The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Personalized Time-Aware Tag Recommendation

Keqiang Wang,[1] Yuanyuan Jin,[1] Haofen Wang,[2] Hongwei Peng,[1] Xiaoling Wang[1]*
[1]International Research Center of Trustworthy Software
Shanghai Key Laboratory of Trustworthy Computing
East China Normal University, Shanghai, China
[2] Shenzhen Gowild Robotics Co. Ltd
sei.wkq2008@gmail.com, sei_12jyy@126.com, wang_haofen@gowild.cn
penghongwei_phw@163.com, xlwang@sei.ecnu.edu.cn

Abstract

Personalized tag recommender systems suggest a list of tags to a user when he or she wants to annotate an item. They utilize users' preferences and the features of items. Tensor factorization techniques have been widely used in tag recommendation. Given the user-item pair, although the classic PITF (Pairwise Interaction Tensor Factorization) explicitly models the pairwise interactions among users, items and tags, it overlooks users' short-term interests and suffers from data sparsity. On the other hand, given the user-item-time triple, time-aware approaches like BLL (Base-Level Learning) utilize the time effect to capture the temporal dynamics and the most popular tags on items to handle cold start situation of new users. However, it works only on individual level and the target resource level, which can not find users' potential interests. In this paper, we propose an unified tag recommendation approach by considering both time awareness and personalization aspects, which extends PITF by adding weights to user-tag interaction and item-tag interaction respectively. Compared to PITF, our proposed model can depict temporal factor by temporal weights and relieve data sparsity problem by referencing the most popular tags on items. Further, our model brings collaborative filtering (CF) to time-aware models, which can mine information from global data and help improving the ability of recommending new tags. Different from the power-form functions used in the existing time-aware recommendation models, we use the Hawkes process with the exponential intensity function to improve the model's efficiency. The experimental results show that our proposed model outperforms the state of the art tag recommendation methods in accuracy and has better ability to recommend new tags.

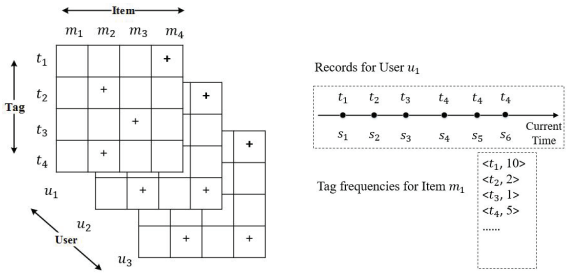
Introduction

Nowadays, there is a great amount of information emerging in the internet every day and it's difficult for users to find items that are really appealing to them. Recommender system is an effective tool to overcome information overload. It can model users' interests by analyzing users' behavior and offer users the information that meets their preferences. Tag systems like Delicious, Last.fm and Movielens allow users to annotate the web pages, songs and movies using key words, which is called tagging. Tags can make managing and searching web items easier. A tag can be regarded as a kind of implicit rating, which can identify not only features of items, but also the users' personalities. Tag recommendation is to provide the target user tags when he wants to annotate an item. Personalized tag recommendation will analyze users' past tagging behaviors to predict the tags being used in the future, which depends on both users and items. For instance, if two users have used the same tags to mark the same items, it's likely for them to use the same tag for another item. On the other hand, if a certain user recently uses a tag very frequently, there is a high probability for him to reuse the same tag. It has also been proved that the personalized approaches outperform the theoretically best unpersonalized method (Rendle et al. 2009a). Hence, in this paper, we only focus on the personalized tag recommendation.

Some systems utilize tensor factorization techniques to rank the candidate tags. Tensor factorization based models decompose the user-item-tag tensor into feature matrices to represent the latent features of users, items and tags respectively. RTF (Rendle et al. 2009a) is based on Tucker Decomposition, which is cubic in feature dimensionality and thus unfeasible for mid-sized and large data sets with high factorization dimension. To tackle this problem, Rendle et al. proposed the Canonical Decomposition based PITF (Rendle and Schmidt-Thieme 2010) model. As the example in figure 1(a) shows, given u_1 and m_1, PITF utilizes the global user-item-tag tensor to recommend tags, which is linear in both the data set size and feature dimensionality. PITF can process high dimensional factorization well and recommend new tags. It also won the ECML/PKDD Discovery Challenge 2009 (Rendle and Schmidt-Thieme 2009) for graph-based tag recommendation.

Although the classic tensor factorization methods can find the potential interests of users, they can not depict the temporal dynamics in users' tagging behaviour. Some studies (Yin et al. 2011; Charlin et al. 2015; Koren 2010) have demonstrated that users' behaviors will change with time, which indicates that users have their short-term interests and their recent tagging behaviors will influence the next tagging. Therefore, some time and frequency based BLL

*Corresponding author. This work was supported by National Key R&D Program of China (No. 2017YFC0803700), NSFC grants (No. 61532021 and 61472141), Shanghai Knowledge Service Platform Project (No. ZF1213), and Shanghai Agriculture Applied Technology Development Program (No. G20160201).



(a) User-Item-Tag Tensor (b) Records of u_1 and m_1

Figure 1: Data Examples

(Base-Level Learning)-like studies are proposed , like GIPR (Zhang, Tang, and Zhang 2012), GIRPTM (Zhang, Tang, and Zhang 2012), BLL (Anderson et al. 2004) and BLL_{ac} (Kowald et al. 2015b). These methods utilize the recency, which is the time elapsed since the past tagging behavior, and the frequency tags are used to predict the probability that users reuse a certain tag. Further, they also reference the most popular tags on target resources to handle cold start of new user. For example, in figure 1(b), every black spot on the time line represents that user u_1 used a tag t at time s. Given user u_1, item m_1 and the current time, considering records of user u_1, BLL-like models may choose t_4 because it is used 3 times recently. In addition, referring to tag frequencies on item m_1, t_1 is also suitable to be recommended because the frequency it is used on item m_1 is the biggest. However, this kind of approaches can only recommend tags used by the target user or tags assigned by other users to the target item, which lack the ability to recommend new tags.

