Tensor-based Item Recommendation using Probabilistic Ranking in Social Tagging Systems

Noor Ifada*, Richi Nayak
Queensland University of Technology
Gardens Point Campus, 2 George St, Brisbane, Queensland 4000, Australia
noor.ifada@{if.trunojoyo.ac.id, qut.edu.au}, r.nayak@qut.edu.au

ABSTRACT

A common problem with the use of tensor modeling in generating quality recommendations for large datasets is scalability. In this paper, we propose the Tensor-based Recommendation using Probabilistic Ranking method that generates the reconstructed tensor using block-striped parallel matrix multiplication and then probabilistically calculates the preferences of user to rank the recommended items. Empirical analysis on two real-world datasets shows that the proposed method is scalable for large tensor datasets and is able to outperform the benchmarking methods in terms of accuracy.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information Filtering

Keywords

Tensor Model; Item Recommendation; Probabilistic Ranking; Folksonomy; Social Tagging Systems

1. INTRODUCTION

The Social Tagging Systems (STS) play significant role in helping users to manage their resources (called as items) online by adding tags [1]. Tags, which are reusable and shareable, reveal user interest and allow users to recognize items. This additional source of information helps online systems for building improved user profiles that can be used in recommendation [2, 3]. This type of system called as the tag-based recommender system uses Folksonomy, a triplet called as tag assignment, for representing the correlation between users, items, and tags. The success of a tag-based recommender system depends on how the correlation in Folksonomy will be exploited [4].

A two-dimensional user profiling approach, modeling the users and items correlation, cannot be directly employed to build a tag-based recommender system. This approach cannot efficiently capture the three-dimensional characteristic and, fails to model the many-to-many relationships that exist among these dimensions [5]. Researchers have solved this problem by projecting the <user, item, tag> correlation into three two-dimensional matrices, <user, item>, <user, tag>, and <item, tag> [4]. This approach, however, is unable to capture the total interaction among dimensions and cannot expose their latent relationship. It can result in poorer recommendation quality [5, 6].

Since folksonomy is a multi-dimensional data, user profiles can be naturally modeled with higher-order data models. Tensor modeling is a well-known approach to represent and analyze latent relationships inherent in the multi-dimensions data [7]. The approach consists of three steps: (1) tensor construction for modeling multi-dimension data; (2) tensor decomposition to derive latent relationships inherent in the data; and (3) tensor reconstruction to reveal the latent relationships between dimensions of tensor model.

Using tensor modeling for generating recommendation is challenging due to the following reasons:

- **Reconstructing large size tensor.** Tensor reconstruction is computed by multiplying all decomposed element. This process is memory expensive and, therefore, reconstructing large size tensors (for instance, more than the size of $1000 \times 1000 \times 1000$ for a 3-D tensor model) is infeasible [5, 6, 8-10]. Several memory efficient methods for enabling decomposition process have been proposed [11] and they fulfill the purpose of many applications that do not need a tensor to be reconstructed. The decomposed elements are sufficient for their purpose; however, the tensor-based recommendation methods, as proposed in this paper, require the tensor to be reconstructed for identifying new entries and, that so, in large sizes.
- **Utilizing reconstructed tensor to generate recommendation.** Existing approaches assume that tag assignment value in the reconstructed tensor represents the level of user preference for an item based on the tag, and generate the recommendations based on the maximum values of tag assignments in each user-item combinations [5, 9]. However, these approaches ignore the user's past tagging activities that have been found most influencive in forming user likelihood for the recommended items [4].

In this paper we address the aforementioned challenges of scalability and accuracy, and propose a novel Tensor-based Recommendation using Probabilistic Ranking (TRPR) method. The block-striped parallel matrix multiplication is proposed for combining the decomposed elements of the tensor model to enable a scalable tensor reconstruction. The list of candidate items and tag preferences for each user are then generated. The probabilistic approach is proposed for calculating the preferences of users to

* Noor Ifada is currently on leave from the Informatics Engineering Department, University of Trunojoyo Madura, Indonesia.

WWW'14 Companion, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2745-9/14/04.
http://dx.doi.org/10.1145/2567948.2579243

rank the recommended items. The proposed TRPR method is evaluated on two real-world STS datasets, namely Delicious and LastFM. Experiments have been conducted to find the effectiveness of the method over the benchmarking methods. Empirical analysis shows that the proposed method is scalable and is able to outperform the benchmarking methods in accuracy of recommendation.

The contribution of this paper can be summarized as: (1) a novel method of scalable tensor reconstruction, and (2) an effective tag-based recommender system using tensor modeling and probabilistic approach.

2. RELATED WORK

Tensor modeling has been used to build recommender systems for offering various types of outputs. In the STS, researchers have used tensor models to generate tags recommendations [5, 8, 12, 13] and items recommendations [5, 6, 9] based on the usage of tags. The task of recommending tags to a user for an item differs from the task of recommending an item to a user. Tag recommendation is generated using the user-item predefined combination, while the item recommendation is generated with only the user specified [14]. Therefore, the recommender system for each type should be developed using different approach.

