A New Weighted-learning Approach for Exploiting Data Sparsity in Tag-based Item Recommendation Systems

Noor Ifada [1]*, Richi Nayak [2]

[1]*Informatics Department, University of Trunojoyo Madura, Indonesia*
[2]*School of Computer Science, Queensland University of Technology, Australia*
* Corresponding author's Email: noor.ifada@trunojoyo.ac.id

Abstract: The tag-based recommendation systems that are built based on tensor models commonly suffer from the data sparsity problem. In recent years, various weighted-learning approaches have been proposed to tackle such a problem. The approaches can be categorized by how a weighting scheme is used for exploiting the data sparsity – like employing it to construct a weighted tensor used for weighing the tensor model during the learning process. In this paper, we propose a new weighted-learning approach for exploiting data sparsity in tag-based item recommendation system. We introduce a technique to represent the users' tag preferences for leveraging the weighted-learning approach. The key idea of the proposed technique comes from the fact that users use different choices of tags to annotate the same item while the same tag may be used to annotate various items in tag-based systems. This points out that users' tag usage likeliness is different and therefore their tag preferences are also different. We then present three novel weighting schemes that are varied in manners by how the ordinal weighting values are used for labelling the users' tag preferences. As a result, three weighted tensors are generated based on each scheme. To implement the proposed schemes for generating item recommendations, we develop a novel weighted-learning method called as *WRank* (Weighted Rank). Our experiments show that considering the users' tag preferences in the tensor-based weighting-learning approach can solve the data sparsity problem as well as improve the quality of recommendation.

Keywords: Data sparsity, Tag-based Item recommendation, Tensor model, User tag preference, Weighted-learning approach, Weighting scheme.

1. Introduction

Tag-based systems permit their users to freely annotate their favourite items using tags. Users may annotate an item using various tags as well as repeatedly using a tag for annotating various items. When a user annotates an item using a tag, a $\langle user, item, tag \rangle$ ternary relation is naturally formed. Over time, these ternary relations become the tagging data that can be used to learn the users' past behaviours. The learning results can then be used to generate ranked list of recommended items that the users might like [1, 2] or to retrieve items that they are searching for [3, 4].

The tag-based recommendation methods based on the tensor models have shown to outperform those of the matrix models due to their capacity of representing the latent relations inherent in tagging data more efficiently [5]. However, these methods commonly suffer from data sparsity problem and compromise accuracy. This happens since users of tag-based systems typically use and annotate only a few tags and items, respectively. Existing methods mostly use the binary interpretation scheme to populating the tensor model, i.e., the observed tagging entries are denoted as "1" while the unobserved ones are denoted as "0" [5]. Thus, a tensor model populated with such *Boolean* scheme usually results in an over-domination of unobserved entries.

Researchers have been proposing to deal with the data sparsity problem occurs in tensor models by implementing a weighted-learning approach that used a weighting scheme to employ both observed and unobserved data in either the modelling or learning process. In this ongoing research topic, a weighting scheme naturally regards the observed data as positive feedback. Meanwhile, the non-observed ones are discriminated as negative and/or other types of feedback. In other words, not all of the non-observed data should be simply regarded as negative feedback.

In this paper, we propose a new weighted-learning approach for exploiting data sparsity in tag-based item recommendation system. We introduce a technique to represent the users' tag preferences that are used for leveraging the weighted-learning approach such that both observed and unobserved entries are taking into account in the learning process.

The key idea of the proposed technique comes from the fact that users use different choices of tags to annotate the same item while the same tag may be used to annotate various items in tag-based systems. This points out that users' tag usage likeliness is different and therefore their tag preferences are also different. We then present three novel weighting schemes that are varied in manners by how the ordinal weighting values are used for labelling the users' tag preferences. As a result, three weighted tensors are generated based on each weighting scheme. To implement the proposed weighting schemes for generating item recommendations, we develop a novel weighted-learning method called as *WRank* (Weighted Rank).

The contribution of this paper is proposing a new weighted-learning approach for tag-based item recommendation. In the approach, we present a technique to generate the users' tag preferences that are used to formulate three distinct weighting schemes, resulting in three weighted tensors. We then develop a new tag-based item recommendation method to employ the proposed weighting schemes. Our experiments show that considering the users' tag preferences in the tensor-based weighting-learning approach can solve the data sparsity problem as well as improve the quality of recommendation.

The rest of this paper is organized as follows. Section 2 describes the previous works related to the tag-based recommendation and weighted-learning approach. Section 3 details our proposed weighted-learning approach, while Section 4 describes the experiment setup. Section 5 shows the empirical results as well as the discussions based on the experiments. The conclusion of this paper is presented in Section 6.

2. Related Work

2.1 Tag-based Recommendation using Tensor Model

A tag-based recommendation is an application that incorporates tags in the recommendation process to improve the recommendation quality [1]. In this case, users of such application are given the opportunity to freely label items of their interest using tags and forming tagging data. One efficient solution to represent the $\langle user, item, tag \rangle$ ternary relationships within the tagging data is by using a tensor model [5]. The model reveals the latent relationships within the multi-dimensions by generating latent factors through tensor factorization process. The latent factors are then used for generating a reconstructed tensor that generated a ranked list of recommendations. Tensor factorization can be categorized as Tucker or Candecomp/Parafac (CP) based techniques. Tucker factorization decomposes a tensor into mode-n product of a core tensor and three factor matrices of each dimension. While CP factorization decomposes a tensor into a sum of component rank-one tensors.

Symeonidis, et al. [6] is among the first researchers that studied the implementation of the tensor model for a tag-based recommendation. They proposed a framework for the tag-based recommendation that implemented HOSVD, i.e., a common factorization of Tucker-based. In our previous work, we developed *TRPR* [7] to outcompete *HOSVD* in tag-based item recommendation. Our method implemented various factorization techniques to generate the candidates of recommendations. Afterwards, we implemented a probabilistic technique to re-rank the candidates and the results are used as the final list of recommendations for the target users. Experiment results on Delicious and Last.fm datasets showed that TRPR performs best when it is implemented using CP factorization technique.

Klašnja-Milićević, et al. [8] proposed a tag-based recommendation method that uses ranking with tensor factorization technique for a programming tutoring system. The method tackled the scalability problem by employing a clustering technique to the learners' data. Experiment results showed that the implementation of ranking factorization technique can outperform FolkRank and HOSVD.

Yang, et al. [9] developed the *Tagrec-CMTF* method for tag recommendation. The method combined the implementation of CP factorization, tag graph, and also the tag usage of items and users. The results of experiments on Last.fm and Bibsonomy datasets showed that *Tagrec-CMTF* outcompetes *CP* and *PITF* methods. However, The nature of the combinatory approach makes the process of profiling and learning stages more complex than the regular factorization based methods.

Tang, et al. [10] proposed a Tucker based approach for tag-based item recommendation. They tried to incorporate the users and items profiles, constructed by implementing a partial-Tucker factorization, within the learning process. Despite its promising results, the approach assumed that the information used to create the profiles always contains a homogonous type of data, which make it difficult to be implemented for the heterogonous problems.