For item recommendation, the existing studies have integrated the temporal factor into personalized methods. However, item recommendation is different from tag recommendation. In item recommendation, only the interaction between users and items needs to be modeled. In tag recommendation, there are three entities including users, items and tags, and more interactions need to be considered. Hence, item recommendation models cannot be directly used to do tag recommendation and tag recommendation is more complex. For tag recommendation models, there are two ways to combine the temporal factor with personalized methods: integrating time factor into personalized model or making the time-aware tag recommendation personalized. Integrating temporal factor into personalized tag recommendation methods can help depict temporal dynamics and relieve data sparsity problem. On the other hand, introducing CF to time-dependent tag recommendation models can improve the models's ability to capture users' potential interests.

However, directly merging the results of both models may be ineffective. In this paper, we propose an unified model considering both personalization and temporal factor: Time-Aware PITF (TAPITF) model. It extends the PITF model by incorporating the user-tag-time weights and the item-tag weights into PITF. Given user u_1 and item m_1, TAPITF can not only mine the user-item-tag tensor to find preference and potential interests of u_1, but also adjust the preference weights according to the time interval between u_1's past behavior and current time as well as the popularity of tags on item m_1. Therefore, the proposed model can depict the phenomenon that users' tagging behaviors change with time and exploit the similar users, similar items and similar tags effectively to improve the accuracy and novelty of tag recommendation. Generally, the main contributions of this work are summarized as follows:

- To the best of our knowledge, we are the first to propose an unified tag recommendation approach by considering both time awareness and personalization aspects. Specifically, we extend the classic tag recommendation model PITF by adding user-item-time weights as well as itemf-tag weights.
- We explicitly model temporal information in users' tagging behaviors by the Hawkes process to improve the model efficiency. Specifically, we substitute exponential function for the power function widely used in existing BLL-like work to calculate the user-item-time weight. The preference value of a user on a certain tag at the current time can be computed in a recursion form and the computation time can be greatly reduced accordingly.
- We conduct comparative experiments on some real data sets. The experimental results show that our proposed method can outperform the state-of-art methods in terms of accuracy and has better ability to recommend new tags.

Related Work

Personalized Tag Recommendation The simplest personalized tag recommendation methods are frequency-based. For example, MP_u (Jäschke et al. 2007) recommends a user the tag he used the most times; $MP_{u,m}$ (Jäschke et al. 2007) combines MP_u and MP_m linearly to consider both user himself's tagging behavior and popular tags on items. Marinho et al. handle tag recommendation in the collaborative filtering way (Marinho and Schmidt-Thieme 2008). They calculate the similarity between users to recommend tags. Similar to PageRank algorithm used in search engine, Hotho et al. propose Adaptive Page Rank (APR) algorithm. The main idea is that if an item has been annotated by an important user using an important tag, this item can also be regarded as an important item (Hotho et al. 2006). Jäschke et al. extend APR to FolkRank (FR) model (Jäschke et al. 2007), which exploits both the PageRank method and similarities between users. This model outperforms the frequency and collaborative filtering based methods. In addition, tensor factorization technique is also widely applied to recommend tags. Symeonidis et al. (Symeonidis, Nanopoulos, and Manolopoulos 2008) utilize the High Order Singular Vector Decomposition (HOSVD) (De Lathauwer, De Moor, and Vandewalle 2000) technique. They transform the user-item-tag tensor into three matrices and SVD is used to train latent features of users, items and tags. Because there are three matrices to be factorized, this model costs more training time. To reduce the number of matrices to be factorized, Tucker Decomposition (Tucker 1966) based RTF (Rendle et al. 2009a) model directly factorizes the user-item-tag tensor in three dimension by maximizing AUC. However, the

time complexity of tag prediction in RTF is cubic in feature dimensionality. To improve the computing efficiency, Rendle et al. propose PITF model (Rendle and Schmidt-Thieme 2010). It explicitly models the pairwise interactions among users, items and tags and the time complexity of tag prediction is linear in the feature dimension.

The above models, except the frequency-based methods, suffer from the big computation cost. Moreover, they all ignore that users' tagging behaviors will change with time. Recently, there are some studies incorporating time factor into tag recommendation. Yin et al. verify the existence of short-term user interests in social tagging systems through experiments (Yin et al. 2011). Zhang et al. propose the frequency and time information based GIPR model (Zhang, Tang, and Zhang 2012). It applies exponential distribution to model the first used time and last used time of a certain tag. GIRPTM (Zhang, Tang, and Zhang 2012) extends GIPR by considering the most popular tags on the target items, which helps dealing with cold start situation for new users. Besides, there are some time-dependent models based on cognitive science, including BLL_{ac} (Kowald et al. 2015b), $BLL + MP_m$ (Kowald et al. 2014) and $BLL_{ac} + MP_m$ (Kowald et al. 2015a). The core idea is that tags most frequently and recently used by users are favored for recommendation. Specifically, BLL_{ac} extends BLL by adding association component to capture the features of items. However, It can only suggest tags used by the user himself. Further, $BLL + MP_m$ and $BLL_{ac} + MP_m$ extend BLL and BLL_{ac} respectively, which both consider the most popular tags on the target item.

For the time information modeling, the main difference between this paper and the BLL-like models is that, BLL-like models employ the power function to calculate the influence value of users' past tagging behaviors. If the current time changes, the impact value needs to be recalculated for each history record. While in this paper, we explicitly use the Hawkes process in exponential function to model time impact. In our model, the accumulation operation can be replaced by the recursion form. It results in that a user's preference value to tags at current time is only related to the last tagging time and computation time is saved greatly.