Scalability is a common problem in generating recommendations for large datasets using tensor modeling. Rendle & Schmidt-Thieme applied the PITF technique [13] which models the pairwise interaction between users, items, and tags. The method able to perform better in terms of runtime and recommendation quality than other factorization techniques. Nevertheless, this approach is not applicable to this study since our focus is to generate item recommendation while their work is designed for generating tag recommendation. Kolda & Sun [11] solve the construction and decomposition memory problems by storing only the non-zeros values and then implementing a memory efficient method. However, the tensor-based recommendation methods, as proposed in this paper, require the tensor to be reconstructed by multiplying all decomposed elements for identifying new entries. Consequently, the process of reconstructing, especially when the data is large, is very expensive. This has not been properly addressed yet.

The current item recommendation methods [5, 6, 9] either use the decomposition factors or the reconstructed tensor directly for generating the recommendations. These approaches disregard the previous activities [15] of items and tags for the users, and solely use the final result of tensor model. This affects the recommendation quality. The method by Symeonidis et al [5] comes closest to our work. The method builds tensor model from a relatively small size data ($105 \times 246 \times 591$ representing users, items, tags) and directly utilizes the reconstructed tensor after decomposition to generate item recommendation based on the maximum values of tensor elements. Although the method is promising compared to the non-tensor methods, it has two inherent limitations: (1) not scalable for reconstructing large tensor; and (2) uses the reconstructed tensor result directly to generate recommendations and disregards the user past tagging activities which possibly affect the recommendation quality. To the best of our knowledge, this paper is the first tensor-based study of tag-based item recommender system that proposes to improve the quality of recommendations after the candidate items is achieved from the reconstructed tensor.

3. TENSOR-BASED RECOMMENDATION USING PROBABILISTIC RANKING

3.1 Preliminaries

Let $U = \{u_1, u_2, u_3, \dots, u_{|U|}\}$ be the set of all users, $I = \{i_1, i_2, i_3, \dots, i_{|I|}\}$ be the set of all items and $T = \{t_1, t_2, t_3, \dots, t_{|T|}\}$ be the set of all tags. In STS, a vector of tag assignment, $a(u, i, t) \in A$ represents the tagging activity of user u who has tagged item i with tag t. For each tag assignment, possible value of v_A for $a(u, i, t)$ is {0, 1} where 1 indicates that $a(u, i, t)$ exists and 0 indicates otherwise. The post O_A denotes the set of all distinct user-item combinations in A as a user can tag an item with multiple values For each post, the possible value of v_{O_A} for $o_{a(u,i)}$ is {0, 1} where 1 indicates a posting activity for a user u who tagged item i with any tags and 0 indicates user u has not tagged item i. The value set $v_{O_A}$is used for ranking the recommendations, whereas, the value set v_A is used in generating candidate items as well as in ranking item recommendations.

3.2 Overview of the TRPR Method

Figure 1 illustrates the proposed Tensor-based Recommendation using Probabilistic Ranking (TRPR) method for generating item recommendation to users based on their tagging activities. The first step is building tensor model that includes construction of a third-order tensor from the tag assignment data A, factorization of the tensor using a decomposition method, and then reconstruction of the decomposed elements. Given U as User dimension, I as Item dimension and T as Tag dimension, a third-order tensor $\mathcal{Y} \in \mathbb{R}^{U \times I \times T}$ is represented. The tensor is populated with the value of tag assignment v_A. Accordingly, the next step is ranking the candidate items for recommendation. It includes the generation of list of candidate items and tag preferences from the reconstructed tensor, and then probabilistically calculating the preferences of users to rank the Top-N list of item recommendations.

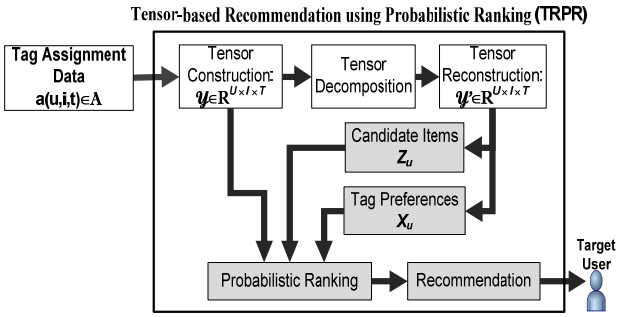

Figure 1: Overview of the Proposed TRPR Method

3.3 Initial Tensor Construction

From the A triplets dataset, an initial third-order tensor $\mathcal{Y} \in \mathbb{R}^{U \times I \times T}$ is constructed, where $|U|$, $|I|$, and $|T|$ are the number of users, items and tags respectively. Each element of tensor is given a binary initial value of v_A (which will change to a continuous value illustrating the importance of this triplet in the tensor model when the tensor will be reconstructed). As scalability and sparsity are the well-known issues in building tensor models [11], only the non-zeros values are uploaded when constructing the tensor. In this way, even the large tensor size can be uploaded as the majority of tensor entries are usually zero.