In this paper, we propose a tag-based item recommendation method that implements a CP

factorization technique. We name our method as *WRank* (Weighted Ranking) as it enhances the capability of the factorization technique for generating a ranked list of recommendation by the implementation of weighted-learning approach (discussed in the next sub-section). We benchmark our proposed method with *HOSVD* [6] and *TRPR* [7].

2.2 Tensor-based Weighted-learning Approach

The tensor-based weighted-learning approach can be categorized by how a weighting scheme is used for exploiting the data sparsity. A weighting scheme can be used for: (a) populating the entries of the tensor model used as the learning model, (b) weighing the regularization during the learning process, and (c) weighing the tensor model during the learning process.

Implementing a weighting scheme for building the tensor model is to make sure that both the observed and unobserved data are represented in the learning model. Commonly, a tensor model is built by simply interpreting the data based on a binary scheme, i.e., the observed data is regarded as positive entries and weighted as "1" while letting the rest as "0". As a result, the tensor becomes very sparse since the number of observed entries is usually a lot smaller than those of the unobserved ones. Rendle and Schmidt-Thieme [11] proposed the *PITF* method for tag recommendation. The method used the *set-based* scheme, pairwise factorization, and AUC-based optimization. The scheme discriminates the weight of the observed and non-observed tagging data in the tensor model. In this case, the initial tensor model is built as pairwise sets of the positive entries of the observed data and the negative entries of the non-observed ones, in which the former always have higher weight than the latter. Despite the promising results compared to CP and Tucker based methods, *PITF* oversimplifies its weighting interpretation towards the non-observed data. Moreover, the implementation of the AUC-based pairwise learning does not differentiate the mistakes occur at the top and bottom list of recommendations. In our previous work, we developed *Do-Rank* [12] to outcompete PITF in tag-based item recommendation. The method used the *User Tag Set* (UTS) scheme, CP factorization, and DCG-based optimization. The scheme determines the positive entries of observed data in the same way as the *set-based*, however, it further discriminates the non-observed data such that not all of them are regarded as the negative entries. Anyhow, both aforementioned methods proved that a portion of unobserved entries is effecting the recommendation quality in the weighted-learning based methods. Lim and Kim [13] developed the *RateRT* method for tag-based item recommendation. The method employed the *tag-weighting* scheme, HOSVD factorization, and MSE-based optimization. The scheme populates the tensor learning model based on a linear combination of the rating-based and emotion-based tag weights. Lim and Kim [14] then tried to extend their aforementioned work by adopting the BM25 technique to construct the tag-based user profile and tag-based item profiles. The recommendation results are ranked using the similarity of the profiles. Meanwhile, Zhang, et al. [15] proposed the *TRS* method that is based on the improvement of Artificial Fish Swarm Algorithm (AFSA) for tag recommendation. The method used rating-tag scheme, Tucker factorization, and MSE-based optimization. The scheme populates the tensor model according to the weight of the tag in each user-item-tag combination multiplies by the rating given by the user to the item. The three aforementioned methods indeed enrich the tensor learning model, however their implementations always require the availability of both the tagging and rating data.

The regularization is the parameter, commonly set as constant, that can be used to avoid overfitting in the tensor-based optimization process. Hosono, et al. [16] proposed the *WTNN* method for image restoration that implemented the generalization of Local Color Nuclear Norm (LCNN) regularization function and HOSVD factorization. The method conducts colour image denoising towards the image tensor model. Mu, et al. [17] developed the *WTNN2* method for image recovery and video completion that used a weighted tensor nuclear norm regularization function and HOSVD factorization. The regularization is defined as the sum of the weighted nuclear norm of all the tensor frontal slices.

The weighted-learning approach can also be implemented by distinctly weighing the tensor learning model using a weighted tensor during the learning process. In other words, such an approach always contains two tensors of the same size. Acar, et al. [18] is among the first researchers that studied the implementation of the weighted-learning approach that employs the usage of a weighted tensor. They proposed the *CP-WOPT* method for completing the missing values in the EEG and network traffic data. *CP-WOPT* is based on CP factorization and uses the *Boolean* weighting scheme for building the weighted tensor. The weighting scheme generates each entry of the weighted tensor such that it is a direct bijective mapping of the tensor learning model. The main idea of *CP-WOPT* is to focus on emphasizing the observed entries and disregarding the unobserved ones. Experiment results

showed that *CP-WOPT* outperformed the *INDAFAC* method. Filipović and Jukić [19] proposed the Tucker based weighted optimization method for image reconstruction that also used the *Boolean* weighting scheme for building the weighted tensor. The experiment results were promising, however, the method has not been proven that it can be implemented on a large dataset. Yuan, et al. [20] worked on the *TT-WOPT* method for image data completion that is based on a tensor-train factorization. The weighted tensor is built using the *Boolean* weighting scheme and is implemented to the sequences of the three-order image tensor-train models. The main idea of *TT-WOPT* is to find the factor core tensors of the tensor-train factorization. Iwata, et al. [21] tried to advance the *TT-WOPT* method for predicting the unknown parts of drug and the new drug therapeutic indications. However, due to the nature of the tensor-train factorization, *TT-WOPT* is naturally very sensitive to the setting of the TT-ranks since each number of ranks must be distinguishingly defined. Moreover, the initial values of the core tensors can influence the performance of *TT-WOPT*.

In this paper, we propose a new weighted-learning approach in which a proposed weighting scheme is used to construct a weighted tensor for differentiating the weight of each entry of the tensor model during the learning process. Our weighting scheme is formulated such that it takes into account the users' tag preferences. We benchmark our method with *PITF* [11], *Do-Rank* [12], and *CP-WOPT* [18]. Note that we adapt *PITF* for the task of tag-based item recommendation since it was originally built for tag recommendation.

3. The Proposed Approach

The framework of the proposed weighted-learning approach for exploiting data sparsity in tag-based item recommendation system consists of four main stages (Fig. 1): user profiling for building the tensor model, weighting scheme for generating the weighted tensor, weighted-based learning for generating the latent factors, and recommendation generation.

3.1 User Profiling

User profiling is the process of formally representing the user's past tagging activity that will be used as the learning model. In this paper, we denote $U = \{u_1, u_2, u_3, \dots, u_Q\}$ as the set of Q users, $I = \{i_1, i_2, i_3, \dots, i_R\}$ as the set of R items, and $T = \{t_1, t_2, t_3, \dots, t_S\}$ as the set of S tags. Given the tagging data $A : U \times I \times T$, a vector of a: $(u, i, t) \in A$ denotes that user u uses tag t for annotating item i. The ternary relations of tagging data can be modelled as a tensor model $\mathcal{Y} \in \mathbb{R}^{Q \times R \times S}$ where y_{uit} denotes the valuation of a tagging entry. In this case, the slices of the tensor can be seen as user matrices that store the annotation of items using tags.

The observed tagging data, A_{ob} where $A_{ob} \subseteq A$ and $|A_{ob}| \ll |A|$, defines that users have annotated items by using tags in the past. The *Boolean* interpretation scheme [6] is commonly used to build the user profile such that the tagging data is directly interpreted as binary data. In this case, each entry of $y_{uit} \in [0,1]$ corresponds to the type of the tagging data entries, i.e. if the entry is observed then $y_{uit} = 1$ and otherwise $y_{uit} = 0$.