Content-based Tag Recommendation In recent years, as the fast development of deep learning, some researches try to recommend tags for items with content information, like images or texts. The convolutional neural network is employed to mine the latent information of images and tags are recommended to images by solving the multi-classification problem (Nguyen et al. 2017; Rawat and Kankanhalli 2016). Wang et al. utilize the SDAE model to capture items' content features and model context information to recommend tags (Wang, Shi, and Yeung 2015). Further, several studies (Li et al. 2016; Gong and Zhang 2016) use LSTM and CNN respectively to recommend hash tags for micro blogs. In this paper, we only consider the interactive information among users, items and tags, and do not incorporate the content information of items. Therefore, we only focus on the personalized collaborative filtering methods.

Notations and Preliminaries

Problem Statement The task of tag recommendation is offering a tag list for users when they want to annotate an item. Personalized tag recommendation will recommend tags according to users' tagging behaviors. For example, system may use tags that the target user has used to mark other items or utilize tags other users have used to mark the target items. Formally, personalized tag recommendation is to generate the personalized tag list $T(u,m) \subseteq T$ that user $u \in U$ can use to annotate items $m \in M$, given all users' past tagging records R. In this paper, we use U to present the user set. M is for item set, T is for tag set and $R \subseteq U \times M \times T$ represents users' tagging records. (u,m) is the user-items pair and $P_R\{(u,m)|\exists t \in T : (u,m,t) \in R\}$ is the observed user-item pair.

PITF(Pairwise Interaction Tensor Factorization) Given the user-item pair (u,m), PITF defines tag recommendation as a rank task to generate the total rank $>_{u,m} \subset T \times T$ on tag set T. Specifically, PITF designs the rating function $\hat{Y} : U \times M \times T \to \mathbb{R}$, where $\hat{Y}$ is a three dimensional tensor. Prediction rating of the $< u,m,t >$ triple is $\hat{y}_{umt}$, which indicates the probability user u uses tag t to mark item m. After ranking tags according to their ratings, we employ the top n tags as the recommendation list as follows.

$$Top(u,m,n) = argmax^n_{t \in T} \hat{y}_{u,m,t} \tag{1}$$

To get $\hat{y}_{u,m,t}$, PITF uses Canonical Decompositon to explicitly model the pairwise interactions among users, items and tags. Given user-item pair $< u,m >$, interaction between then has no impact on the final tag recommendation list. The model equation can be simplified as

$$\hat{y}_{u,m,t} = \sum_{k=1}^{K} \hat{u}_{u,k} \cdot \hat{t}^U_{t,k} + \sum_{k=1}^{K} \hat{m}_{m,k} \cdot \hat{t}^M_{t,k} \tag{2}$$

where $\hat{u}_u$ is the latent vector of user u, $\hat{t}^U_t$ is the latent vector of tag t in the user-tag relation. $\hat{t}^m_t$ is the latent vector of tag t in the item-tag relation and $\hat{m}_m$ is the latent vector of item m. The Bayesian Personalized Ranking (Rendle et al. 2009b) is used to estimate parameters. Training data D_R with the pairwise constraint is

$$D_R := \{(u,m,t_A,t_B) : (u,m,t_A) \in R \wedge (u,m,t_B) \notin R\} \tag{3}$$

According to the Maximum A Posteriori (MAP), the optimization function of PITF is written as

$$\begin{aligned} BPR - Opt :&= ln \prod_{u,m.t_A,t_B} \sigma(\hat{y}_{u,m,t_A,t_B}) p(\Theta) \\ &= \sum_{(u,m,t_A,t_B) \in D_R} ln\sigma(\hat{y}_{u,m,t_A,t_B}) - \lambda_\Theta ||\Theta||^2_F \end{aligned} \tag{4}$$

where $\hat{y}_{u,m,t_A,t_B} = \hat{y}_{u,m,t_A} - \hat{y}_{u,m,t_B}$. $\sigma(x)$ is the sigmoid function $\frac{1}{1+e^{-x}}$, $p(\Theta)$ represents the priori distribution of parameters, and λ_Θ is the regularization parameter. PITF is essentially a pairwise ranking model, which can be optimized by stochastic gradient descent algorithm with negative sampling.

BLL + $\mathbf{MP_m}$(Base-Level Learning with $\mathbf{MP_m}$) BLL + $\mathbf{MP_m}$ (Kowald et al. 2014) exploits the frequency and the recency for tag recommendation. Meanwhile, it also takes the impact of the most popular tags on the target items into consideration.

Firstly, according to the record $R_{u,t}$ of user u using tag t, If $i = 1...n$ index all tag assignments in $R_{u,t}$, the recency of a particular tag assignment is given by $s_{ref} - s_i$, where s_i is the time when user u used tag t for the ith time and s_{ref} is the current time. Finally, the BLA of tag t for a user u is given by the BLL equation:

$$BLA(t,u) = ln(\sum_{i=1}^{n}(s_{ref} - s_i)^{-d}) \tag{5}$$

In order to map the values into $[0-1]$, the normalization is applied as follows

$$||BLA(t,u)|| = \frac{exp(BLA(t,u))}{\sum_{t'=1}^{m} exp(BLA(t',u))} \tag{6}$$

where m is the total number of tags user u has used.

Furthermore, this model takes into account the most popular tags on target item m, which is MP_m. Given user-item pair (u,m), prediction score of tag t is defined as

$$\hat{y}_{u,m,t} = \beta||BLA(t,u)|| + (1-\beta)|||Y_{t,m}||| \tag{7}$$

where β is the weight parameter. Particularly, when $\beta = 0$, the model is the same as MP_m. When $\beta = 1$, the model is the basic BLL model. $|Y_{t,m}|$ is the frequency tag t is used to annotate item m by all users.