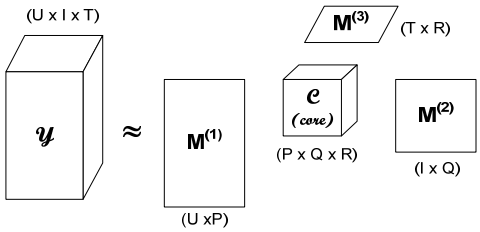

Figure 2: Tucker Decomposition for Third-Oder Tensor

3.4 Tensor Decomposition

Given tensor $\mathcal{Y}$, a decomposition technique is applied to derive latent relationship between users, items, and tags. In this paper, we explained the TRPR method by applying Tucker decomposition technique to derive latent relationships inherent in multi-dimensions of the tensor data. Tucker factorizes a third-order tensor $\mathcal{Y} \in \mathbb{R}^{U \times I \times T}$ into one core tensor $\mathcal{C} \in \mathbb{R}^{P \times Q \times R}$ and three factor matrices of $M^{(1)} \in \mathbb{R}^{U \times P}$, $M^{(2)} \in \mathbb{R}^{I \times Q}$, and $M^{(3)} \in \mathbb{R}^{T \times R}$ as follows:

$$\mathcal{Y} \approx \mathcal{C} \times_1 M^{(1)} \times_2 M^{(2)} \times_3 M^{(3)} \quad (1)$$

The core tensor $\mathcal{C}$ must satisfy:

$$\mathcal{C} = \mathcal{Y} \times_1 M^{(1)\prime} \times_2 M^{(2)\prime} \times_3 M^{(3)\prime} \quad (2)$$

Core tensor defines the interaction between the users, items and tags [7] while $|P|$, $|Q|$, and $|R|$ is the number of columns in the corresponding factor matrices. The Tucker decomposition for the third-order tensor $\mathcal{Y} \in \mathbb{R}^{U \times I \times T}$ is illustrated in Figure 2. A well-known Tucker-based techniques is Higher-Order Orthogonal Iteration (HOOI) [7] which generalizes Singular Value Decomposition (SVD) into a higher-order form by performing SVD on the matricized data for each dimension and then uses an iterative least square optimization approach to optimize the core tensor $\mathcal{C}$ [7].

3.5 Tensor Reconstruction

The core tensor and factor matrices are multiplied to generate the reconstructed tensor $\hat{\mathcal{Y}}$. The n-mode (matrix) product of a tensor $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_N}$ with a matrix V $\in \mathbb{R}^{J \times I_n}$ is denoted by $\mathcal{Y} \times_n V$ with a tensor of size $I_1 \times \ldots \times I_{n-1} \times J \times I_{n+1} \times \ldots \times I_N$. This means that each n-mode fiber is multiplied by matrix V. The n-mode (matrix) product of tensor $\mathcal{Y}$ with matrix V is equivalent to multiplying V by the appropriate matricization of tensor $\mathcal{Y}$ into matrix Y [7]:

$$\mathcal{B} = \mathcal{Y} \times_n V \iff B_{(n)} = VY_{(n)} \quad (3)$$

Due to the memory overflows during n-mode matrix product, reconstruction becomes non-scalable for large data. Given that decomposed elements consist of one core tensor and three factor matrices (for user, item, and tag), we split the reconstruction process into three parts. In the first two parts, we implement 1-mode and 2-mode (matrix) products to the core tensor (C) with first two factor matrices (R_U, R_I) sequentially and clear the memory after each multiplication once the results are saved. In the third part, we implement a 3-mode block (matrix) product when multiplying the last factor matrix (R_T) with the intermediate tensor result ($\widehat{Y_2}$) obtained from the previous step. The 3-mode block (matrix) product is based on the block parallel matrix multiplication operation [16].

Using the row wise block-striped matrix decomposition scheme, the multiplication task is split into N number of subtask multiplications of W_N (using block-strip of size $q = (|T|/N)$ rows in each iteration) with matrix $\widehat{Y_2}$ (the mode-3 matricization of tensor $\widehat{\mathcal{Y}_2}$). We obtain the complete reconstructed tensor $\hat{\mathcal{Y}}$ by combining all subtask results. This technique results in a memory efficient loop approach for scalable tensor reconstruction.