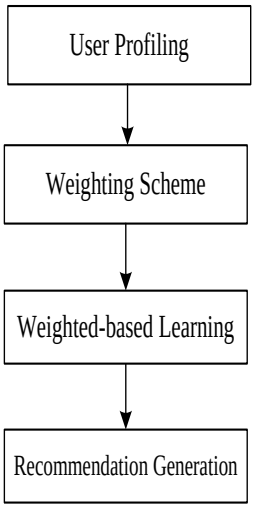


Figure 1. The framework of the proposed weighted-learning approach for tag-based item recommendation system

3.2 Weighting Scheme

Weighting scheme, the main component in a weighted-learning approach, controls the significance of the observed and unobserved entries in the learning process. Given the learning model represented as tensor $\mathcal{Y}$, we store the weighting values in a weighted tensor $\mathcal{W}$. Note that w_{uit} is a bijective mapping of y_{uit} that is not changing over the learning process.

3.2.1 Existing Weighting Scheme

The existing weighting scheme, i.e., *Boolean* scheme, emphasizes the observed entries of $\mathcal{Y}$ using the "1" values, while the unobserved entries are regarded as '0" values. The scheme determines the

weighting value of w_{uit} as a direct bijective mapping of the value of y_{uit} [18]:

$$w_{uit} = \begin{cases} 1 & \text{if } y_{uit} = 1 \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The fact that users can use various tags to annotate an item as well as use a tag to annotate numerous items points out that each user's tag usage likeliness is unique and therefore their tag preferences are also different. We employ this assumption to leverage the weighted-learning approach such that both observed and unobserved entries of $\mathcal{Y}$ are distinctively taking into account in the learning process.

3.2.2 Proposed User Tag Preference

We propose to form the user tag preference by capturing the characteristics of a user's tag usage likeliness through the latent features of the users and tags. In this case, we implement a non-negative matrix factorization (NMF) technique [22] to the mode-1 $Y_{(1)} \in \mathbb{R}^{Q \times RS}$ and mode-3 $Y_{(3)} \in \mathbb{R}^{S \times QR}$ matricizations of tensor $\mathcal{Y} \in \mathbb{R}^{Q \times R \times S}$:

$$Y_{(1)} = AC' \quad (2)$$

$$Y_{(3)} = BD' \quad (3)$$

where $A \in \mathbb{R}^{Q \times F}$ and $B \in \mathbb{R}^{S \times F}$ are respectively the latent features of the users and tags of size F. Meanwhile, $C \in \mathbb{R}^{RS \times F}$ and $D \in \mathbb{R}^{QR \times F}$ are the coefficient matrices. The user u tag usage likeliness towards tag t is then calculated as:

$$l_{ut} = \sum_{k=1}^{F} a_{uk} b_{kt} \quad (4)$$

where l_{ut}, a_{uk}, and b_{kt} are respectively the elements of matrices of the *User Tag Usage Likeliness* $L \in \mathbb{R}^{Q \times S}$, the *User Latent Feature* $A \in \mathbb{R}^{Q \times F}$, and the *Tag Latent Feature* $B \in \mathbb{R}^{R \times F}$. Fig. 2 shows the algorithm of generating the *User Tag Usage Likeliness*.

Two sets of user tag preferences are formed based on the likeliness scores: (i) *User Positive Tag Preference* set L_u^+, i.e., the list of tags that the user is possibly interested to use for annotating items, yet it is not explicitly exposed; and (ii) *User Negative Tag Preference* set L_u^-, i.e., the list of tags that the user does not want to use. The size of the sets is determined by v. In this case, L_u^+ is created based on the top scores of L_{u*}, where $L_u^+ \subseteq T$ such that $|L_u^+| \leq v$. While L_u^- is created based on the bottom scores of L_{u*}, where $L_u^- \subseteq T$ such that $|L_u^-| \leq v$.

Example: As shown in Fig. 3, the mode-1 and mode-3 matricizations of tensor $\mathcal{Y} \in \mathbb{R}^{3 \times 4 \times 5}$ result in $Y_{(1)} \in \mathbb{R}^{3 \times 20}$ and $Y_{(3)} \in \mathbb{R}^{5 \times 12}$. The implementation of NMF to $Y_{(1)}$ and $Y_{(3)}$, where $F = 2$, results in $A \in \mathbb{R}^{3 \times 2}$ and $B \in \mathbb{R}^{5 \times 2}$. Hence, by calculating $L \in \mathbb{R}^{3 \times 5}$ using Eq. (4) and by choosing $v = 2$, the *User Positive Tag Preference* and *User Negative Tag Preference* sets are formed as $L_{u_1}^+ = \{t_1, t_3\}$, $L_{u_2}^+ = \{t_2, t_4\}$, $L_{u_3}^+ = \{t_2, t_4\}$, $L_{u_1}^- = \{t_2, t_5\}$, $L_{u_2}^- = \{t_1, t_3\}$, and $L_{u_3}^- = \{t_1, t_3\}$.

3.2.3 Proposed Weighting Scheme

Based on the results of the proposed *User Tag Preference* stage, we propose three novel weighting schemes: (1) *Quaternary-Tag-Preference*, (2) *Ternary-Positive-Tag-Preference*, and (3) *Ternary-Negative-Tag-Preference*. The bottom side of Fig. 3 shows the examples of the constructed $\mathcal{W}$ based on each scheme. Note that, for ease of illustration, the examples only show the $\mathcal{W}$ entries for u_1.

Algorithm: Generating *User Tag Usage Likeliness*
Input: Training set $D_{train} \subseteq U \times I \times T$, the size of latent feature F
Output: *User Tag Usage Likeliness* $L \in \mathbb{R}^{Q \times S}$
$Q = |U|$, $R = |I|$, $S = |T|$, $\mathcal{Y} \in \mathbb{R}^{Q \times R \times S}$
Generate y_{uit}
Get $Y_{(1)}$ via mode-1 matricization of $\mathcal{Y}$
Get $Y_{(3)}$ via mode-3 matricization of $\mathcal{Y}$
Get $A \in \mathbb{R}^{Q \times F}$, $B \in \mathbb{R}^{S \times F}$ via NMF on $Y_{(1)}$ and $Y_{(3)}$
for $u \in U$ **do**
  for $t \in T$ **do**
    for $k \leftarrow 1$ *to* F **do**
      $l_{ut} \leftarrow a_{uk} b_{kt}$
/* $L \in \mathbb{R}^{Q \times S}$ */

Figure 2. The algorithm of generating *User Tag Usage Likeliness*

***Quaternary-Tag-Preference* weighting scheme:** This scheme generates four ordinal weighting values of $\{2,1,0,-1\}$. The "2" value represents the observed entries, whereas "1" and "-1" values represent the unobserved entries that respectively belong to L_u^+ and L_u^-. The rest of the entries are denoted as "0". The rule for generating the values of $\mathcal{W}$ entries is formulated as:

$$w_{uit} = \begin{cases} 2 & \text{if } y_{uit} = 1 \\ 1 & \text{if } y_{uit} \neq 1 \wedge t \in L_u^+ \\ -1 & \text{if } y_{uit} \neq 1 \wedge t \in L_u^- \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

