is introduced, which is the DCG of the ideal ranked sequence. Normalized Discounted Cumulative Gain (NDCG) is defined as

$$NDCG@p = \frac{DCG@p}{IDCG@p}. \tag{8}$$

Performance on Smaller Datasets

Comparisons with Baselines Table 3 shows the performance of different methods on three smaller datasets *Delicious-small*, *Last.fm* and *Movielens*. For NLTF and PITF, the number of features $R = 64$, and for SVD, $R =$

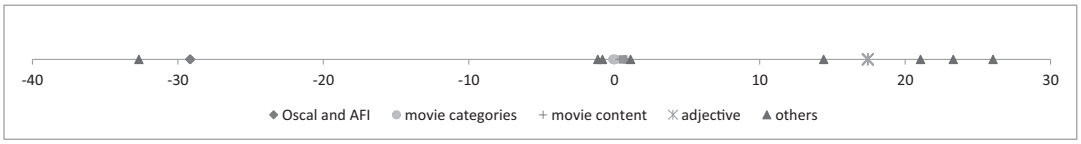

Figure 4: Tag distribution on dataset *Movielens* with the number of features $R = 1$

Table 1: Description of tags on dataset *Movielens* with the number of features $R = 1$

Oscar and AFI	Movie Categories	Movie Content	Adjective	Others
afi 100	scifi	war	atmospheric	erlends dvds
oscar (best directing)	fantasy	world war ii	disturbing	my movies
oscar (best cinematography)	comedy	surreal	deliberate	johnny depp
oscar (best supporting actor)	action	serial killer	quirky	boring
afi 100 (thrills)	anime	time travel	reflective	based on a book
oscar (best actor)	classic	true story	visceral	franchise

256 because NLTF and PITF saturate at 64 features and SVD saturates at 256 features. It can be seen from Table 3 that NLTF outperforms other competitive methods on *Delicious-small* and *Last.fm*, but PITF performs a little bit better than our method on *Movielens*. Since the data in *Delicious-small* and *Last.fm* is relatively sparser than the data in *Movielens*, it should be the case that our method has some advantages when the data is sparser. Moreover, our method beats Top-UT+IT on all the datasets, which integrates tag popularity and affinity between users and tags, but PITF works worse than Top-UT-IT on *Delicious-small*.

Table 3: NDCG@5 of different methods on three smaller datasets

	Delicious-small	Last.fm	Movielens
PITF-64	0.2518	0.6715	0.6466
NLTF-64	0.2718	0.6986	0.6309
SVD-256	0.1089	0.3633	0.1673
Top-UT	0.2416	0.3811	0.4082
Top-IT	0.1110	0.4683	0.2851
Top-UT+IT	0.2607	0.4588	0.4909

Impact of Number of Features The performances of the NLTF and PITF depend on the number of features. Table 4 shows the impact of the number of features on the three datasets *Delicious-small*, *Last.fm* and *Movielens*. As expected, the performances of NLTF and PLTF increase when we increase the number of features. Their performances saturate at 64 features on all three datasets. Besides, the performance of NLTF is much better than PITF when the number of feature is small, because models exploiting Gaussian like NLTF can make better use of the latent features than models exploiting dot product like PITF, just as we analyze in Section and the effect is more significant when only a smaller number of features are used.

Besides, in order to verify the point that using Gaussian can make better use of features than using dot product, we carry out another experiment. We train our proposed model NLTF by using only one feature on *Movielens*, so as to observe whether the tag distribution is corresponding to the toy example in Figure 3. In order to get better observation, the unpopular tags are removed. The tag distribution on dataset *Movielens* with the number of features $R = 1$ is depicted in Figure 4 and the description of which is showed in Table 1. As we can see, NLTF divides the tags into four groups automatically, including *Oscar and AFI*, *movie categories*, *movie content* and *adjective*. Tags in the same group are close to each other and some of their labels overlapped. Tags in *others* do not belong to any groups, thus they might do little help to personalized tag recommendation.

Table 4: NDCG@5 for different number of features on three smaller datasets

	feature	4	8	16	32	64
Delicious -small	PITF	0.068	0.100	0.157	0.231	0.252
	NLTF	0.075	0.126	0.188	0.247	0.272
Last.fm	PITF	0.369	0.502	0.601	0.654	0.672
	NLTF	0.445	0.562	0.652	0.692	0.699
Movielens	PITF	0.321	0.468	0.576	0.629	0.647
	NLTF	0.479	0.564	0.606	0.624	0.631

Performance on Large-Scale Dataset

In addition, we compare our method NLTF with PITF on a large-scale dataset *Delicious-large*, which contains more than 0.6 billion triples and about 0.2 billion posts. There is no previous work has conducted experiments for personalized tag recommendation on such a large-scale dataset before to our best knowledge. We conduct experiments on a computer with a 6-core intel i7 CPU and the performance after 30 iterations is showed in Table 5. From the figure, we can see that our method greatly outperforms PITF even on large scale dataset.