The reconstructed tensor $\hat{\mathcal{Y}}$ is able to reveal the latent relationship of users, items, and tags in the form of new entries, $O_{\hat{A}} - O_A$. The reconstructed tensor $\hat{\mathcal{Y}}$ includes a set of triplets $\hat{a}(u,i,t) \in \hat{\mathrm{A}}$, where $A \subset \hat{A}$. Each $\hat{a}(u,i,t) \in \hat{\mathrm{A}}$ will have a value $v_{\hat{A}}$ which represents the likeliness that user u will tag item i with tag t. Each post in $\hat{A}$ ($O_{\hat{A}}$) is assigned the binary value of $v_{O_{\hat{A}}}$.

Example: Tensor Modeling. Let us consider the toy problem in Table 1 and 2 to explain the processes in tensor modeling. Table 1 shows an initial third-order tensor $\mathcal{Y} \in \mathbb{R}^{3\times3\times5}$ populated with tag assignment data that contain a total of 3 users, 3 items, and 5 tags. Tensor is constructed using only non-zeros values, showing the existence of 8 tagging activities, represented as v_A, of user u who has tagged item i with tag t. Likewise, there are exist 5 posting activities, represented as v_{O_A}, of user u who tagged item i with any tags. This latter information is used in the probabilistic ranking process. Applying the decomposition technique to tensor $\mathcal{Y}$ will result in three factor matrices of $R_U \in \mathbb{R}^{3\times2}$, $R_I \in \mathbb{R}^{3\times2}$, and $R_T \in \mathbb{R}^{5\times2}$ and one core tensor $C \in \mathbb{R}^{2\times2\times2}$ when we choose 2 as the reduction size. Table 2 shows the reconstructed tensor $\hat{\mathcal{Y}}$ derived by multiplying the core tensor and factor matrices. For ease of illustration, we are only showing the top 11 entries out of a total 45 entries in the reconstructed tensor. It can be noted that tensor decomposition has recalculated the value ($v_{\hat{A}}$) for each existing entry as well as it has identified some new entries showing the latent relationships. As highlighted in Table 2, the reconstructed tensor $\hat{\mathcal{Y}}$ generates three new entries that are not present in the original tensor $\mathcal{Y}$ and, consequently, the items in these entries become the candidate items for recommendation.

Table 1: Toy Dataset for Initial Tensor $\mathcal{Y}$ Construction

User (u)	Item (i)	Tag (t)	Value of A (v_A)	Value of P_A (v_{P_A})
1	1	1	1	1
1	1	2	1	
1	1	3	1	
1	2	5	1	1
1	3	2	1	1
2	1	2	1	1
2	1	4	1	
3	1	2	1	1

Table 2: Reconstructed Tensor $\hat{\mathcal{Y}}$ from Toy Dataset

User (u)	Item (i)	Tag (t)	Value of $\hat{A}$ ($v_{\hat{A}}$)	Value of $P_{\hat{A}}$ ($v_{P_{\hat{A}}}$)
1	1	1	1.35	1
1	1	2	1.02	
1	1	3	0.89	
1	2	5	1.22	1
1	3	2	0.98	1
2	1	2	0.99	1
2	1	4	1.54	
2	**2**	**5**	**0.45**	1
2	**3**	**2**	**0.66**	1
3	1	2	1.28	1
3	**2**	**5**	**0.77**	1

3.6 Probabilistic Ranking and Recommendation Generation

Existing tensor-based recommendation approaches rank the recommendations based on the maximum value of $v_{\hat{A}}$ within every $O_{\hat{A}}$ for all new entries [5, 6, 9, 10]. These approaches give poor accuracy results as they disregard the previous activities [15] of each item and tag, and solely use the final result of tensor model.

We propose to rank the result of tensor modeling for generating the Top-N list of item recommendations to user u using Naïve Bayes [17]. Naïve Bayes generates a probabilistic model based on previously observed data. It is an efficient approach [18] if the problem of item recommendation can be treated as a classification problem.

Using the reconstructed tensor $\hat{\mathcal{Y}}$, for each user u, two lists are created: (1) a list of candidate items that the user u might be interested in, $Z_u = \{i_1, i_2, i_3, \dots, i_r\}$ from each $O_{\hat{A}}$ where $Z_u \subseteq I$; and (2) a list of tag preferences $X_u = \{t_1, t_2, t_3, \dots, t_s\}$ based on the maximum values of $v_{\hat{A}}$ which are sorted in descending order where $X_u \subseteq T$ of size s such that $|X_u| \leq s$. The probabilistic model calculates the posterior probability, $p(Z_u|X_u)$ using the Bayes' theorem for predicting the class candidate item Z_u that have the highest posterior probability conditioned on X_u. It is calculated as follows:

$$p(Z_u|X_u) = \frac{p(Z_u)p(X_u|Z_u)}{p(X_u)} \tag{4}$$

Where prior $p(Z_u)$ is the prior distributions of parameter set Z_u before X_u is observed; $p(X_u|Z_u)$ is the likelihood probability of observing the tag preference X_u given Z_u; and $p(X_u)$ is the probability of observing the instance X_u. We will use the posterior probability as the preference probability of user u to select candidate item Z_u by observing the tagging activity data of user u within A.

