Exploiting Relational Tag Expansion for Dynamic User Profile in a Tag-aware Ranking Recommender System

Yinghui Pan[a], Yongfeng Huo[b], Jing Tang[c], Yifeng Zeng[d], Bilian Chen[b]

[a]*Department of Information Management, Jiangxi University of Finance and Economics, Nanchang, China*
[b]*Department of Automation, Xiamen Key Laboratory of Big Data Intelligent Analysis and Decision-making, Xiamen University, Xiamen, China*
[c]*Newcastle Business School, Northumbria University, UK*
[d]*Department of Computer and Information Sciences, Northumbria University, UK*

Abstract

A tag-aware recommender system (TRS) presents the challenge of tag sparsity in a user profile. Previous work focuses on expanding similar tags and does not link the tags with corresponding resources, therefore leading to a *static* user profile in the recommendation. In this article, we have proposed a new social tag expansion model (STEM) to generate a *dynamic* user profile to improve the recommendation performance. Instead of simply including most relevant tags, the new model focuses on the *completeness* of a user profile through expanding tags by exploiting their relations and includes a sufficient set of tags to alleviate the tag sparsity problem. The novel STEM-based TRS contains three operations: 1) *Tag cloud generation* discovers potentially relevant tags in an application domain; 2) *Tag expansion* finds a sufficient set of tags upon original tags; and 3) *User profile refactoring* builds a dynamic user profile and determines the weights of the extended tags in the profile. We analysed the STEM property in terms of recommendation accuracy and demonstrated its performance through extensive experiments over multiple datasets. The analysis and experimental results showed that the new STEM technique was able to correctly find a sufficient set of tags and to improve the recommendation accuracy by solving the tag sparsity problem. At this point, this technique has consistently outperformed state-of-art tag-aware recommendation methods in these extensive experiments.

Keywords:
Tag Expansion, Dynamic User Profile, Bayesian Networks, Recommender System

1. Introduction

In recent years, online commerce has outpaced the growth of traditional business, resulting from a rapid expansion of Internet and Web 4.0 applications. Recommender systems have been widely investigated to address information overload problems as the amount of

Email addresses: panyinghui@jxufe.edu.cn (Yinghui Pan), yfhuo@foxmail.com (Yongfeng Huo), jing.tang@northumbria.ac.uk (Jing Tang), yifeng.zeng@northumbria.ac.uk (Yifeng Zeng), blchen@xmu.edu.cn (Bilian Chen)

available information increases exponentially during the same time frame [21]. A collaborative filtering approach, which recommends users on the basis of opinions or actions of other like-minded users, is one of the most promising and popular techniques in the area of recommender systems [24]. This approach considers only whether or not (or the degree to which) a user likes a specific resource and does not take into account the reason why she/he likes it. This could be a major limitation of collaborative filtering based recommendation systems, since two users may like an identical resource for completely different reasons. For example, *Anna* may like the design of a particular mobile, whereas *Bron* may buy this mobile due to its low price. Thus, it is incorrect to conclude that they share an identical or similar preferences. Hence, in order to make a good recommendation, we need to take into account relevant reasons that motivate them to act in either similar or different ways.

On the flip side, diverse types of social media websites have been established, e.g., Del.icio.us, MovieLens, BibSonomy and Last.fm, along with the success of Web 4.0. They generally include a social tagging system that allows users to annotate resources through self-defined tags. In response, a tag-aware recommender system (TRS) has received increasing attention, since it incorporates a social tagging system into a traditional recommendation approach to enhance its performance. Most of the existing TRS work has been limited to tag recommendation while using tags as supplements for resource recommendation has seldom been studied [2].

Learning tagging records of suggesting resource recommendations is a salient part of a social tagging recommender system where applications are user-centred. The connections among users, resources and tags, usually called a folksonomy, enable the users to exploit personal profiles for improving resource recommendations. More importantly, the tagging system makes it easy for people to add meta-data to various resources. Subsequently, the additional meta-data can be used to personalize resource recommendations. For example, Cantador *et al.* [4] considered tags to be features of resources and thus performed a content-filtering recommendation approach. Shepitsen *et al.* [32] employed a set of tag clusters to generate personalized recommendations in folksonomies.

These extensions were promising and depended on sufficient tags that drove the success of tag-aware recommendation systems. However, the unwillingness of users to share tags leads to the tag sparsity problem, which is the challenge we faced in the TRS research. The recommendation accuracy is significantly compromised when few tags are attached to the users or resources. A natural idea is to expand the tags in TRS; however, this expansion is not easy because it requires *completeness* of the tags to describe user profiles, although there is no standard for judging completeness. This completeness greatly affects the recommendation effectiveness. Most of the existing TRS approaches tend to expand similar tags, which makes users reluctant to select the synonym tags [24]. Hence, they do not really address the tag sparsity problem. Yang and Chen [45] adopted Rocchio's algorithm to expand similar tags in personal profiles. They concentrated on investigating the completeness issue that is highly related to the study of query processing or query expansion in information retrieval. Their approach required explicit feedbacks and relied heavily on data characteristics. In this article, we considered completeness of a user profile as follows: A user's tags on a target resource are not sufficient to describe all aspects, but all the users' tags on the resources can completely represent all the aspects. The tags attached to the corresponding resources have a high probability of being preferred by the user. Thus, in this article, we aimed to exploit the relations among these tags so as to discover a complete set of tags for extending a user profile in TRS.