The Proposed Model - TAPITF

Time Information Modelling Temporal point process (Schoenberg 2010) is a kind of random process that depicts the discrete events in time series. Recently, temporal point process is widely used in sequence related scenarios, such as citation counts prediction (Xiao et al. 2016), contagious merger and acquisition (Yan et al. 2016), social system evolution (EmBree and Handcock 2016), conversion prediction of online advertising (Xu, Duan, and Whinston 2014) and so on. In users' past tagging history, each point represents that the user uses a tag at a certain time stamp. In this paper, we use the Hawkes process (Hawkes 1971) to model the time information in users' tagging behavior. It is a non-Markov extension of the Poisson process to describe the self-motivated process, in which the occurrence of each event will contribute to the next occurrence of it. It is defined as

$$\tau(s) = \tau_0 + \sum_{s_i<s} E(s,s_i) \tag{8}$$

where $E(s,s_i) \geq 0$ is the incentive function to describe the time interval and τ_0 is the initial intensity. Users tend to visit some kind of items in a short period and annotate items with same tags. When some tags with similar meaning can be used, users tend to use tags they are familiar with. Hence, users will use tags they have used most frequently and most recently again, which has the self-motivate feature. In this paper, we employ the intensity function in Hawkes process to fit users' tagging pattern.

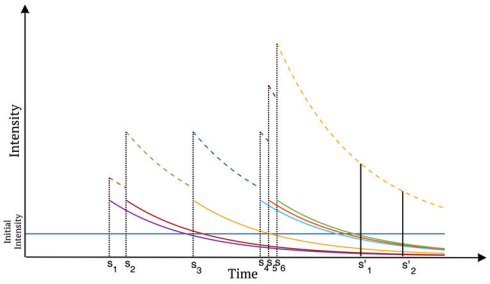


Figure 2: The example of intensity function

Note that if $\tau_0 = 0$ and $E(s,s_i) = (s-s_i)^{-d}$, where d is the intensity parameter, this function is the same as the time function in BLL-like models (Kowald et al. 2014; 2015a; 2015b). Different from it, we use exponential function $E(s,s_i) = exp^{-d(s-s_i)}$. It is because that exponential function can reduce time complexity greatly by changing accumulation form of power function to recursion form. Meanwhile, it can also ensure almost the same performance as power function. Figure 2 is an example of Hawkes process's conditional intensity function. $s_1, s_2, ..., s_6$ are the time stamps of events. The colorful solid line indicates the impact value of each event's influence on the future event, and the colorful dotted line represents the influence of all the past events on the future event. Our task is to calculate the intensity values at time $s_1^{'}$ and $s_2^{'}$. The incentive functions' sum $\tau^*(s_1')$ at time $s_1^{'}$ in recursion form is written as

$$\begin{aligned}\tau^*(s_1') &= \sum_{i=1}^{6} E(s_1', s_i) = \sum_{i=1}^{6} \exp^{-d(s_1'-s_i)} \\ &= \exp^{-d(s_1'-s_6)} \left(1 + \sum_{i=1}^{5} \exp^{-d(s_6-s_i)}\right) \\ &= \exp^{-d(s_1'-s_6)} \left(1 + \tau^*(s_6)\right)\end{aligned} \tag{9}$$

We can find that the intensity value of the event at time $s_1^{'}$ is only related to the intensity value of the event at time s_6. Finally, the intensity that user u uses tag t at time s is

$$\tau(u,t,s) = \sum_{s_i<s} \exp^{-d(s-s_i)} = \exp^{-d(s-s_{last})} \left(1 + \tau(s_{last})\right) \tag{10}$$

where $\tau_0 = 0$ and s_{last} is the last event occurrence time before time s.

Time-aware PITF In this paper, considering both time awareness and personalization aspects, we propose the Time Aware PITF (TAPITF) model. Given user-item pair $< u, m >$, the prediction value of candidate tag t is as

$$\hat{y}^s_{umt} = w^s_{ut}\mathbf{P}_u^\top \mathbf{T}_t^{\mathcal{P}} + w_{mt}\mathbf{Q}_m^\top \mathbf{T}_t^{\mathcal{Q}} \tag{11}$$

where $w^s_{u,t}$ is the weight of user-tag-time triple $< u,t,s >$, $w_{m,t}$ is the weight of item-tag pair $< m,t >$. Specifically, if $w^s_{u,t}$ and $w_{m,t}$ are set to constant 1, it is equal to PITF. When removing the interactions $\mathbf{P}_u^\top \mathbf{T}_t^{\mathcal{P}}$ and $\mathbf{Q}_m^\top \mathbf{T}_t^{\mathcal{Q}}$ and adding tuning parameters to $w^s_{u,t}$ and $w_{m,t}$, our model is

in the same form as $BLL + MP_m$. In the tagging record of user u , the higher the frequency of the past occurrences of tag t is, the larger $w^s_{u,t}$ is. Smaller recency between the user's past tagging time stamp s and the current time will rise $w^s_{u,t}$ too. Similarly, the higher frequency of all users use tag t to annotate item m is, the higher $w_{m,t}$ is. Meanwhile, if u has never used t before s, or nobody has assigned t to item m, w^s_{ut} and w_{mt} also need to be bigger than 0 to guarantee that tags never occurring in the record can also be recommended. Specifically, w^s_{ut} is represented as

$$w^s_{ut} = 1 + \log_{10}(1 + 10^{\alpha^{\mathcal{P}}} \cdot ||\tau(u,t,s)||) \tag{12}$$

where constant $\alpha^{\mathcal{P}}$ controls the growth rate of weight. $||\tau(u,t,s)|| = \frac{\tau(u,t,s)}{\sum_{t \in \mathcal{T}_u} \tau(u,t,s)}$ is the normalized value. $\mathcal{T}_u$ is the tag set used by user u. The larger $||\tau(u,t,s)||$ is, the larger w^s_{ut} is. Specifically, $||\tau(u,t,s)|| = 0$, $w^s_{ut} = 1$ means that user u had not used tag t before time stamp s. For weight w_{mt}, similar to $BLL + MP_m$ (Kowald et al. 2014), it is modeled as

$$w_{mt} = 1 + \log_{10}(1 + 10^{\alpha^{\mathcal{Q}}} \cdot |||\hat{\mathbf{Y}}_{mt}|||) \tag{13}$$

where constant $\alpha^{\mathcal{Q}}$ controls the growth rate, $|\hat{\mathbf{Y}}_{mt}|$ is the frequency that items m is annotated by tag t. Furthermore, in Eq.11, $\mathbf{P}^\top_u \mathbf{T}^{\mathcal{P}}_t$ (or $\mathbf{Q}^\top_m \mathbf{T}^{\mathcal{Q}}_t$) inherits from PITF, which models the latent features of users, items and tags well and improves the recommendation novelty.