Using the assumption of multinomial event model distribution for the Naïve Bayes classifier, we calculate the posterior probability p_{u,i_r} of user u who has tag preference X_u on an item i_r by multiplying the prior probability of candidate item i_r, $p(Z_u = i_r)$, with the likelihood probability of tag preference t_c within the candidate item i_r, $p(t_c|Z_u = i_r)$:

$$\begin{aligned} p_{u,i_r} &= p(i_r|X_u) \\ &= p(Z_u = i_r) \prod_{c=1}^{s} p(t_c|Z_u = i_r)^{\left(\left(\sum_{i=1}^{|I|} v_{a(u,i_*,t_c)}\right)+1\right)} \end{aligned} \tag{5}$$

Where $v_{a(u,i_*,t_c)}$ denotes the value of assignment A for user u who has used tag preference t_c to tag any item i. The $p(Z_u = i_r)$ and $p(t_c|Z_u = i_r)$ are determined as:

$$p(Z_u = i_r) = \frac{\sum_{u=1}^{|U|} v_{O_{a(u_*,i_r)}}}{\sum_{i=1}^{|I|}\sum_{u=1}^{|U|} v_{O_{a(u_*,i_*)}}} \tag{6}$$

$$p(t_c|Z_u = i_r) = \frac{1 + \sum_{u=1}^{|U|} v_{a(u_*,i_r,t_c)}}{|T| + \sum_{u=1}^{|U|}\sum_{t=1}^{|T|} v_{a(u_*,i_r,t_*)}} \tag{7}$$

Where $v_{O_{a(u_*,i_r)}}$ and $v_{O_{a(u_*,i_*)}}$ denote the value of post O_A for any user u who has tagged candidate item i_r and any item i, respectively. The $v_{a(u_*,i_r,t_c)}$ and $v_{a(u_*,i_r,t_*)}$ denote the value of assignment A where the candidate item i_r has been tagged by any user u using tag preference t_c and any tag t, respectively. We apply the Laplacean estimate [18] as a smoothing method by adding one to $\sum_{i=1}^{|I|} v_{a(u,i_*,t_c)}$ in Equation (5) and $\sum_{u=1}^{|U|} v_{a(u_*,i_r,t_c)}$ in Equation (7) to avoid zero values.

When the probabilities of a target user for items are calculated, the list of recommended items are sorted in descending order based on the values of p_{u,i_r}. The Top-N recommendation is an ordered set of N items, $TopN_u$, that may be of interest to the target user u. The complete algorithm of TRPR method is described in Figure 3.

Algorithm: ***Tensor-based Recommendation using Probabilistic Ranking (TRPR)***

Input: Tag assignment triplets (A) with $|U|$, $|I|$, and $|T|$ as the number of users, items and tags; block-strip row size (q), $|T|\ div\ q = N$ and $|T|\ mod\ q = 0$; tag preference size (s)
Output: $TopN_u$ list of item recommendation

1. Construct initial tensor $\mathcal{Y} \in \mathbb{R}^{U\times I\times T}$ from A where each element presents the existence of user u who has tagged item i with tag t
2. Apply a decomposition technique to tensor $\mathcal{Y}$ to get:
   a. Left singular factor matrices: M_U, M_I, M_T
   b. Size reduction by choosing: $j \in \{1, |U|\}, k \in \{1, |I|\}, l \in \{1, |T|\}$ The reduced matrices: $R_U \in \mathbb{R}^{U\times j}, R_I \in \mathbb{R}^{I\times k}, R_T \in \mathbb{R}^{T\times l}$
   c. Core tensor: $C \leftarrow \mathcal{Y} \times_1 R_U{}' \times_2 R_I{}' \times_3 R_T{}'$ where $C \in \mathbb{R}^{j\times k\times l}$
3. Reconstruct tensor:
   a. 1-mode (matrix) product: $\widehat{\mathcal{Y}_1} \leftarrow C \times_1 R_U$, $\widehat{\mathcal{Y}_1} \in \mathbb{R}^{U\times k\times l}$; Clear C
   b. 2-mode (matrix) product: $\widehat{\mathcal{Y}_2} \leftarrow \widehat{\mathcal{Y}_1} \times_2 R_I$, $\widehat{\mathcal{Y}_2} \in \mathbb{R}^{U\times I\times l}$; Clear $\widehat{\mathcal{Y}_1}$
   c. **For** $n \leftarrow 1$ to N
      $W_n \leftarrow R_T{}^{((n-1)q+1,l)}$, $W_n \in \mathbb{R}^{q\times l}$
      3-mode block (matrix) product: $\widehat{\mathcal{Y}_{3n}} \leftarrow \widehat{\mathcal{Y}_2} \times_3 W_n$, $\widehat{\mathcal{Y}_{3n}} \in \mathbb{R}^{U\times I\times q}$; $\hat{\mathcal{Y}} \leftarrow \widehat{\mathcal{Y}_{3n}}$; Clear W_n, $\widehat{\mathcal{Y}_{3n}}$
      End for /* $\hat{\mathcal{Y}} \in \mathbb{R}^{U\times I\times T}$ */
4. Get the list of candidate items: $Z_u \leftarrow$ new item in $O_{\hat{A}}$ from $\hat{\mathcal{Y}}$ with $O_{\hat{A}} - O_A$
5. Get the list of tag preferences such that ($|X_u| \leq s$): $X_u \leftarrow \max_{v_{\hat{A}}} \hat{\mathcal{Y}}$ with $O_{\hat{A}} - O_A$
6. Calculate posterior probability of each item in candidate item list: $p_{u,i_r} \leftarrow p(i_r)p(X_u)$
7. Generate Top-N item recommendation using posterior probability value:
   If $p_{u,i_r} \in TopN_u$ **then**
      $TopN_u \leftarrow TopN_u \cup \{i_r\}$
   End if