Instead of simply using similar tags, we exploited tags' relations to suggest additional tags in the tag expansion. The expansion contained a new set of tags that had probabilistic relations to the original tags in TRS. To discover relational tags, we resorted to learning Bayesian networks [30] that could structure relations among the tags. The identified tags in a Bayesian network compose a complete set of additional tags; however, this would have been more than what was needed for the recommendation purpose. We took a step further to identify a local set of personal tags from the learned networks that represented the relations of all potential tags in a tagging system. The identified tag composed a *sufficient* set of tags that we subsequently weighted to complete the user profile. Next, the weights assigned to the extended tags were adjusted in proportion to their preferences to corresponding resources. The tag expansion with the weighting mechanism was used to complete the user profiles for the purpose of resource recommendation. Hence, the recommendation could also be explainable, and the user could understand why she/he received the suggested resources. In this setting, we proved that the relational tag expansion with the adjusted weights could contribute to improving recommendation performance.

The remainder of this paper is organized as follows: In Section 2, we review existing recommendation techniques and elaborate both the tag sparsity problem and our intuition on solving it. In Section 3, we give preliminaries on tag-based recommender systems and Bayesian network learning techniques. Section 4 presents our approaches for finding the expanded tags and deciding their weights in TRS. In the last sections, we show our experimental results in Section 5 and conclude the article in Section 6.

2. Related works

In this section, we first review traditional recommender systems. We then discuss related works on recommendations that consider either tag sources or profile expansions. Finally, we present our intuition of how to expand user profiles, i.e., the tag completeness.

2.1. General Recommendation Techniques

Recommender systems generally suggest resources or items (books, news, movies, restaurants, webpages, etc.) that are likely to be of interest to users according to historical records of users, features of resources, and/or user preference feedback. Various types of recommender systems have been proposed and can be broadly classified into either content filtering, collaborative filtering, social network-based, group-based techniques or their hybrids. Content filtering and collaborative filtering are the most popular recommendation approaches [24].

Collaborative filtering (CF) is a widely used technique and has been applied to resource recommendations [28, 41, 47]. CF identifies users whose tastes are similar to those of a target active user and recommends resources that others in the group prefer. The neighbourhood-based technique is most prevalent among different CF approaches, and a key point is to discover a suitable neighbourhood size. The content filtering recommendation approach [26, 33, 34] is based on the idea that the features of resources can be useful in generating interesting recommendations for users [24]. This approach intends to recommend resources similar to those a target user has liked in the past. Thus, the features of resources and a user profile are the only factors influencing recommendation decisions for the user. Association rules are also applied to recommender systems to solve the public cold start problem [27]. For example, Timur *et al.* [29] proposed to extract user preferences based on pairwise association rules from a transaction among the local population, which required a simple system of ratings and solved the cold start problem. They used the association rules to represent item-to-group relationships and inferred the relevance of generated recommendations from the likelihood of their appearing with the observed items, which was done for the final recommender results, whereas we focused on the completeness of user profiles before making recommendations.

Furthermore, a hybrid recommendation combines CF and content filtering methods to suggest innovative recommendations [44]. For example, Ding et al. [10] proposed a method to recommend the right learning resources for users with different learning needs through a hybrid filtering method that learns topics in text to represent resources. Li *et al.* [20] proposed a novel hybrid system to recommend question-and-answer documents in order to alleviate information overload problems. To better extract user features and resources and to improve recommendation quality, deep learning-based recommendation has gained much popularity. Wang *et al.* [38] proposed a collaborative deep learning model by using a bag of words to express text information and learning latent features for resources via Bayesian stacked denoising auto-encoders. Covington *et al.* [8] presented a method for YouTube video recommendations, which turns a recommendation problem into a multi-classification problem based on multilayer perceptron. Wang *et al.* [40] used convolutional neural networks to map images in latent feature space for improving the accuracy of point-of-interest recommendation. Chen *et al.* [6] presented a novel, effective Top-K recommender system for YouTube, which adopts an algorithm based on a strategic gradient called REINFORCE. These neural network models are very good at extracting high-dimensional nonlinear features of users and resources, which improves recommendation accuracy. The biggest difference between these methods and our model was that we have improved the recommendation accuracy through the expansion of user profiles, which is more effective in dealing with the cold start problem.

Moreover, many works have been proposed to eschew dishonest parties in the interest of generating fair recommendations [9, 17]. Hasan *et al.* [17] presented a framework to reserve honest recommendations before applying the suggested trust, based on a dissimilarity function among the suggested trust, which could screen out the dishonest recommendations. Wahab *et al.* [36] presented a two-level dishonesty discouragement mechanism against unfair recommendations on newly deployed cloud services using an endorsement-based trust bootstrapping approach; cloud service is a domain where little evidence about trustworthiness is available.