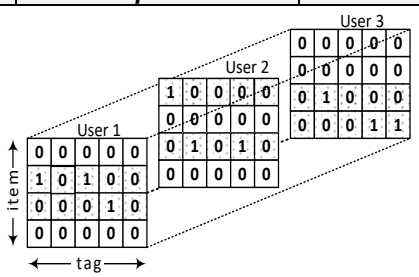
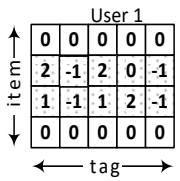
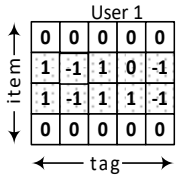
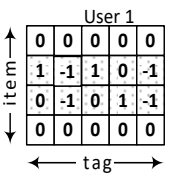
	Weighting Scheme		
	Quaternary-Tag-Preference	***Ternary-Positive-Tag-Preference***	***Ternary-Negative-Tag-Preference***
Tensor Model $\mathcal{Y} \in \mathbb{R}^{3\times4\times5}$			
Mode-1 $Y_{(1)} \in \mathbb{R}^{3\times20}$ **and Mode-3** $Y_{(3)} \in \mathbb{R}^{5\times12}$ **matricization**	$Y_{(1)} = \begin{bmatrix} 0&1&0&0&0&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0 \\ 1&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&0&1 \end{bmatrix}$ $Y_{(3)} = \begin{bmatrix} 0&1&0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&1&0 \\ 0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&1&0&0&0&0&1 \\ 0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix}$		
User Latent $A \in \mathbb{R}^{3\times2}$ **and** ***Tag Latent*** $B \in \mathbb{R}^{5\times2}$ ***Feature*** **matrices**		β_1 β_2 u_1 0.3943 1.0000 u_2 0.7835 0.0000 u_3 0.4803 0.0000	φ_1 φ_2 t_1 0.0000 0.7071 t_2 0.7013 0.0000 t_3 0.0000 0.7071 t_4 0.7106 0.0000 t_5 0.0570 0.0000
User Tag Usage Likeliness **Matrix** $L \in \mathbb{R}^{3\times5}$		t_1 t_2 t_3 t_4 t_5 u_1 0.7071 0.2765 0.7071 0.2802 0.0225 u_2 0.0000 0.5495 0.0000 0.5568 0.0447 u_3 0.0000 0.3368 0.0000 0.3413 0.0274	
User Positive L_u^+ **and** ***Negative*** L_u^- ***Tag Preference*** **Sets where** $v = 2$	$L_{u_1}^+ = \{t_1, t_3\}, L_{u_2}^+ = \{t_2, t_4\}, L_{u_3}^+ = \{t_2, t_4\}$ $L_{u_1}^- = \{t_2, t_5\}, L_{u_2}^- = \{t_1, t_3\}, L_{u_3}^- = \{t_1, t_3\}$		
Weighted Tensor $\mathcal{W} \in \mathbb{R}^{3\times4\times5}$ **displaying the matrix of** u_1			

Figure 3. Examples showing the process of forming the user tag preference and constructing the weighted $\mathcal{W}$ of u_1

Ternary-Positive-Tag-Preference **weighting scheme:** This scheme generates three ordinal weighting values of $\{1,0,-1\}$. The "1" value represents the observed entries or unobserved entries that belong to L_u^+, whereas "-1" value represents the unobserved entries that belong to L_u^-. The rest of the entries are denoted as "0". This scheme gives equal reward to both the observed entry of $\mathcal{Y}$ and the entry of unobserved L_u^+. It assumes that the user has the same interest to use the tags to annotate items though the latter is not explicitly revealed. The rule for generating the values of $\mathcal{W}$ entries is formulated as follows:

$$w_{uit} = \begin{cases} 1 & \text{if } y_{uit}=1 \ \vee \ (y_{uit}\neq1 \ \wedge \ t\in L_u^+) \\ -1 & \text{if } y_{uit}\neq1 \ \wedge \ t\in L_u^- \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

Ternary-Negative-Tag-Preference **weighting scheme:** This scheme generates three ordinal weighting values of $\{1,0,-1\}$. The "1" value represents the observed entries, whereas "-1" value represents the unobserved entries that belong to L_u^-. The rest of the entries are denoted as "0". This scheme disregards the entries of unobserved L_u^+. It assumes that though the user has a positive preference for the list of tags, yet the fact that it was not revealed

simply means that the user does not want to generally use them to annotate all items. In other words, the observed entries of $\mathcal{Y}$ themselves sufficiently represent the user's positive tag preference. The rule for generating the values of $\mathcal{W}$ entries is:

$$w_{uit} = \begin{cases} 1 & \text{if } y_{uit} = 1 \\ -1 & \text{if } y_{uit} \neq 1 \wedge t \in L_u^- \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

3.3 Weighted-based Learning

Weighted-based learning is the stage where the learning process deals with the sparsity problem by focusing only on the known entries in the tensor model [18]. Here, we named our proposed weighted-learning method as *WRank* (Weighted Ranking).

3.3.1 Objective Function and Factorization Technique

The learning model of a weighted-learning approach is optimized as a weighted loss function. The objective function is formulated as:

$$L(\Theta) = \sum_{u=1}^{Q} \sum_{i=1}^{R} \sum_{t=1}^{S} w_{uit} \cdot \ell(y_{uit}, \hat{y}_{uit}) \quad (8)$$

where $\ell(y_{uit}, \hat{y}_{uit})$ is a loss function while y_{uit} is the valuation of tagging activity denoted as either 1 or 0. Whereas $\hat{y}_{uit}$ is the prediction score of user u for annotating item i using tag t. Meanwhile, w_{uit} is the weighted reward or penalty value assigned to y_{uit}.

We use CP tensor-based factorization technique to infer the latent relationship within the tensor model $\mathcal{Y}$. This technique requires less memory and time consumptions than Tucker [23]. CP factorizes tensor $\mathcal{Y} \in \mathbb{R}^{Q \times R \times S}$ into a sum of rank-one factors of $m_f^{(1)} \in \mathbb{R}^Q$, $m_f^{(2)} \in \mathbb{R}^R$, and $m_f^{(3)} \in \mathbb{R}^S$ for $f = 1, \ldots, F$. The prediction score of user u to annotate item i using tag t is calculated as:

$$\hat{y}_{uit} = \sum_{f=1}^{F} \prod_{n=1}^{3} m_{c_n f}^{(n)} = [\![M^{(1)}, M^{(2)}, M^{(3)}]\!] \quad (9)$$

where $c \in \{u, i, t\}$ and F is the size of the latent factors. Note that for brevity, we later use w, y and $\hat{y}$ to denote w_{uit}, $\hat{y}_{uit}$ and $\hat{y}_{uit}$, respectively.