Figure 3: The Algorithm of TRPR Method

Example: Recommendation Generation. The reconstructed tensor of toy problem as shown in Table 2 generates three new entries that correspond to User 2 and User 3. In order to generate Top-N list of items, we derive the list of candidate items and tag preferences for both users. User 2 (u_2) gets candidate items, $Z_2 = \{i_2, i_3\}$, and tag preferences $X_2 = \{t_2, t_5\}$. User 3 (u_3) receives candidate item, $Z_3 = \{i_2\}$, and tag preference $X_3 = \{t_5\}$. Equation 5 is used to calculate the posterior probability values of

u_2 of item i_2 and item i_3. Since p_{u_2,i_2}: 0.0063 > p_{u_2,i_3}: 0.0031, item i_2 is more likely to interest user u_2 than i_3. As a result, the Top-N list of recommended items to u_2 shall be in the sequence order of i_2, i_3. This result differs from the existing tensor-based recommendation approaches which rank the Top-N list of recommended items to u_2 as a sequence order of i_3, i_2.

4. EXPERIMENTAL RESULT

4.1 Dataset and Evaluation Criteria

We used real-world datasets from Delicious (hhtp://delicious.com/) and LastFM (http://www.last.fm/) websites for the experiments. The datasets are refined by selecting users, items, and tags that have occurred in at least k number of posts from the original set. The Delicious dataset is generated with 32,839 tag assignments and 17,077 posts resulted from 15 posts refinement, and is consisted of 1,609 users, 719 items and 1,761 tags. The LastFM dataset is generated with 99,211 tag assignments and 37,163 posts resulted from 10 posts refinement, and is consisted of 867 users, 1,715 items and 1,423 tags.

To evaluate the quality of recommendation, we implemented 2-fold cross-validation and divided the dataset randomly into a training set D_{train} (80%) and a test set D_{test} (20%) based on the number of posts data. D_{train} and D_{test} do not overlap in posts, i.e., there exist no triplets for a user-item combination (u, i) in the training set if a triplet (u, i, t_*) is present in the test set. The recommendation task is to predict and rank the Top-N items for the users present in D_{test}. The performance is measured using F1-Score. F1-Score is a harmonic mean of overall precision and recall. Precision is the ratio of number of relevant items (all items in the post by the user in D_{test}) in the Top-N list to the total number of Top-N recommended items. Recall is the ratio of the number of relevant items in the Top-N list to the total number of relevant items.

$$Precision(D_{test}, N) = avg_{(u,i)\in D_{test}} \frac{|Test_u \cap TopN_u|}{|TopN_u|} \quad (8)$$

$$Recall(D_{test}, N) = avg_{(u,i)\in D_{test}} \frac{|Test_u \cap TopN_u|}{|Test_u|} \quad (9)$$

$$F1(D_{test}, N) = \frac{2 \cdot Precision(D_{test},N) \cdot Recall(D_{test},N)}{Precision(D_{test},N) + Recall(D_{test},N)} \quad (10)$$

Where $Test_u$ is the set of items tagged by target user in the D_{test} and $TopN_u$ is the Top-N list of items recommended to user from the reconstructed tensor $\mathcal{Y}$ which do not exist in the original tensor $\mathcal{Y}$.

4.2 Benchmarking Methods

Results of the proposed method are benchmarked with the standard tensor-based method (denoted as "Max") [5] and the state-of-the-art matrix-based method (denoted as "CTS") [4]. We also used other tensor-based methods which applied CP and HOOI decomposition techniques (denoted as "CP" and "HOOI") for benchmarking. Decomposition techniques were implemented using Matlab Tensor Toolbox [19].