2.2. Recommendations Fusing with Tags

TRS is a type of content filtering recommender system that has been explored in a considerable amount of literature [2, 11, 43]. It offers users the possibility to annotate resources with personalized tags and to obtain interesting recommendations. A typical tag-aware recommender system [35] for resource recommendation allows tags to be merged into traditional CF algorithms by reducing their ternary relations to three two-dimensional correlations and then fusing user-based and item-based CF algorithms together.

Marek Lipczak *et al.* [23] discussed a potential role of the three tag sources, i.e., resource content, as well as resource profiles and user profiles. They considered a resource title as a starting point to the recommendation process and included both the tags related to the title and the tags present in the profiles of the resource and user. Wang *et al.* [37] proposed a novel topic regression model with social regularization, which could seamlessly integrate an item-tag matrix, item content information and social links between items into a hierarchical Bayesian model. In another study, Yu *et al.* [46] presented a tag recommendation method in folksonomy based on user tagging status, in which the concept of user tagging status is divided into the growing status, the mature status and the dormant status. Subsequently, three corresponding designs are provided to calculate the tag probability distribution based on a statistical language model to suggest tags tagged by the user with a high probability. Li *et al.* [16] designed a tag-based neural attention network by extracting potential tag information to solve the problem of assigning personalized weights for users such that their model could reveal more interrelations between the users and items to make better recommendations. To alleviate the data sparsity problem in TRS, Wang *et al.* [39] proposed a tag-informed cross domain recommendation model to jointly learn resource latent factors in multiple domains, which can represent resource more comprehensively. In addition, Li *et al.* [14] proposed a novel distributional embedding model for capturing useful relationships among tags of users and resources in a CF recommender system.

Deep learning is also widely used in TRS. Huang *et al.* [15] presented a deep semantic similarity model to map queries and documents into low-dimensional latent factors to suggest resources. Kim *et al.* [19] developed a hybrid framework for both resource and tag recommendations to provide interesting content for its users. Liang *et al.* [22] proposed a content-filtering approach by adopting word embeddings to mine semantic factors of tags in order to construct user and resource profiles based on deep models.

2.3. Recommendations Fusing with Profile Expansions

In TRS, users and resources can be assigned profiles defined in terms of weighted lists of social tags [42]. Cantador *et al.* [4] presented and evaluated various content-based recommendation models that made use of the weighted tag profiles of users and resources. Kim *et al.* [18] proposed a new collaborative approach to user modelling by leveraging user-generated tags as preference indicators. They also enriched the user profiles from the neighbours so that the proposed method could provide proper recommendations even if users rated few items. Yang and Chen [45] presented a user-and-resource profile expansion model through the Rocchio algorithm to personalize recommendation in social tagging systems. Ma *et al.* [25] developed a combination of tags and social networks to enrich users' tag profiles from their friends in order to address the data sparsity and cold start problems. Similarly, Xu *et al.* [42] used a deep learning method to map a tag-based user and resource profile into an abstract deep feature space, intending to maximize the effect of the deep-semantic similarity between users and their preferred content in resource recommendation. Furthermore, Fernández *et al.* [12] proposed a high order profile expansion technique to mix various profile expansion methods to alleviate the cold start problem, which achieved 110% improvements when compared with the algorithm without profile expansions.

In this article, we extended user tagging profiles. Differing from the existing work [18] and [25], our idea was to expand a user tag profile by including the tags annotated in the

Symbols	Definitions and Descriptions
F	a folksonomy
U	a set of users; lowercase indicates a user
R	a set of resources; lowercase indicates a resource
T	a set of tags; lowercase indicates a tag
Y	a set of tuples (user, resource, tag)
P_a	user u_a's profile
P_b	resource r_b's profile
$ExtP_a$	the extended user u_a's profile
$ExtT_a$	the extended user u_a's tags
$ExtW_a$	the extended weight vector, corresponding to $ExtT_a$
W	weight vector of a profile; lowercase indicates a weight
K	the size of a tag cloud
T_k	a tag cloud containing K tags
p	the probability
α	the adjustment parameter of W

Table 1. Notation Summary

resources that the user has tagged. The tags do not belong to the user's nearest neighbours or friends. We proposed a general alternative approach to generating resource recommendations by means of relational tag expansion. We sought to find the user's potentially interested tags to help recommend resources that were within her/his usual taste. For example, even if one user likes a resource, she may not comprehensively annotate it (i.e., multiple users can express all aspects of this resource). She could also not have annotated other resources within her interests. To produce a good recommendation, we needed to extend her original tags according to her specific tastes by mining correlations among the tags attached to the resources. In this setting, we considered the tag expansion as a tag variable selection problem in TRS, and we therefore resorted to learning Bayesian networks for relevant variable selections.

3. Preliminaries

In this section, we first introduce a folksonomy in TRS and subsequently detail how to generate profiles for both users and resources in the folksonomy. Finally, we present a general framework of our recommendation model and elaborate on the motivation for exploiting Bayesian networks for tag expansion in order to enrich user profiles in TRS. The mathematical notations used throughout this article are summarized in Table 1.