Parameter Learning of TAPITF To learn the model's parameters, TAPITF utilizes the BPR framework to maximize the pair-wise ranking objective function. The whole algorithm is demonstrated in algorithm 1. Before iteration, according to the time stamps and frequencies in training set, weight w^s_{ut} of the $< u,t,s >$ triple and weight w_{mt} of the $< m,t >$ pair in each record $y =< u,m,t,s >$ are calculated. In each iteration, we need to sample positive tag and the corresponding negative tag. Because the weight w^s_{ut} of positive sample $< u,t,s >$ and weight w_{mt} of $< m,t >$ pair have been precalculated, we only need to compute weight $w^{s_A}_{u,t_B}$ of the negative sample $< u,t_B,s_A >$. Finally, the latent factors are optimized by stochastic gradient descent iteratively from line 3 to line 13.

Time Complexity Analysis In algorithm 1, for every record $y =< u,m,t,s >$, the weight w^s_{ut} and weight w_{mt} can be precalculated before iteration. Therefore, in each iteration, compared to PITF, only the negative weight $w^{s_A}_{ut_B}$ needs to be computed additionally. It is only related to user u, the time stamp s_A, tag set of user u and the negative tag t_B. There are two situations when calculating $w^{s_A}_{ut_B}$: (1) negative tag sample t_B does not exist in user u's tag set $\mathcal{T}_u$, then $w^{s_A}_{ut_B} = 1$; (2) if negative tag sample t_B exists in user u's tag set $\mathcal{T}_u$, we only need to find the last time u used t_B before time s_A by binary search to compute $w^{s_A}_{ut_B}$, which has the time complexity of $O(\log(n_{ut_B}))$. n_{ut_B} is the frequency that user u used tag t_B. Here we assume that each record is iterated one time, then expectation of the time complexity to compute $w^{s_A}_{ut_B}$ is related to the average frequency that tag t_B is used. In the worst case, where each negative tag sample has been used by the target user, the time complexity of each iteration is $O(K + \log(n_{ut_B}))$. It can be found that even when the data set is relatively dense, TAPITF will only cost a little more time in each iteration than PITF. Considering that the real world data is very sparse, the additional time cost is limited and can be accepted.

Algorithm 1: An Optimization Algorithm for TAPITF.

1 For each record $y =< u,m,t,s >$, calculate the user-tag-time weight w^s_{ut} and the items-tag weight w_{mt};
2 Initialize $\mathbf{P}, \mathbf{Q}, \mathbf{T}^{\mathcal{P}}, \mathbf{T}^{\mathcal{Q}}$ by gaussian distribution $N(0, 0.01)$; **repeat**
3 Uniformly sample $y =< u,m,t_A,s_A >$ and the corresponding negative tag sample t_B from train data, and calculate negative weight $w^{s_A}_{ut_B}$;
4 $\hat{y}_{umt_At_B} \leftarrow \hat{y}_{umt_A} - \hat{y}_{umt_B}$;
5 $\delta \leftarrow (1 - \sigma(\hat{y}_{umt_At_B}))$;
6 **for** *k from* 1 *to* K **do**
7 $\mathbf{P}_{uk} \leftarrow \mathbf{P}_{uk} + \iota \cdot (\delta \cdot (\hat{w}^{s_A}_{ut_A} \cdot \mathbf{T}^{\mathcal{P}}_{t_Ak} - \hat{w}^{s_A}_{ut_B} \cdot \mathbf{T}^{\mathcal{Q}}_{t_Bk}) - \lambda \cdot \mathbf{P}_{uk})$
8 $\mathbf{Q}_{mk} \leftarrow \mathbf{Q}_{mk} + \iota \cdot (\delta \cdot (\hat{w}^{s_A}_{ut_A} \cdot \mathbf{T}^{\mathcal{P}}_{t_Ak} - \hat{w}^{s_A}_{ut_B} \cdot \mathbf{T}^{\mathcal{Q}}_{t_Bk}) - \lambda \cdot \mathbf{Q}_{mk})$
$\mathbf{T}^{\mathcal{P}}_{t_Ak} \leftarrow \mathbf{T}^{\mathcal{P}}_{t_Ak} + \iota \cdot (\delta \cdot \mathbf{P}_{uk} \cdot w^{s_A}_{ut_A} - \lambda \cdot \mathbf{T}^{\mathcal{P}}_{t_Ak})$
9 $\mathbf{T}^{\mathcal{P}}_{t_Bk} \leftarrow \mathbf{T}^{\mathcal{P}}_{t_Bk} + \iota \cdot (\delta \cdot (-\mathbf{P}_{uk}) \cdot w^{s_A}_{ut_B} - \lambda \cdot \mathbf{T}^{\mathcal{P}}_{t_Bk})$
10 $\mathbf{T}^{\mathcal{Q}}_{t_Ak} \leftarrow \mathbf{T}^{\mathcal{Q}}_{t_Ak} + \iota \cdot (\delta \cdot \mathbf{Q}_{mk} \cdot w_{mt_A} - \lambda \cdot \mathbf{T}^{\mathcal{Q}}_{t_Ak})$
11 $\mathbf{T}^{\mathcal{Q}}_{t_Bk} \leftarrow \mathbf{T}^{\mathcal{Q}}_{t_Bk} + \iota \cdot (\delta \cdot (-\mathbf{Q}_{mk}) \cdot w_{mt_B} - \lambda \cdot \mathbf{T}^{\mathcal{Q}}_{t_Bk})$
12 **end**
13 **until** *Convergence*;