4.3 Empirical Analysis

Firstly, we evaluate the effectiveness of TRPR method for large size tensor reconstruction by comparing the scalability of TRPR with the Max method [5]. Using the 15, 50 , 80 and 100 post refinements (selecting users, items, and tags that have occurred in at least k number of posts), we built the tensor models of different dimensions for the Delicious dataset. The four tensor models were of $1{,}609 \times 719 \times 1{,}761$; $665 \times 52 \times 422$; $362 \times 13 \times 189$; and $250 \times 7 \times 125$ dimension sizes which give the tensor total dimension sizes of 2,037,249,831; 14,592,760; 889,434; and 218,750, respectively. Accordingly, the bigger the number of k being used, the smaller the tensor total dimension size is achieved.

Figure 4 demonstrates the scalability analysis of TRPR and Max methods on a single processor. Note that for the largest data (when tensor total dimension size is 2,037,249,831 derived with k = 15), only TRPR can run while the Max method failed due to memory overflow. The trends show that TRPR is scalable for large tensor size with a linear time computation to the tensor total dimensionality size. The implementation of the block matrix operation [16] simplifies the conventional 3-mode (matrix) product into a memory efficient loop approach. This result also confirms that it is not feasible to apply Tensor models directly on the large datasets as used in this paper.

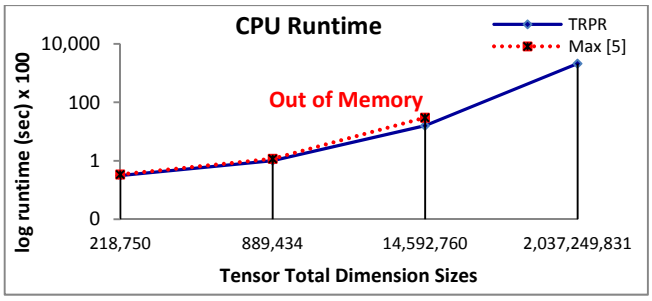

Figure 4: Scalability Comparison by Varying the Tensor Total Dimensionality Sizes of the Delicious Dataset

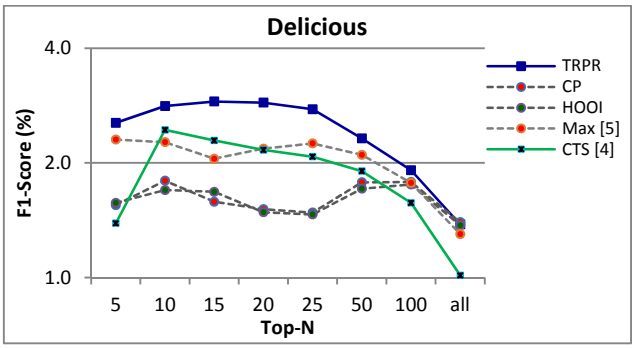

Figure 5: F1-Score Comparison on Top-N list of Recommendations on Delicious Dataset

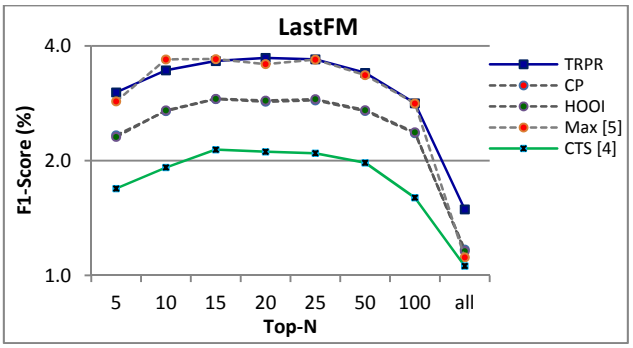

Figure 6: F1-Score Comparison on Top-N list of Recommendation on LastFM Dataset

In order to compare the recommendation quality, we had to implement the block-striped parallel matrix multiplication for scalable tensor reconstruction to the tensor-based benchmarking methods making it applicable for Delicious and LastFM datasets. Using F1-score values, we compare the Top-N lists recommendation quality between TRPR and the benchmarking methods. Figure 5 shows that the proposed method outperforms the benchmarking methods on the Delicious dataset. Likewise, on the LastFM dataset (Figure 6), TRPR is superior compared to other methods, although Max [5] performance is quite comparable with it.

It is to be noted that, in general, F1-Scores achieved for the offline experiments are low. Our experimental setting may be the reason behind this as the dataset has been randomly divided into D_{train} and D_{test} based on the number of posts data. This does not guarantee that for each user in D_{train}, at least one of its post will be selected as D_{test}. Consequently, a target user in D_{test} may not possibly exist in D_{train} (the target user is actually a completely new user).