3.1. Tag-aware Recommender Systems

In TRS, a folksonomy is a system of classification that allows its users to manage tags in order to annotate and categorize resources. A folksonomy F is defined as a collection of a set of users U, a set of tags T, a set of resources R and a ternary relation among them $Y \subset U \times T \times R$ as $F = (U, T, R, Y)$, in which Y is a set of tuple (u, r, t): the user u annotates the tag t to the resource r. A user can annotate a resource with one or more distinctive tags from T. In addition, for a particular user $u_a \in U$ and a resource $r_b \in R$, $T(u_a, r_b)$ includes all the tags annotated by the user on the resource. We assume that $T(u_a, r)$ represents all the tags annotated by the user u_a. The TRS can recommend not only a set of tags T but also a set of interesting resources R to a user. It first ranks a set of resources according to a quality or relevance criteria, and then the top-N resources are selected as the recommendation results.

3.2. User and Resource Profiles

In this paper, we selected a user profile model to suggest recommendations - a strategy commonly used in the existing research [42]. A user u_a's profile P_a was modelled by her/his tagging records, including tag's names $T(u_a) = \{\ldots, t_i, \ldots, t_{|T(u_a)|}\}(\subseteq T \wedge \exists t_i \in TC)$ and weights $W(u_a) = \{\ldots, w_i, \ldots, w_{|T(u_a)|}\}$, where w_i is the frequency of each tag in her tagging history $T(u_a, r)$. Note that we developed a domain *tag cloud* (TC) - a set of related tags and the corresponding weights in the domain of interest, and used it to shrink the overloaded tagging domain.

Two general techniques can be used to generate the tag cloud in an application. One is based on inputs from *experts* and has the advantages of accurate and standard tags; however, it suffers from expensive processes and non-scalability. The other technique is *mined tags*, and is automatic and can thus avoid cold start problems, although it is also noisy and computationally complicated. In our experiments, we adopted the former method to process the *MovieLens* dataset and the latter in other three datasets.

In Eq. 1, the *cosine* similarity is a common way of calculating the correlation between two vectors. We used it to obtain the ranking score between P_a and P_b to generate a ranked recommender list. A resource r_b's profile P_b was defined as that of P_a.

$$cos(\overrightarrow{W(u_a)}, \overrightarrow{W(r_b)}) = \frac{\overrightarrow{W(u_a)} \cdot \overrightarrow{W(r_b)}}{\left|\overrightarrow{W(u_a)}\right| \times \left|\overrightarrow{W(r_b)}\right|} \tag{1}$$

3.3. Bayesian Networks and Markov Blankets

We modelled every user's profile using her/his own tags. In practice, many users, including but not limited to new users, do not have enough tags due to a tag sparsity problem. Therefore, we aimed to expand the tags for refactoring user profiles to improve the recommendation reliability. In this section, we provide an overview of how to handle tag expansions in a formal way.

A Bayesian network (BN) is a commonly used probabilistic graphical model to structure probabilistic relations among variables [30]. A random variable is represented as a node in a directed acyclic graph and an arc in the network models probabilistic relations between variables. The relational strength is encoded in a conditional probabilistic table (CPT) assigned to each node.

We formulated a social tagging expansion model in the form of Bayesian networks, in which each node represents a type of tag in an application domain. The BN provides probabilistic relations among the tags assigned to existing resources over all users in TRS. However, for a specific user, only a local set of tags may be relevant to the user's interests and would then be ascertained from the entire BN model. To find a set of most relevant tags, we resorted to the concept of Markov blanket [30]. A Markov blanket contains only the set of parents, children, and parents of children (i.e., "spouses") of a targeted tag in a BN model. Given the nodes in the Markov blanket, the targeted tag becomes conditionally independent from other tags, which is a useful property of a Markov blanket in the BN model.

In addition, we developed a social tagging expansion model by learning a Markov blanket directly because learning a complete Bayesian network for identifying a local set of tags would have been rather time consuming. We then compared the effectiveness of the two proposed models in the empirical study (Section 5).

3.4. Recommendation Generation

After enriching the user profiles with relational tags, we could then generate a set of personalized recommendations for an active user u_a based on her extended profile $ExtP_a$ and the defined profiles of all the resources described in Section 3.2. Specifically, for each resource r_b for which the active user had not yet expressed a preference, we computed a cosine similarity between its profile, P_b and the user's new profile, $ExtP_a$. The cosine similarity measurement was then adopted and defined in Eq. 1. Finally, top-N resources with top-ranked similarity scores to the active user's profile were selected and recommended to the current user.

4. Social Tagging Expansion Model through Bayesian Networks and Markov Blankets

We developed a social tagging expansion model (STEM) by exploiting relations among tags and assigning proper weights to the expanded tags. By expanding relational tags for a targeted tag in a user profile, we updated the user profile dynamically through assigned weights (see Section 4.2.3) therefore achieving better recommendation accuracy. As the weights were calculated according to the user's profile data that was changing over time, the resulting profile was dynamically towards personal profile.