3.3.2 Generating Latent Factors

Generating the latent factors $M^{(1)}$, $M^{(2)}$, and $M^{(3)}$, are conducted by optimizing the objective function in Eq. (8). The gradients of weighted loss function are formulated as follows:

$$\frac{\partial L}{\partial \theta} = \frac{\partial}{\partial \theta} \left(\sum_{u=1}^{Q} \sum_{i=1}^{R} \sum_{t=1}^{S} w \cdot \ell(y, \hat{y}) \right) \quad (10)$$

$$\frac{\partial L}{\partial \theta} = \sum_{u=1}^{Q} \sum_{i=1}^{R} \sum_{t=1}^{S} w \frac{\partial \ell(y, \hat{y})}{\partial \theta} \quad (11)$$

This paper uses the quadratic loss as it is the most common optimization criterion for a learning model populated from the *Boolean* scheme [23]. Nevertheless, the gradient of Eq. (11) can accommodate any loss function. The formulation is:

$$\ell^q(y, \hat{y}) = \frac{1}{2}(y - \hat{y})^2 \quad (12)$$

The quadratic loss gradient can then be formulated as:

$$\frac{\partial \ell^q(y, \hat{y})}{\partial m_{c_n}^{(n)}} = w \frac{\partial ((y - \hat{y})^2)}{\partial m_{c_n}^{(n)}} \quad (13)$$

$$\frac{\partial \ell^q(y, \hat{y})}{\partial m_{c_n}^{(n)}} = 2 \cdot w \cdot (y - \hat{y}) \frac{\partial (y - \hat{y})}{\partial m_{c_n}^{(n)}} \quad (14)$$

$$\frac{\partial \ell^q(y, \hat{y})}{\partial m_{c_n}^{(n)}} = -2 \cdot w \cdot (y - \hat{y}) \prod_{\substack{d=1 \\ d \neq n}}^{3} m_{c_d f}^{(d)} \quad (15)$$

where $c \in \{u, i, t\}$ and $n \in \{1,2,3\}$. The learning algorithm of *WRank* is presented in Fig. 4.

Algorithm: *WRank* Learning
Input: Initial tensor $\mathcal{Y} \in \mathbb{R}^{Q \times R \times S}$, Weighted tensor $\mathcal{W} \in \mathbb{R}^{Q \times R \times S}$, Column size of latent factor matrix F, Maximal iteration $iterMax$
Output: Latent factors $M^{(1)}$, $M^{(2)}$, $M^{(3)}$
Initialize $M^{(1)^{(0)}} \in \mathbb{R}^{Q \times F}$, $M^{(2)^{(0)}} \in \mathbb{R}^{R \times F}$, $M^{(3)^{(0)}} \in \mathbb{R}^{S \times F}$
$h = 0$
$c \in \{u, i, t\}$
repeat
  for $f \leftarrow 1$ *to* F **do**
    for $n \leftarrow 1$ *to* 3 **do**
      $\frac{\partial L}{\partial m_{c_n f}^{(n)}} \leftarrow \sum_{u=1}^{Q} \sum_{i=1}^{R} \sum_{t=1}^{S} w \frac{\partial \ell(y, \hat{y})}{\partial m_{c_n f}^{(n)}}$
  $++h$
until $h \geq iterMax$

Figure 4. The *WRank* learning algorithm

3.4 Recommendation Generation

The list of item recommendations are generated based on the maximum value of $\hat{y}_{uit}$, calculated according to Eq. (9).

4. Experiment Setup

4.1 Datasets and Evaluation Method

For the experiments, we use two real-world tagging datasets commonly used in recommendation researches, i.e., LastFM [24] and CiteULike retrieved

from the CiteULike corpus (http://static.citeulike.org/data/current.bz2).

We filter the datasets using p-core technique [25] to remove the noise and reduce the data sparsity [6, 26]. This paper implements a variety of p-cores to avoid bias and maximizing the likelihood to produce stable results in the experiments [27]. The details of the datasets after the implementation of various p-cores implementation are shown in Table 1.

The 5-fold cross-validation technique is implemented to randomly divide each core of each dataset into an 80% training set D_{train} and a 20% test set D_{test} according to the number of posts. There is no overlap post between D_{train} and D_{test} such that no triplets of a user-item set exist in D_{train} when there are triplets of $(u, i, *)$ present in D_{test}. D_{train} is used as the learning model for generating the list of top-N item predictions for the users in D_{test}. The recommendation performances are evaluated and reported over the average scores of all folds using AP and NDCG evaluation metrics.

4.2 Variant of The Proposed *WRank* Method

We generate three variants of *WRank* to evaluate the performance of the three proposed weighting schemes. They are labelled as *WRank-Q*, *WRank-TP*, and *WRank-TN* to respectively following the implementation of the *Quaternary-Tag-Preference*, *Ternary-Positive-Tag-Preference*, and *Ternary-Negative-Tag-Preference* weighting schemes to *WRank*. Table 2 lists the non-zero entries densities of $\mathcal{Y}$ and $\mathcal{W}$ of *WRank-Q*, *WRank-TP* and *WRank-TN* where $v = 10$. Note that the density of *WRank-Q* is always equivalent to that of *WRank-TP*. Meanwhile, the density of *WRank-TN* is always less than its counterparts.

The factor matrices of the model are randomly initialized following the common approach [28]. The non-linear conjugate gradient (NCG) method of Poblano Toolbox [29] is used as the optimization method. We empirically tuned the parameters as $tolerance = 1e^{-05}$ and $regularization = 0$.

Table 1. The details of datasets

Data-set	p-core	#user (Q)	#item (R)	#tag (S)	Observed Entries ($\lvert A_{ob} \rvert$)	Density (%)
Last-FM	10	867	1,715	1,423	99,211	0.0047
	15	703	1,018	1,063	76,808	0.0100
	20	601	681	838	61,739	0.0180
Cite-ULike	10	1,129	548	2,403	17,161	0.0012
	15	721	203	1,334	8,099	0.0042
	20	529	89	844	4,254	0.0100

Table 2. Densities comparison of $\mathcal{Y}$ and $\mathcal{W}$ ($v = 10$)

Data-set	Tensor	Density of Non-zero entries on D_{train} (%)		
		10-core	15-core	20-core
Last-FM	$\mathcal{Y}$	0.0042	0.0092	0.0163
	$\mathcal{W}$ of *WRank-Q*	0.0302	0.0640	0.1122
	$\mathcal{W}$ of *WRank-TP*	0.0302	0.0640	0.1122
	$\mathcal{W}$ of *WRank-TN*	0.0183	0.0389	0.0686
Cite-ULike	$\mathcal{Y}$	0.0010	0.0037	0.0095
	$\mathcal{W}$ of *WRank-Q*	0.0087	0.0309	0.0808
	$\mathcal{W}$ of *WRank-TP*	0.0087	0.0309	0.0808
	$\mathcal{W}$ of *WRank-TN*	0.0049	0.0182	0.0480