Table 1: Data statistics

	core	N	M	T	U	U/M
Movielens	-	2,113	5,908	9,079	27,712	4.69
	3	656	2,376	2,061	18,427	7.76
LastFM	-	1,892	12,523	9,749	71,064	5.68
	3	1,277	5,940	2,761	59,692	10.05
Delicious	-	1,867	69,223	40,897	104,799	1.51
	3	1,458	5,074	4,233	20,543	4.05

Experiments

We evaluate the models on the three public data sets Movielens, LastFM and Delicious described in table 1. N is the number of users, M is the number of items and T is the number of tags. U is the number of user-item pairs and U/M indicates the sparsity of the data set. We use leave-one-out to split data set into train set and test set, which is that for each user, his tagging records on a certain item are randomly removed from the training set S_{train} and put into the test set S_t. When the user only annotated one item, his tagging records are all put into train set. All data sets are p-cores. The $pcore$ of data set S is the largest subset of S with the property that every user, every item and every tag has to occur for at least p times in all user-item pairs. In our experiments, we have the no core (unfiltered data set) and core 3 data set.

Experimental Settings

Metrics We evaluate the performance of algorithms in two aspects:

- Accuracy: Accuracy is the most important measurement in the off line evaluation of recommender system. The common measurements include precision, recall and $F1$ value. For all user-item pairs (u, m) in the test data set S_t, they are defined by:

$$Prec(S_t, n) = \frac{|Top(u, m, n) \bigcap \{t|(u, m, n) \in S_t\}|}{n} \quad (14)$$

$$Rec(S_t, n) = \frac{|Top(u, m, n) \bigcap \{t|(u, m, t) \in S_t\}|}{|\{t|(u, m, t) \in S_t\}|} \quad (15)$$

$$F1(S_t, n) = \frac{2 \cdot Prec(S_t, n) \cdot Rec(S_t, n)}{Prec(S_t, n) + Rec(S_t, n)} \quad (16)$$

  We use $F1@5$ to measure the accuracy.

- Novelty: Novelty is used to measure the model's ability to recommend new tags to users. The novelty of the recommended tag list is calculated using the Average Inverse Popularity ($AIP@10$) metric in (Belém et al. 2013). A recommended tag is novel if it was not previously used to annotate the target item. Thus, the lower the popularity of a tag for an item is, the higher its novelty.

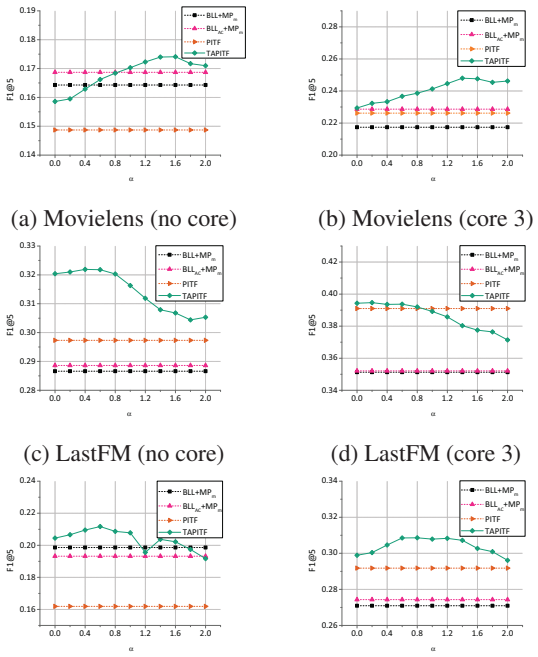
Comparison Methods We compare our proposed method TAPITF with the following methods.

- **MP**$_m$: recommends the most popular tags on the target resource to the target user.
- **PITF** (Rendle and Schmidt-Thieme 2010): a modified tensor factorization approach explicitly modeling the pairwise interaction among users, items and tags.
- **BLL+MP**$_m$ (Kowald et al. 2014): combines BLL and MP_m. The BLL part exploits recency and frequency in users' tagging behavior to predict the probability that users reuse a certain tag.
- **BLL**$_{AC}$**+MP**$_m$ (Kowald et al. 2015a): similar to **BLL+MP**$_m$. This method adds association component to BLL to model the target item's impact on users. According to the experimental result in (Kowald and Lex 2015), it can achieve the highest accuracy among existing approaches.

Settings For $BLL + MP_m$ and $BLL_{AC} + MP_m$, we follow the parameter setting in (Kowald and Lex 2015). Latent factor dimension $K = 64$, regularization factor $\lambda = 0.00005$ and learning rate is 0.05. In TAPITF, $d = 0.5$, time unit is day. Latent factor dimension and regularization factor is the same as PITF. The iteration number of PITF and TAPITF are both 100.

Experimental Results

Impact of weight parameter α Parameter α is used in equation 12 and 13. In Eq.12, it is used to distinguish observed user-tag pairs from unobserved ones. In Eq.13, it is to distinguish observed item-tag pairs from unobserved ones. α controls the influence of time factor and popular tags. In this section, we adjust α to see its influence on our model. As figure 3 shows, for all data sets, performances of TAPITF improve with the increase of α to the optimum. After that, the performances will drop as α increases. When α is set to a proper value, TAPITF outperforms PITF and $BLL + MP_m$ almost on all the experimental data sets. The experimental results show that it is helpful to incorporate the temporal factor and consider the popular tags on the target item. As for data sparsity problem, on the unfiltered data sets Movielens (no core) and Delicious (no core), $BLL + MP_m$ and $BLL_{AC} + MP_m$ outperform PITF, which shows that $BLL + MP_m$ and $BLL_{AC} + MP_m$ are better at handling sparse data. After incorporating temporal factor and the impact of the most popular tags on target items, TAPITF can achieve better performance than $BLL_{AC} + MP_m$ on all unfiltered data sets. Therefore, it is helpful for relieving data sparsity to reference the most popular tags on the target items. On the other hand, on the filtered data set (core 3), PITF has better performance than BLL-like models. The reason may be that PITF model is suitable for dense data sets. Further, TAPITF also considers time information, and thus it can outperform other comparative methods greatly. In the next experiments, α is set to its proper value for TAPITF model.