Table 5: Comparison of NLTF and PITF on larger dataset

	NDCG@1	NDCG@3	NDCG@5	NDCG@10
NLTF	0.3143	0.3053	0.3214	0.3579
PITF	0.2845	0.2758	0.2907	0.3247

Conclusion and Future Work

We introduce a nonlinear tensor factorization method based on Canonical Decomposition in this paper. Our proposed method exploits Gaussian radial basis function and can make better use of features than dot product. The models exploiting Gaussian can be considered as a multi-class classifiers, while the models exploiting dot product can be considered as a two-class classifiers. Additionally, our experimental results show that our method outperforms the baseline methods and can achieve very good performance with only a small number of features.

For future work, we plan to leverage the profile of users, the content of the items and tags to make better predictions for tag recommendation.

Acknowledgements

We would like to thank the many referees of the previous version of this paper for their extremely useful suggestions and comments. This work was supported by Huawei Innovation Research Program (HIRP) and National Science Foundation of China (61033010).

References

Carroll, J., and Chang, J. 1970. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika* 35(3):283–319.

Chi, Y., and Zhu, S. 2010. Facetcube: a framework of incorporating prior knowledge into non-negative tensor factorization. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, 569–578. ACM.

Dunlavy, D. M.; Kolda, T. G.; and Acar, E. 2011. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 5(2):10.

Durao, F., and Dolog, P. 2010. Extending a hybrid tag-based recommender system with personalization. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, 1723–1727. ACM.

Ermiş, B.; Acar, E.; and Cemgil, A. T. 2012. Link prediction via generalized coupled tensor factorisation. *arXiv preprint arXiv:1208.6231*.

Gao, D.; Zhang, R.; Li, W.; and Hou, Y. 2012. Twitter hyperlink recommendation with user-tweet-hyperlink three-way clustering. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, 2535–2538. ACM.

Guan, Z.; Bu, J.; Mei, Q.; Chen, C.; and Wang, C. 2009. Personalized tag recommendation using graph-based ranking on multi-type interrelated objects. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, 540–547. ACM.

Harshman, R. A. 1970. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis. *UCLA working papers in phonetics* 16:1.

Karatzoglou, A.; Amatriain, X.; Baltrunas, L.; and Oliver, N. 2010. Multiverse recommendation: n-dimensional tensor factorization for context-aware collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, 79–86. ACM.

Krestel, R.; Fankhauser, P.; and Nejdl, W. 2009. Latent dirichlet allocation for tag recommendation. In *Proceedings of the third ACM conference on Recommender systems*, 61–68. ACM.

Lipczak, M.; Hu, Y.; Kollet, Y.; and Milios, E. 2009. Tag sources for recommendation in collaborative tagging systems. *ECML PKDD discovery challenge* 157–172.

Liu, D.; Hua, X.-S.; Yang, L.; Wang, M.; and Zhang, H.-J. 2009. Tag ranking. In *Proceedings of the 18th international conference on World wide web*, 351–360. ACM.

Nickel, M.; Tresp, V.; and Kriegel, H.-P. 2011. A three-way model for collective learning on multi-relational data. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, 809–816.

Rendle, S., and Schmidt-Thieme, L. 2010. Pairwise interaction tensor factorization for personalized tag recommendation. In *Proceedings of the third ACM international conference on Web search and data mining*, 81–90. ACM.

Rendle, S.; Balby Marinho, L.; Nanopoulos, A.; and Schmidt-Thieme, L. 2009. Learning optimal ranking with tensor factorization for tag recommendation. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 727–736. ACM.

Spiegel, S.; Clausen, J.; Albayrak, S.; and Kunegis, J. 2012. Link prediction on evolving data using tensor factorization. In *New Frontiers in Applied Data Mining*. Springer. 100–110.

Sun, J.-T.; Zeng, H.-J.; Liu, H.; Lu, Y.; and Chen, Z. 2005. Cubesvd: a novel approach to personalized web search. In *Proceedings of the 14th international conference on World Wide Web*, 382–390. ACM.

Symeonidis, P.; Nanopoulos, A.; and Manolopoulos, Y. 2008. Tag recommendations based on tensor dimensionality reduction. In *Proceedings of the 2008 ACM conference on Recommender systems*, 43–50. ACM.

Symeonidis, P.; Nanopoulos, A.; and Manolopoulos, Y. 2010. A unified framework for providing recommendations in social tagging systems based on ternary semantic analysis. *Knowledge and Data Engineering, IEEE Transactions on* 22(2):179–192.

Tucker, L. R. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika* 31:279–311.

Wetzker, R.; Zimmermann, C.; and Bauckhage, C. 2008. Analyzing social bookmarking systems: A del.icio.us cookbook. In *Mining Social Data (MSoDa) Workshop Proceedings*, 26–30.