5. Results and Discussion

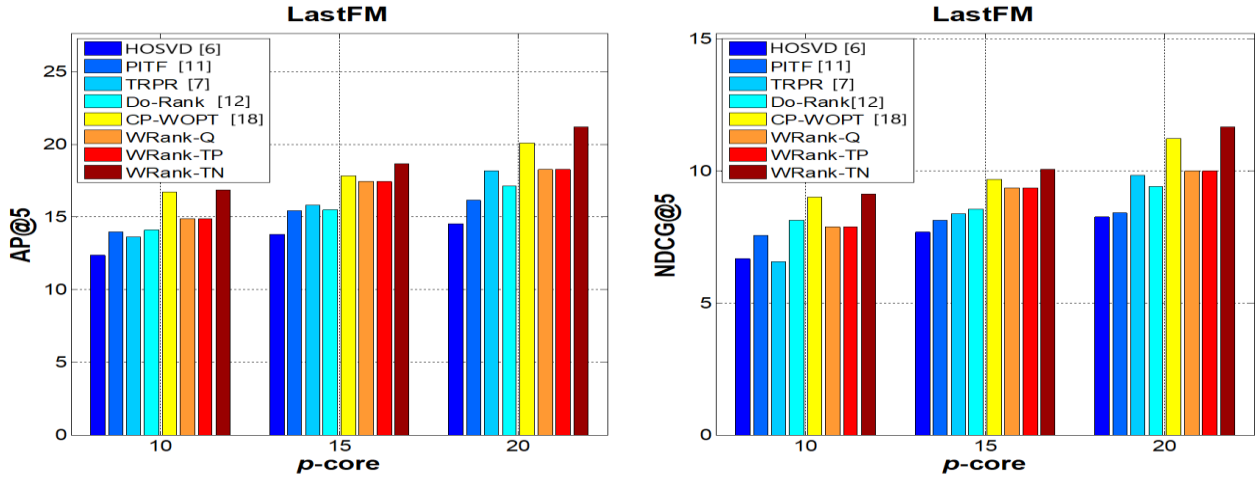
5.1 Performance Comparison

Fig. 5 and Fig. 6 respectively show the performance comparisons on the LastFM and CiteULike datasets. Recall that our proposed methods are benchmarked with the following methods: (i) *HOSVD* [6] where the parameters are set as $tolerance = 1e^{-04}$ and $regularization = 0$; (ii) *PITF* [11] where the parameters are set as $learning\ rate = 0.01$ and $regularization = 5e^{-05}$; (iii) *TRPR* [7] where the parameters are set as $tolerance = 1e^{-04}$, $regularization = 0$ and $voc\_size = 20$; (iv) *Do-Rank* [12] where the parameters are set as $learning\ rate = 0.01$ and $regularization = 1e^{-05}$, and (v) *CP-WOPT* [18] where the parameters are set as $tolerance = 1e^{-05}$ and $regularization = 0$. Note that we set $F = 128$ for all methods, including ours. We discuss four observations based on those results.

First, the proposed *WRank-TN* outperforms *WRank-Q* and *WRank-TP*, in terms of AP and NDCG on all datasets and p-cores. Interestingly, the performance comparison results contradict the density comparison results listed in Table 2, i.e., the densities of *WRank-TN* is the least yet it performs the best. These results indicate that the weighting schemes of *WRank-Q* and *WRank-TP* include unnecessary relationships, impacting the recommendation quality as well as causing extra computation cost. On the other hand, the performance equality of *WRank-Q* and *WRank-TP*, that comprehends their equal densities, indicates that the variation of entry values used to label the observed data and unobserved positive tag preference set has no impact on performances. Moreover, including the entries of unobserved positive tag preference set in

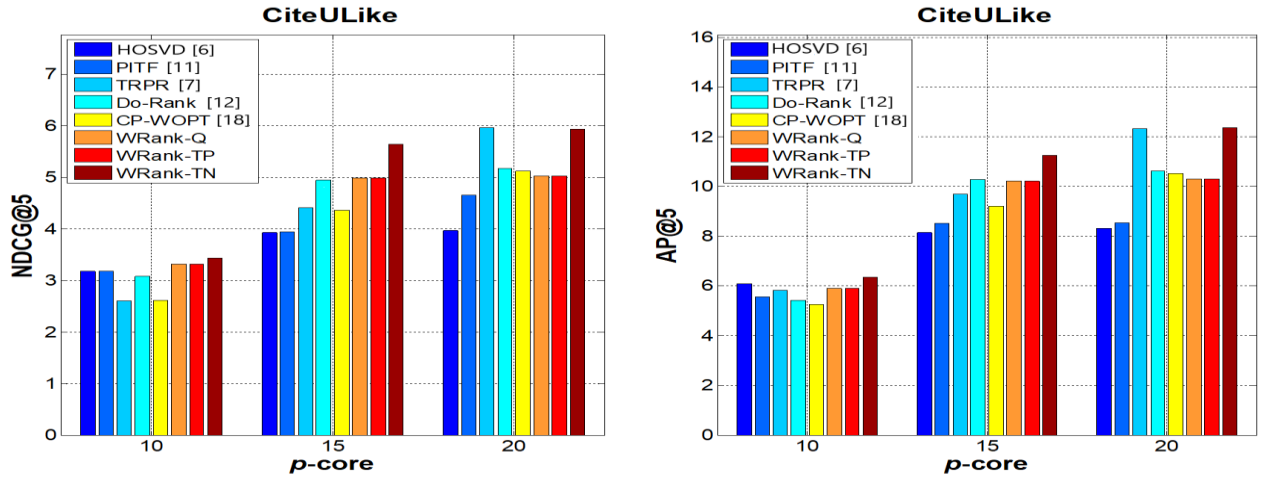
the weighted-learning approach is impractical since it deteriorates the recommendation quality. In this case, we can conclude that the user tag preference is best represented as combination entries of the observed data and unobserved negative tag preference set only. Henceforward, we use *WRank-TN* in the discussion comparison with the benchmarking methods.

Second, *WRank-TN* outperforms *HOSVD*, *TRPR*, *PITF*, and *Do-Rank*. *HOSVD* and *TRPR* underperformances are because they directly learn from a model $\mathcal{Y}$ that does not exploit the data sparsity. Though *TRPR* implements a probabilistic ranking approach, yet the procedure is executed following, and not during, the learning process. Meanwhile, *PITF* and *Do-Rank* respectively implement the *set-based* and *UTS* schemes as a weighted-learning approach, exploiting the data sparsity, for building $\mathcal{Y}$. However, the exploitation is conducted before, instead of during, the learning process. These results ascertain that the implementation of the weighted-learning approach that presenting $\mathcal{Y}$ as binary data and uses granular $\mathcal{W}$ results in better recommendation quality compared to those that present $\mathcal{Y}$ as granular data but with no usage of $\mathcal{W}$.