Fig. 1 shows the framework of the proposed STEM-based TRS in this work. We first decomposed the ternary data tensor (U, R, T) into three two-dimensional correlations which are denoted as (U, R), (U, T) and (R, T) [35], so as to obtain the original user-tag and resource-tag matrices. Thereafter, the proposed TRS contains two main steps: a model construction step and a recommendation step. In the model construction step, we first generated a domain tag cloud to shrink the user-tag matrix for an application purpose and subsequently expanded the latent tags that would be preferred by a target user based on her/his profile, collectively called STEM, using Bayesian networks. In the recommendation step, a user profile refactoring operation was applied to build the target user's new profile according to the expanded tags and corresponding weights returned from STEM. Finally, the TRS suggested recommendations to the target user as described in Section 3.4.

4.1. Construction Strategy of STEM

A problem domain of interest often involves a large number of tags while many of these tags do not the represent preferences of most users. To address this issue, we developed a tag cloud technique to shrink the tagging domain, which subsequently facilitated either a BN or Markov blanket learning process.

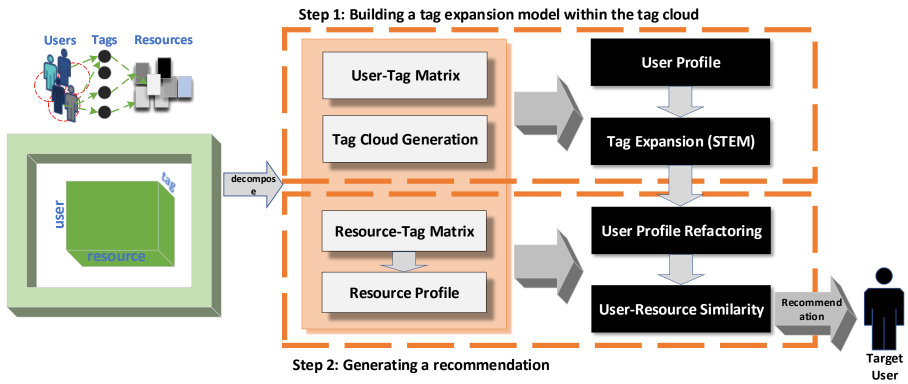

Fig. 1. The framework of the proposed STEM-based TRS

4.1.1. Generation of Tag Cloud

As mentioned in Section 3.2, we resorted to two commonly used approaches to generate a tag cloud. In this work, we used the *expert* approach to deal with the *MovieLens* dataset, since the application domain has already been well studied in the community. There were 18 genre tags annotated with all the movies and we used the 18 tags to constitute the *MovieLens* tag cloud, which was a default setting.

For other datasets, we adopted the *mining* approach to generate their domain tag clouds following the equation below.

$$TagCloud = \{tag|tag\ popularity\ rank \geq K\},$$

where K is adjusted to represent the size of a tag cloud. In Fig. 2, a double logarithmic function shows the distribution of the number of tags in the tag popularity. Notably, the number of tags with a tag popularity of more than 10 is smaller than 1%. Therefore, we were able to select those tags with a popularity of more than 10 to form the domain tag cloud, according to the *Pareto's principle* [31]: That is, in any group of things, the most important one is only a small part of it. We then explored the most suitable size of these domain tag clouds K in Section 5.5 and experimented their impact on recommendation performance.

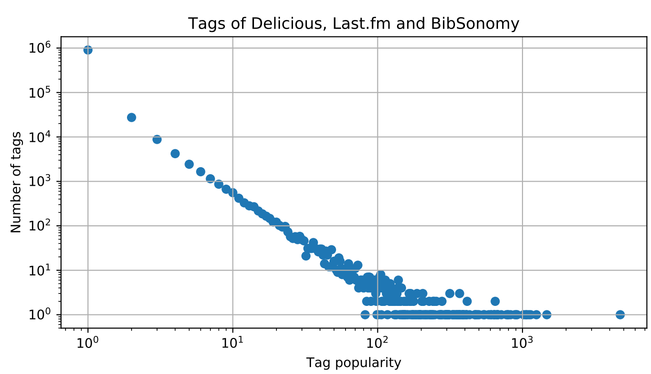

Fig. 2. Distribution of the number of tags on the tag popularity (through a double logarithmic function)

4.1.2. Learning BN for Relational Tags

BN learning contains both structure and parameter learning (i.e., CPT) attached to each node in the network. Parameter learning is based on the known network structure. Hence, we first learned the network structure from the dataset. The methods for learning BN structures upon a complete set of data include the $K2$ algorithm [7], greedy search, hill-climbing, etc. There are also two general approaches to parameter learning in a complete data. One is the maximum likelihood estimation, and the other is Bayesian estimation. Given the reduced number of tags in a tag cloud and a complete dataset, we used the $K2$ algorithm for BN structure learning and the maximum likelihood estimation to calculate the CPTs. In the learned BN, a node represents one tag from the tag cloud, and the node has a binary state - e.g., *yes* or *no* - indicating whether the tag has been assigned to a specific resource or not.