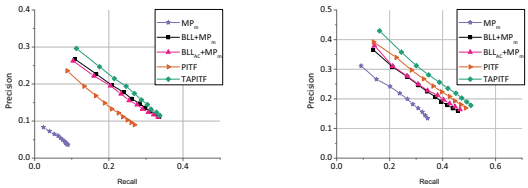
(a) Movielens (no core) (b) Movielens (core 3)

(c) LastFM (no core) (d) LastFM (core 3)

(e) Delicious (no core) (f) Delicious (core 3)

Figure 3: Performance with different α

Performance Comparison and Analysis In this section, we compare our model TAPITF with the existing state-of-art tag recommendation approaches. Figure 4 demonstrates the recall/precision curves from top 1 to top 10 on Delicious data set. As we can see, TAPITF performs almost the best on Delicious, which presents that incorporating time information



(a) Delicious (no core) (b) Delicious (core 3)

Figure 4: Recall/Precision curves for Delicious dataset

Table 2: Comparison of AIP@10

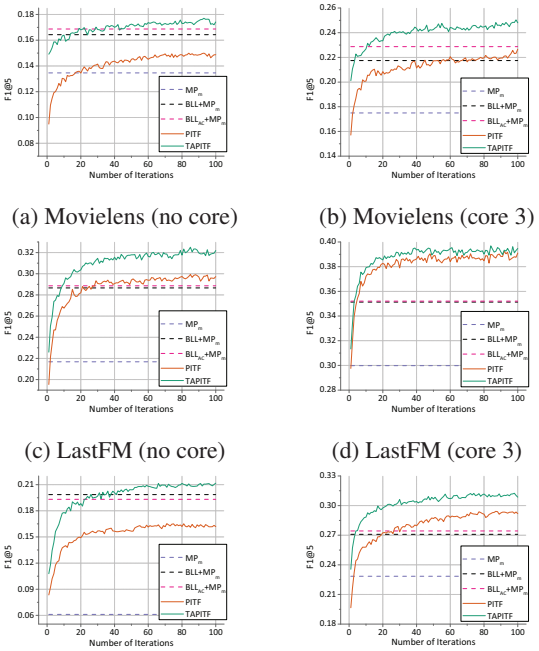
Dataset	Core	MP_m	BLL $+MP_m$	BLL_{AC} $+MP_m$	PITF	TAPITF
Movielens	-	0.806	0.788	0.786	**0.854**	0.795
	3	0.761	0.762	0.757	**0.803**	0.77
LastFM	-	0.732	0.724	0.709	**0.779**	0.739
	3	0.668	0.660	0.639	**0.697**	0.696
Delicious	-	0.881	0.873	0.878	**0.949**	0.894
	3	0.787	0.767	0.767	**0.827**	0.798

into PITF is very effective. On the unfiltered Delicious data, the performance of TAPITF is worse than $BLL+MP_m$ and $BLL_{AC}+MP_m$ when the top n list size is large. In most real applications, the candidate tag size is often small and sometimes users only care the top one recommended tag. Hence, it is reasonable that TAPITF model is effective for real applications.

In terms of recommendation novelty, table 2 shows that PITF can achieve the best recommendation novelty for all experimental data sets. TAPITF is slightly worse than PITF and better than MP_m, $BLL+MP_m$ and $BLL_{AC}+MP_m$. Here we discuss that why PITF can achieve the best novelty among these methods. Popular tags on the target item are very likely to be recommended by MP_m and BLL-like models. However, PITF is a collaborative filtering method. It can exploit the collaborative influence between users and items. In this way, tags not been used on the target item also have the chance to be recommended. Hence, PITF has the best novelty.

Next we compare TAPITF with PITF in detail. Figure 5 compares the accuracies of TAPITF and PITF with different iteration number. To ensure the convergence of algorithm, the iteration count is set to 100. TAPITF model outperforms PITF in terms of accuracy and convergence speed for data sets Movielens, LastFM, Delicious no core and Delicious core 3. TAPITF can get convergence after 40 iterations on all experimental data sets, while PITF can gradually get convergence after 60 iterations. In addition, the convergence speed of TAPITF in the first 20 iterations is faster than PITF. The main reason is that TAPITF has the user-tag-time weight and the items-tag weight, then it can achieve similar recommendation performance like BLL models even without parameter learning.

As for running time, compared to PITF, TAPITF has the additional cost to calculate the user-tag-time weight in each iteration. Table 3 compares the running time of TAPITF and PITF on different data sets. In our experiments, we choose



(a) Movielens (no core) (b) Movielens (core 3)

(c) LastFM (no core) (d) LastFM (core 3)

(e) Delicious (no core) (f) Delicious (core 3)

Figure 5: Accuracy and convergence speed

Table 3: Parameter learning time

Dataset	Core	PITF	TAPITF
Movielens	-	10.3	15.2
	3	5.8	8.2
LastFM	-	41.4	63.3
	3	32.6	47.4
Delicious	-	110.7	150.0
	3	16.8	25.6

the average time of 100 iterations to be the final running time (in each iteration, the loop size is 100 times of training set size). It also can be observed that running time on all data sets is slightly longer than PITF, even on the relative dense data set Movielens. It is because that we employ the exponential function in Hawkes Process to reduce the time complexity of computing the temporal weight. Therefore, our proposed model TAPITF is effective and efficient.