(a) AP@5 (b) NDCG@5

Figure 5. NDCG@5 and AP@5 on LastFM dataset



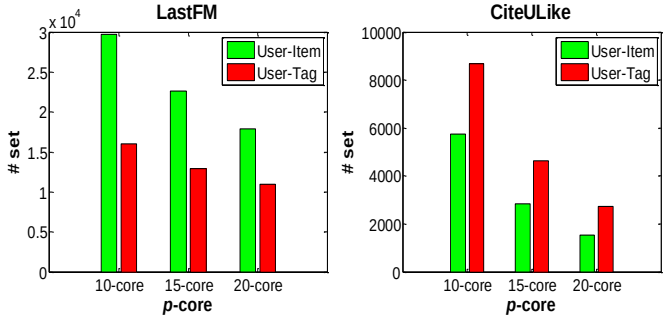
(a) AP@5 (b) NDCG@5

Figure 6. NDCG@5 and AP@5 on CiteULike dataset

Third, *WRank-TN* outperforms *CP-WOPT* in terms of both AP and NDCG on all datasets and *p*-cores. Recall that *WRank-TN* implements a weighting scheme that takes into account the user tag preference. In this case, *WRank-TN* can assign reward or penalty to both the observed and unobserved entries of $\mathcal{Y}$ during the learning process, i.e. leveraging the concept of the weighted-learning approach. In this way, the collaborative method is brined more effectively into the recommendation. On the other hand, *CP-WOPT* uses a *Boolean* weighting scheme that only focuses on rewarding the observed entries of the tensor model and neglects the fact that users may have different user tag preferences. In

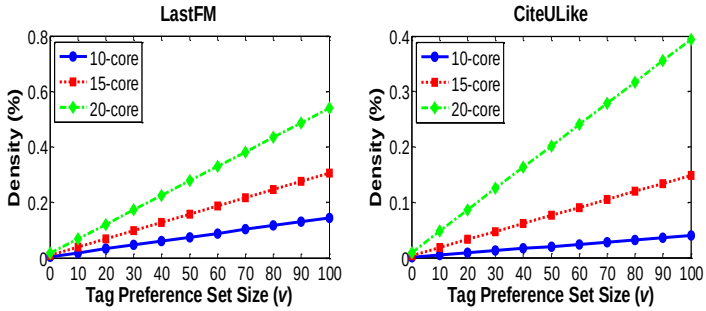
short, this observation confirms that the users tag preferences that taking into account both the observed and unobserved entries in determining the weighting values benefits the learning process and enhances the recommendation quality.

We can also observe that *WRank-TN* has more dominant outperformance over *CP-WOPT* on the CiteULike dataset compared to the LastFM dataset. The reason for this can be revealed from the users' tagging behaviours of the two datasets. Recall that tag-based systems permit their users to use various tags to annotate an item as well as to use a tag to annotate numerous items. Thus, we can observe the users' tagging behaviours through the dominancy of either the observed user-item set or the user-tag set in the tagging data [30]. The user-item set is dominant when the users are typically using very few tags when annotating items. Contrariwise, the user-tag set is dominant when users are using plenty of tags for annotating items. Fig. 7 shows that the LastFM dataset has a less dominant user-tag set than that of user-item. In contrast, the user-tag set of CiteULike dataset is more dominant than its user-item set. This points out that the weighting scheme characteristic of *WRank-TN* works in its favour to a dataset with a dominant user-tag set.



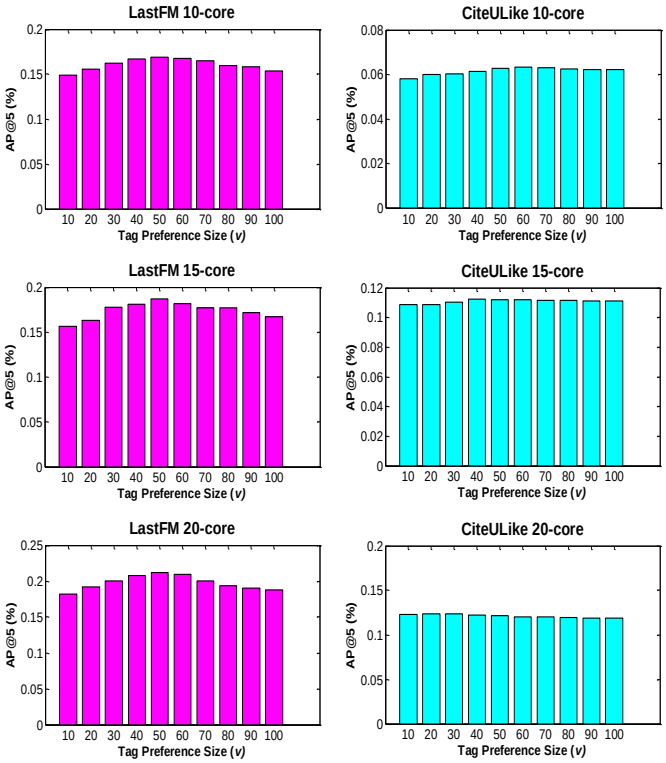
(a) (b)

Figure 7. The comparison of user-item and user-tag sets: (a) LastFM and (b) CiteULike



(a) (b)

Figure 8. The densities of tensor $\mathcal{W}$: (a) LastFM and (b) CiteULike



(a) (b)

Figure 9. The impact of the size of the user tag preference set: (a) LastFM and (b) CiteULike

5.2 Sensitivity Analysis of *WRank*

The sensitivities of *WRank-TN* are examined based on the following terms: (i) the impact of the size of p-core towards its quality of performance, and (ii) the impact of the size of user tag preference, v, towards the density of $\mathcal{W}$ and the quality of performance.

From Fig. 5 and Fig. 6, we can observe that the size of the p-core is linear to the performances of *WRank-TN* on both AP and NDCG. In other words, the bigger the p-core size, the higher the performance quality is. This result indicates the robustness of *WRank-TN* over the p-core refinement procedure.

Fig. 8 shows that the density of $\mathcal{W}$ is linear to v on the LastFM and CiteULike datasets. We can observe that a denser $\mathcal{W}$ can always be constructed out of a sparse $\mathcal{Y}$. This observation confirms that the implementation of $\mathcal{W}$ can exploit the sparsity within $\mathcal{Y}$. Meanwhile, Fig. 9 shows that *WRank-TN* performs the best when v is between 20 to 60 and inclines afterwards. This result suggests that the overuse of tags deteriorates the preference of users.

6. Conclusion

We have presented a new weighted-learning approach for exploiting data sparsity in the tag-based item recommendation system. We propose a technique to generate the user tag preference representation that leads to the formulation of three proposed weighting schemes, i.e., *Quaternary-Tag-Preference*, *Ternary-Positive-Tag-Preference*, and *Ternary-Negative-Tag-Preference*. We also proposed *WRank*, a weighted-learning item recommendation method to implement our proposed weighting schemes. Our experimental results show that *WRank-TN*, i.e., *WRank* that implements the *Ternary-Negative-Tag-Preference* weighting scheme, acquires the best performance in comparison to other proposed weighting schemes. This finding points out that the observed entries of $\mathcal{Y}$ themselves sufficiently represent the user's positive tag preference and that the user's negative tag preference is efficiently revealed from the unobserved entries of $\mathcal{Y}$. Results correspondingly show that the proposed *WRank-TN* method outcompetes the benchmarking methods, i.e., *HOSVD*, *TRPR*, *PITF*, *Do-Rank*, *CP-WOPT*. These outcomes confirm that the users' tag preferences that take into account both the observed and unobserved entries in determining the weighting values benefits the learning process and enhances the recommendation quality. Additionally, the empirical analysis also shows that *WRank-TN* can solve the data sparsity problem and is robust over p-core refinement procedure.