4.1.3. Learning MB for Relational Tags

BN could contain too many tags for a targeted user. Therefore, we proceeded to identify a subset of tags that were sufficient to represent the user's tag profile. We could learn a Markov blanket (MB) directly without spending a lot of time learning a large BN. A tag expansion is actually a variable selection problem for a tag classification. Thus, we employed a well-known MB algorithm - optimal variable selection (HITON) [1] - and learned a Markov blanket for discovering relational tags for the targeted user. Given the generated tag cloud and a tagging dataset, we used the *HITON-PC* method to obtain a specific tag's current parents and children set. Subsequently, we adopted the *HITON-MB* method to get a superset of the tag's Markov blanket (i.e., the parents and children set of the parents and children of the target tag). Finally, we used a cross-validation approach to generate a minimal set, and the minimal set is the Markov blanket for the tag expansion. Given the learned Markov blanket, the target tag is conditionally independent from other tags in the dataset.

4.2. User Profile Expansion with Markov Blanket

For the target user u_a, we aimed to expand her original profile through an identified Markov blanket (MB). We proposed two models to identify the expanded tag set. One is a social tagging expansion model accessed through learning BN (STEM-BN) while the other is social tagging expansion model developed through directly learning MB (STEM-MB). The updated profile $ExtP_a$ is composed of the expanded tags and their corresponding weights, as illustrated below.

$$ExtT_a = T(u_a) \bigcup \left(\bigcup_{t_a \in T(u_a)} MB(t_a) \right)$$

$$ExtW_a = W(u_a) \bigcup \left(\bigcup_{t_a \in T(u_a)} \bigcup_{t \in MB(t_a)} \alpha_i w_{t_a} \cdot w_i \right)$$

where $MB(t_a)$ is the Markov blanket of the tag t_a and w_{t_a} is t_a's original weight in $W(u_a)$, and w_i is $p(t|u_a, t_a)$ in STEM-BN or a constant in STEM-MB, which will be further discussed in the next two subsections. STEM has the nodes in MB as the candidates to be expanded.

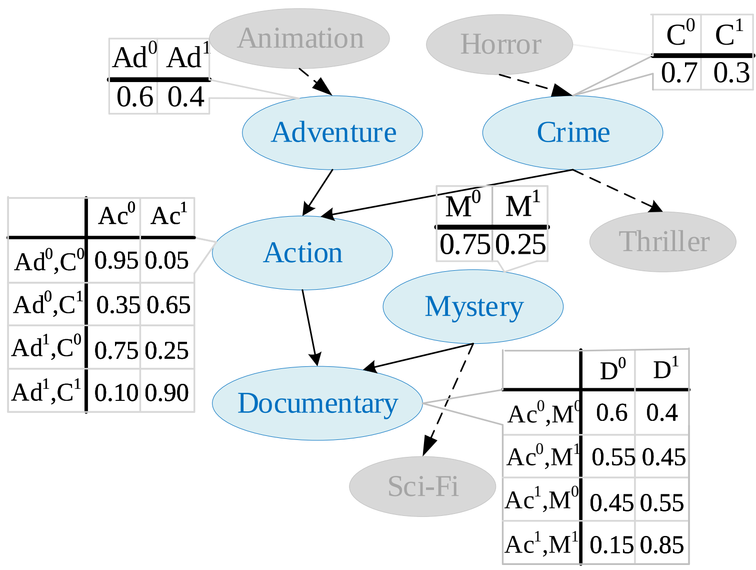

Fig. 3. This shows a BN example for STEM. The superscripts (0 or 1) indicate whether the tag is assigned to a specific resource or not, and the decimals below them represent the corresponding probability of the assignment. The table is the CPT assigned to each node in the BN.

4.2.1. STEM-BN: STEM through Learning BN

As shown in Algorithm 1, we first learned a BN of all the tags in the tag cloud as an input, since the learned BN was shared across all users. Given the tag-based BN example in Fig. 3, we analysed the computation of the posterior probability for the candidate tags $p(MB(t)|U,T)$, given the target user $U = u_a$ and her original tags $T = T(u_a)$. The posterior places a high probability on u_a's potential preferences that best describe her comprehensive interests within a Markov blanket.

In the given BN, the node $Action = Ac$ is the tag assigned by u_a, and its MB is the set of parents ($Adventure = Ad, Crime = C$), child $Documentary = D$ and spouse $Mystery = M$, i.e., $MB(Ac) = \{Ad, C, D, M\}$. We first obtained the posterior probability $p(t|U = u_a, T = Ac)$, where $t \in MB(Ac)$ according to the CPT (lines 6-7). For instance, $p(D|u_a, Ac)$ was calculated: $p(D = yes|U = u_a, Ac = yes, M = no) = p(Ac^1, M^0) = 0.45$. Next, we expanded u_a's profile to $ExtT_a = \{Ac, D, MB(Ac) \setminus D\}$ and $ExtW_a = \{w_{Ac}, 0.45\alpha_1 w_{Ac}, \ldots, \alpha_i w_{Ac} p(t|u_a, Ac)(\forall t \in MB(Ac) \setminus D), \ldots, W_a \setminus w_{Ac}\}$, where α is to be determined as shown in Section 4.2.3 (lines 8-9). We repeated the two steps until all u_a's tags had been expanded (line 3). Note that if a tag had been extended multiple times (i.e., it simultaneously belongs to the Markov blankets of u_a's multiple original tags) we only took its maximum weight in the expansion (line 10).