Conclusion

In this paper, we propose an unified tag recommendation model TAPITF by considering both time awareness and personalization. We utilize temporal point process to explicitly model time information. In addition, we use the exponential-form intensity function to save computation cost. We also consider the popular tags on the target item. In this way, our proposed model can depict the temporal dynamics in users' tagging behavior and capture potential interests of users effectively. The experimental results on real data sets show that TAPITF outperforms the state-of-art personalized tag recommendation methods in accuracy, and can achieve better recommendation novelty.

References

Anderson, J. R.; Bothell, D.; Byrne, M. D.; Douglass, S.; Lebiere, C.; and Qin, Y. 2004. An integrated theory of the mind. *Psychological review* 111(4):1036.

Belém, F.; Martins, E.; Almeida, J.; and Gonçalves, M. 2013. Exploiting novelty and diversity in tag recommendation. In *European Conference on Information Retrieval*, 380–391. Springer.

Charlin, L.; Ranganath, R.; McInerney, J.; and Blei, D. M. 2015. Dynamic poisson factorization. In *Proceedings of the 9th ACM Conference on Recommender Systems*, 155–162. ACM.

De Lathauwer, L.; De Moor, B.; and Vandewalle, J. 2000. A multilinear singular value decomposition. *SIAM journal on Matrix Analysis and Applications* 21(4):1253–1278.

EmBree, J. D., and Handcock, M. S. 2016. Spatial temporal exponential-family point process models for the evolution of social systems. *arXiv preprint arXiv:1610.00048*.

Gong, Y., and Zhang, Q. 2016. Hashtag recommendation using attention-based convolutional neural network. In *IJCAI*, 2782–2788.

Hawkes, A. G. 1971. Spectra of some self-exciting and mutually exciting point processes. *Biometrika* 58(1):83–90.

Hotho, A.; Jäschke, R.; Schmitz, C.; and Stumme, G. 2006. Information retrieval in folksonomies: Search and ranking. In *ESWC*, volume 4011, 411–426. Springer.

Jäschke, R.; Marinho, L.; Hotho, A.; Schmidt-Thieme, L.; and Stumme, G. 2007. Tag recommendations in folksonomies. *Knowledge Discovery in Databases: PKDD 2007* 506–514.

Koren, Y. 2010. Collaborative filtering with temporal dynamics. *Communications of the ACM* 53(4):89–97.

Kowald, D., and Lex, E. 2015. Evaluating tag recommender algorithms in real-world folksonomies: A comparative study. In *Proceedings of the 9th ACM Conference on Recommender Systems*, 265–268. ACM.

Kowald, D.; Seitlinger, P.; Trattner, C.; and Ley, T. 2014. Long time no see: The probability of reusing tags as a function of frequency and recency. In *Proceedings of the 23rd International Conference on World Wide Web*, 463–468. ACM.

Kowald, D.; Kopeinik, S.; Seitlinger, P.; Ley, T.; Albert, D.; and Trattner, C. 2015a. Refining frequency-based tag reuse predictions by means of time and semantic context. In *Mining, Modeling, and Recommending'Things' in Social Media*. Springer. 55–74.

Kowald, D.; Seitlinger, P.; Kopeinik, S.; Ley, T.; and Trattner, C. 2015b. Forgetting the words but remembering the meaning: Modeling forgetting in a verbal and semantic tag recommender. In *Mining, Modeling, and Recommending'Things' in Social Media*. Springer. 75–95.

Li, Y.; Liu, T.; Jiang, J.; and Zhang, L. 2016. Hashtag recommendation with topical attention-based lstm. Coling.

Marinho, L. B., and Schmidt-Thieme, L. 2008. Collaborative tag recommendations. *Data Analysis, Machine Learning and Applications* 533–540.

Nguyen, H. T.; Wistuba, M.; Grabocka, J.; Drumond, L. R.; and Schmidt-Thieme, L. 2017. Personalized deep learning for tag recommendation. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 186–197. Springer.

Rawat, Y. S., and Kankanhalli, M. S. 2016. Contagnet: exploiting user context for image tag recommendation. In *Proceedings of the 2016 ACM on Multimedia Conference*, 1102–1106. ACM.

Rendle, S., and Schmidt-Thieme, L. 2009. Factor models for tag recommendation in bibsonomy. In *ECML/PKDD 2008 Discovery Challenge Workshop, part of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 235–243.

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. 2009a. 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.

Rendle, S.; Freudenthaler, C.; Gantner, Z.; and Schmidt-Thieme, L. 2009b. Bpr: Bayesian personalized ranking from implicit feedback. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, 452–461. AUAI Press.

Schoenberg, F. P. 2010. *Introduction to Point Processes*.

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.

Tucker, L. R. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika* 31(3):279–311.

Wang, H.; Shi, X.; and Yeung, D.-Y. 2015. Relational stacked denoising autoencoder for tag recommendation. In *AAAI*, 3052–3058.

Xiao, S.; Yan, J.; Li, C.; Jin, B.; Wang, X.; Yang, X.; Chu, S. M.; and Zha, H. 2016. On modeling and predicting individual paper citation count over time. In *IJCAI*, 2676–2682.

Xu, L.; Duan, J. A.; and Whinston, A. 2014. Path to purchase: A mutually exciting point process model for online advertising and conversion. *Management Science* 60(6):pgs. 1393–1412.

Yan, J.; Xiao, S.; Li, C.; Jin, B.; Wang, X.; Ke, B.; Yang, X.; and Zha, H. 2016. Modeling contagious merger and acquisition via point processes with a profile regression prior. In *International Joint Conference on Artificial Intelligence*, 2690–2696.

Yin, D.; Hong, L.; Xue, Z.; and Davison, B. D. 2011. Temporal dynamics of user interests in tagging systems. In *Twenty-Fifth AAAI conference on artificial intelligence*.

Zhang, L.; Tang, J.; and Zhang, M. 2012. Integrating temporal usage pattern into personalized tag prediction. *Web Technologies and Applications* 354–365.