References

[1] T. Bogers, "Tag-based recommendation", in *Social Information Access*, Springer, Cham, pp. 441-479, 2018.

[2] C. C. Aggarwal, *Recommender Systems: The Textbook*, Springer International Publishing, Switzerland, 2016.

[3] E. El Sayed, A. El Korany, A. Salah, and H. Hefny, "Exploiting Social Annotations for Personalizing Retrieval", *International Journal of Intelligent Engineering and Systems*, Vol. 10, No. 6, pp. 192-202, 2017.

[4] E. E. S. Mahmoud and A. S. Mohamed, "An Ontology Based Framework for Automatic Web Resources Identification", *International Journal of Intelligent Engineering and Systems*, Vol. 13, No. 2, pp. 299-306, 2020.

[5] E. Frolov and I. Oseledets, "Tensor methods and recommender systems", *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Vol. 7, No. 3, p. e1201, 2017.

[6] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos, "A Unified Framework for Providing Recommendations in Social Tagging Systems Based on Ternary Semantic Analysis", *IEEE Transactions on Knowledge and Data Engineering*, Vol. 22, No. 2, pp. 179-192, 2010.

[7] N. Ifada and R. Nayak, "A Two-Stage Item Recommendation Method Using Probabilistic Ranking with Reconstructed Tensor Model", in *User Modeling, Adaptation, and Personalization*, Springer, Cham, pp. 98-110, 2014.

[8] A. Klašnja-Milićević, M. Ivanović, B. Vesin, and Z. Budimac, "Enhancing e-learning systems with personalized recommendation based on collaborative tagging techniques", *Applied Intelligence*, Vol. 48, No. 6, pp. 1519-1535, 2018.

[9] Y. Yang, L. Han, Z. Gou, B. Duan, J. Zhu, and H. Yan, "Tagrec-CMTF: Coupled matrix and tensor factorization for tag recommendation", *IEEE Access*, Vol. 6, pp. 64142-64152, 2018.

[10] X. Tang, Y. Xu, and S. Geva, "Factorization-based primary dimension modelling for multidimensional data in recommender systems", *International Journal of Machine Learning and Cybernetics*, Vol. 10, No. 8, pp. 2209-2228, 2019.

[11] S. Rendle and L. Schmidt-Thieme, "Pairwise Interaction Tensor Factorization for Personalized Tag Recommendation," In: *Proc. of The 3rd ACM International Conference on Web Search and Data Mining*, New York, USA, pp. 81-90, 2010.

[12] N. Ifada and R. Nayak, "Do-Rank: DCG Optimization for Learning-to-Rank in Tag-Based Item Recommendation Systems", in *Advances in Knowledge Discovery and Data Mining*, Springer International Publishing, pp. 510-521, 2015.

[13] H. Lim and H. J. Kim, "Item recommendation using tag emotion in social cataloging services", *Expert Systems with Applications*, Vol. 89, pp. 179-187, 2017.

[14] H. Lim and H. J. Kim, "Tensor-based tag emotion aware recommendation with probabilistic ranking", *KSII Transactions on Internet and Information Systems (TIIS)*, Vol. 13, No. 12, pp. 5826-5841, 2019.

[15] H. Zhang, Q. Hong, X. Shi, and J. He, "A Social Tagging Recommendation Model Based on Improved Artificial Fish Swarm Algorithm and Tensor Decomposition", in *Advances in Intelligent Systems and Computing*, Vol. 733, Springer, Cham, pp. 3-13, 2018.

[16] K. Hosono, S. Ono, and T. Miyata, "Weighted tensor nuclear norm minimization for color image denoising," In: *Proc. of International Conference on Image Processing (ICIP)*, Phoenix, AZ, USA, pp. 3081-3085, 2016.

[17] Y. Mu, P. Wang, L. Lu, X. Zhang, and L. Qi, "Weighted tensor nuclear norm minimization for tensor completion using tensor-SVD", *Pattern Recognition Letters*, Vol. 130, pp. 4-11, 2020.

[18] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, "Scalable Tensor Factorizations for Incomplete Data", *Chemometrics and Intelligent Laboratory Systems*, Vol. 106, No. 1, pp. 41-56, 2011.

[19] M. Filipović and A. Jukić, "Tucker factorization with missing data with application to low-n-rank tensor completion", *Multidimensional systems and signal processing*, Vol. 26, No. 3, pp. 677–692, 2015.

[20] L. Yuan, Q. Zhao, and J. Cao, "Completion of high order tensor data with missing entries via tensor-train decomposition," In: *Proc. of International Conference on Neural Information Processing*, Guangzhou, China, pp. 222-229, 2017.

[21] M. Iwata, L. Yuan, Q. Zhao, Y. Tabei, F. Berenger, R. Sawada, S. Akiyoshi, *et al.*, "Predicting drug-induced transcriptome responses of a wide range of human cell lines by a novel tensor-train decomposition algorithm", *Bioinformatics*, Vol. 35, No. 14, pp. i191-i199, 2019.

[22] J. Kim, Y. He, and H. Park, "Algorithms for Nonnegative Matrix and Tensor Factorizations: A Unified View based on Block Coordinate Descent Framework", *Journal of Global Optimization*, Vol. 58, No. 2, pp. 285-319, 2014.

[23] T. Kolda and B. Bader, "Tensor Decompositions and Applications", *SIAM Review*, Vol. 51, No. 3, pp. 455-500, 2009.

[24] I. Cantador, P. Brusilovsky, and T. Kuflik, "Second Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec2011)," In: *Proc. of The 5th ACM Conference on Recommender Systems*, Chicago, Illinois, USA, pp. 387-388, 2011.

[25] V. Batagelj and M. Zaveršnik, "Generalized Cores", *arXiv preprint cs/0202039*, 2002.

[26] D. Rafailidis and P. Daras, "The TFC Model: Tensor Factorization and Tag Clustering for Item Recommendation in Social Tagging Systems", *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, Vol. 43, No. 3, pp. 673-688, 2013.

[27] S. Doerfel and R. Jäschke, "An Analysis of Tag-recommender Evaluation Procedures," In: *Proc. of The 7th ACM Conference on Recommender Systems*, pp. 343-346, 2013.

[28] S. Balakrishnan and S. Chopra, "Collaborative Ranking," In: *Proc. of The 5th ACM International Conference on Web Search and Data Mining*, Seattle, Washington, USA, pp. 143-152, 2012.

[29] D. M. Dunlavy, T. G. Kolda, and E. Acar, *Poblano v1.0: A Matlab Toolbox for Gradient-Based Optimization*, Sandia National Laboratories, Albuquerque, NM and Livermore, CA, SAND2010-1422, 2010.

[30] N. Ifada and R. Nayak, "An Efficient Tagging Data Interpretation and Representation Scheme for Item Recommendation," In: *Proc. of The 12th Australasian Data Mining Conference*, Brisbane, Australia, pp. 205-215, 2014.