5. CONCLUSION AND FUTURE WORK

For generating item recommendation in the Social Tagging Systems, we proposed a novel Tensor-based Recommendation using Probabilistic Ranking (TRPR) method by addressing the scalability and accuracy challenges in using tensor models. TRPR developed the simple but effective concept of block-striped parallel matrix multiplication to enable scalable tensor reconstruction as well as it advanced the concept of probabilistic ranking to achieve higher recommendation accuracy. The proposed method is extensively evaluated with the real-world datasets and compared with the state-of-the-art benchmarking methods. Empirical analysis shows that the proposed method is scalable and is able to outperform the benchmarking methods in terms of accuracy. This ascertains that recommendation quality can be improved by using the probabilistic approach after tensor reconstruction. In the future, we are planning to combine the before and after tensor reconstruction approaches by implementing a semantic tag clustering method before we build the tensor model.

6. ACKNOWLEDGEMENTS

This work is supported by the Directorate General of Higher Education (DGHE) Indonesia. Computational resources and services were provided by the HPC and Research Support Group, Queensland University of Technology, Brisbane, Australia.

7. REFERENCE

[1] Mezghani, M., Zayani, C.A., Amous, I., and Gargouri, F. A User Profile Modelling using Social Annotations: A Survey. In *Proceedings of The 21st International Conference Companion on World Wide Web,* pages 969-976, Lyon, France, 2012.

[2] Lü, L., Medo, M., Yeung, C.H., Zhang, Y.-C., Zhang, Z.-K., and Zhou, T., Recommender Systems. *Physics Reports*, 519(1): 1-49, 2012.

[3] Zhang, Z.-K., Zhou, T., and Zhang, Y.-C., Tag-Aware Recommender Systems: A State-of-the-Art Survey. *Journal of Computer Science and Technology*, 26(5): 767-777, 2011.

[4] Kim, H.-N., Ji, A.-T., Ha, I., and Jo, G.-S., Collaborative Filtering based on Collaborative Tagging for Enhancing the Quality of Recommendation. *Electronic Commerce Research and Applications*, 9(1): 73-83, 2010.

[5] Symeonidis, P., Nanopoulos, A., and Manolopoulos, Y., A Unified Framework for Providing Recommendations in Social Tagging Systems Based on Ternary Semantic Analysis. *IEEE Transactions on Knowledge and Data Engineering*, 22(2): 179-192, 2010.

[6] Rafailidis, D. and Daras, P., 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*, 43(3): 673-688, 2013.

[7] Kolda, T. and Bader, B., Tensor Decompositions and Applications. *SIAM Review*, 51(3): 455-500, 2009.

[8] Leginus, M., Dolog, P., and Žemaitis, V., *Improving Tensor Based Recommenders with Clustering*. In *User Modeling, Adaptation, and Personalization*, pages 151-163, Springer Berlin Heidelberg, 2012.

[9] Nanopoulos, A., Item Recommendation in Collaborative Tagging Systems. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 41(4): 760-771, 2011.

[10] Kutty, S., Chen, L., and Nayak, R. A People-to-people Recommendation System using Tensor Space Models. In *Proceedings of The 27th Annual ACM Symposium on Applied Computing,* pages 187-192, Trento, Italy, 2012.

[11] Kolda, T.G. and Sun, J. Scalable Tensor Decompositions for Multi-aspect Data Mining. In *Proceedings of The 8th IEEE International Conference on Data Mining,* pages 363-372, Pisa, Italy, 2008.

[12] Symeonidis, P., Nanopoulos, A., and Manolopoulos, Y. Tag Recommendations based on Tensor Dimensionality Reduction. In *Proceedings of The 2008 ACM Conference on Recommender Systems,* pages 43-50, Lausanne, Switzerland, 2008.

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

[14] Peng, J., Zeng, D.D., Zhao, H., and Wang, F.-y. Collaborative Filtering in Social Tagging Systems based on Joint Item-Tag Recommendations. In *Proceedings of The 19th ACM International Conference on Information and Knowledge Management,* pages 809-818, Toronto, Canada, 2010.

[15] Jain, V. and Varma, M. Learning to Re-rank: Query-dependent Image Re-ranking using Click Data. In *Proceedings of The 20th International Conference on World Wide Web,* pages 277-286, Hyderabad, India, 2011.

[16] Golub, G.H. and Loan., C.F.V., *Matrix Computations*, 4 ed. Baltimore, Maryland, The John Hopkins University Press, 2013.

[17] Baker, L.D. and McCallum, A.K. Distributional Clustering of Words for Text Classification. In *Proceedings of The 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,* pages 96-103, Melbourne, Australia, 1998.

[18] Lops, P., Gemmis, M., and Semeraro, G., *Content-based Recommender Systems: State of the Art and Trends*. In *Recommender Systems Handbook*, pages 73-105, Springer US, 2011.

[19] Bader, B.W., Kolda, T.G., and others. *MATLAB Tensor Toolbox Version 2.5*. [Available online, January 2012], URL: http://www.sandia.gov/~tgkolda/TensorToolbox/.