4.2.2. STEM-MB: STEM through Learning Markov Blanket

STEM-MB is an improved tag expansion model from STEM-BN because it discards the step of learning a huge BN to get the target tag's Markov blanket. Instead, it adopts a novel, sound, sample-efficient and highly-scalable HITON algorithm to select tag variables to form a Markov blanket for the current tag. For a specific tag variable, its Markov blanket is not

Algorithm 1 Social Tagging Expansion through learning BN

Input: $U = u_a$, user profile P_a, the learned BN
Parameter: $T(u_a) = T$, $W(u_a) = W$
Output: u_a's new profile P'_a

Let $i \leftarrow 0$,$N \leftarrow$ length of T.
while $i < N$ **do**
    $t \leftarrow T[i]$;
    $M \leftarrow \{\}, W_t \leftarrow \{\}$;
    Let $M \leftarrow$ the Markov blanket of t through the input BN;
    $W_t \leftarrow$ calculating the posterior $P(t'|u_a, t)$ for each t' in M according to the CPT;
    Extend T with M;
    Extend W with αW_t;
    Remove the duplicate tags in T, leaving only the one with the highest weight;
    $i \leftarrow i + 1$;
end while
return $P'_a \leftarrow (T, W)$.

exactly the same through the two proposed models.

As shown in Algorithm 2, we employed the HITON algorithm in obtaining a target tag's Markov blanket (line 2) and assigned each tag in the MB with the same weight W_t (e.g., the target tag's original weight) in advance (line 3). We extended user u_a's profile with tags in the obtained Markov blanket and assigned the expansion tags with weights αW_t to be determined, the same way as in STEM-BN. In this setting, the weights of these expanded tags would meet the user's hobbies, since we provided a novel dynamic factor, α. These two steps would be repeated until all u_a's tags had been expanded (line 3). Moreover, the main difference between STEM-MB and STEM-BN is that they apply different methods to obtain a Markov blanket of the target tag, which simultaneously brings the difference in determining the weights. The recommendation accuracy of STEM-MB thus increases a little, and STEM-MB is much faster, according to the experimental results in Section 5.4.

Algorithm 2 Social Tagging Expansion through learning MB

Input: $U = u_a$, user profile P_a
Parameter: $T(u_a) = T$, $W(u_a) = W$
Output: u_a's new profile P'_a

1: **for** each tag t in T **do**
2:     $M \leftarrow$ learning Markov blanket for t by the HITON algorithm;
3:     $W_t \leftarrow$ assigning an equal weight (e.g. the target tag's original weight) to each node in M
4:     Extend T with M;
5:     Extend W with αW_t;
6:     Remove the duplicate tags in T, leaving only the one with the highest weight;
7: **end for**
8: **return** $P'_a \leftarrow (T, W)$.

4.2.3. Determining α

We investigated how to decide appropriate weights for the extended tags. Following this, we proposed a dynamic approach to determine α according to the resources to be recommended. To the best of our knowledge, this is the first work that has applied relational tags to build dynamic profiles. Intuitively, the relative size of the weights between the expanded tag and the user's original tag should be kept the same as that of the weights between the corresponding tags of the user's preferred resources. To be precise, given a resource r_b, its profile P_b consists of $T(r_b) = \{\ldots, Ac, Ad, C, D, M\}$ and $W(r_b) = \{\ldots, w'_{Ac}, \ldots, w'_M\}$. We assumed that user u_a was interested in r_b. Note that we suggested recommendations according to the magnitude of cosine similarity between $W(u_a)$ and $W(r_b)$. The user u_a's original weight vector was $W(u_a) = \{\ldots, w_i, 0, 0, \ldots\}$ and her extended weight vector was $ExtW_a = \{\ldots, w_i, \alpha_j w_i w_j, \ldots\}$ as mentioned above. We stated that $\cos(ExtW_a, W(r_b))$ was more than $\cos(W(u_a), W(r_b))$ if we made α satisfy $\alpha_j = w'_k / w'_{Ac} (\forall w'_k \in W(r_b))$, where w'_k corresponded to weight w_k in $W(u_a)$. Hence, the parameter α is dynamic and must be adjusted adaptively for each user. We formulated this statement in Theorem 4.1 and proved it in a multi-tag expansion.

Theorem 4.1. *The dynamic user profile of the proposed STEM leads to a high similarity with a user's interested resources, which is conducive to improved recommendation accuracy in TRS.*

Proof. As a cosine similarity measurement is often used in resource recommendation in TRS, we aimed to prove that the weights assigned to the expanded tags had larger cosine values. Suppose that the user's original weight vector is W_u (Eq. 2), and the resource weight vector is W_r (Eq. 3). There would be an extended weight vector W_e (Eq. 4) if we adopted our STEM, where $p_k \in (0,1] (k = i, \ldots, n+m-2)$ is the extended probability and $W' = w'_i + w'_{n+1}$ (corresponding to w_i, w_{n+1}) in which $w' (\geq 1)$ is from W_r.

$$W_u = \{\ldots, w_i, 0, \ldots, 0, w_{n+1}, 0, \ldots, 0\} \tag{2}$$

$$W_r = \{\ldots, w'_i, \ldots, w'_n, w'_{n+1}, \ldots, w'_{n+m}\}, w' \geq 1 \tag{3}$$

$$\begin{aligned} W_e = &\{\ldots, w_i, \frac{p_i w_i w'_{i+1}}{W'}, \ldots, \frac{p_{n-1} w_i w'_n}{W'}, w_{n+1}, \\ &\frac{p_n w_{n+1} w'_{n+2}}{W'}, \ldots, \frac{p_{n+m-2} w_{n+1} w'_{n+m}}{W'}\} \end{aligned} \tag{4}$$

We let $w = w_i = w_{n+1}$ for simplification. Therefore, according to Eq. 1, there exists

$$\begin{aligned} \frac{\cos(W_e, W_r)}{\cos(W_u, W_r)} &= \frac{\sqrt{2}w}{wW'} \cdot \frac{w\left(w'_i + \sum_{k=i}^{n-1} p_k \frac{w'^2_{k+1}}{W'} + w'_{n+1} + \sum_{k=n}^{n+m-2} p_k \frac{w'^2_{k+2}}{W'}\right)}{w\sqrt{1 + \sum_{k=i}^{n-1} p_k^2 \frac{w'^2_{k+1}}{W'^2} + 1 + \sum_{k=n}^{n+m-2} p_k^2 \frac{w'^2_{k+2}}{W'^2}}} \\ &> \left[\left(2W'^2 + \sum_{k=i}^{n-1} p_k^2 w'^2_{k+1} + \sum_{k=n}^{n+m-2} p_k^2 w'^2_{k+2}\right)^{\frac{1}{2}} - \frac{W'^2}{\sqrt{2W'^2}}\right] \cdot \frac{\sqrt{2}}{W'} \\ &> \left(\sqrt{2}W' - W'/\sqrt{2}\right) \cdot \frac{\sqrt{2}}{W'} \\ &= 1, \end{aligned}$$

where the first inequality holds because of $0 < p \leq 1$ and $W' > 1$. The proof is thus completed. Note that a user may have had more than two tags and we did not show them in W_u because the proof followed the same process. $\square$

4.3. A Toy Example of Generating Recommendations

In the last step, we generated a set of personalized resources for a user u_a based on the STEM-BN example. Given a dataset D and user profile P_a, we first calculated the popularity of each tag and select top-K (K will be determined in Section 5.5) tags T_k. We then filtered records that contained these tags: $t \in T_k$ from D, such that each record was a K-dimensional vector of which each element was a binary value - e.g., 1 or 0 - indicating whether the current tag appeared in this record or not. Next, we input the filtered new data into an easy-to-use BN learning tool (e.g. *Hugin*[1]) to learn a BN with default parameter settings. The learned BN looked like the one in Fig. 3. We subsequently expanded the current P_a to build a new profile $ExtPa$ for u_a, according to the steps in Section 4.2. After obtaining $ExtPa$, we calculated a common cosine similarity between $ExtPa$ and the resources' profiles, and then we sorted them in descending order. Subsequently, the top-N unexposed resources were to be recommended.

5. Experimental Results

We first empirically evaluated STEM against state-of-the-art methods and then studied the impact of different settings of the tag cloud on the STEM performances. We tested STEM versus its competing models to provide a practical STEM-based TRS. The recommender was then implemented in Windows 10 systems, with an Intel(R) Core(TM) i7-6700 @ 3.40GHz CPU and 16GB memory through Matlab programming for learning MB, the *Hugin* tool for learning BN and Python programming for the rest implementation.

5.1. Datasets and Pre-processing

To perform this experiment, we chose four real datasets from these online systems: MovieLens[2], Del.icio.us[3], Last.fm[4] and BibSonomy[5]. The MovieLens dataset contains rating data for multiple movies from multiple users, movie meta-data and user attribute information. We leveraged the movie genre information (18 genres) to form a tag cloud, as described in Section 4.1.1. Next, we obtained the Movielens *ml-20m* dataset [13]. Del.icio.us is a popular social bookmarking system that allows users to not only store and organize their personal bookmarks, but also to annotate and share these URL bookmarks and tag assignments. Besides, Last.fm is an online music system that allows users to tag songs and artists. Both the Delicious and Last.fm datasets can be downloaded from *HetRec 2011* [5]. Finally, BibSonomy is very similar to Del.icio.us and we can obtain the dataset of BibSonomy of July 2012 [3].

[1] https://www.hugin.com
[2] http://www.grouplens.org
[3] http://www.delicious.com
[4] https://www.last.fm/
[5] http://www.bibsonomy.org

Datasets	$\|U\|$	$\|R\|$	$\|T\|$	$\|Y\|$
MovieLens	3,279	18,208	13,222	35,431
Del.icio.us	1,267	33,668	36,686	74,611
Last.fm	892	8,352	10,146	36,378
BibSonomy	2,286	34,146	24,653	84,679

Table 2. Statistic characteristics of the datasets

To reduce the number of calculations required and to adapt to the proposal of a tag cloud, we screened out those users whose tagging records appeared fewer than 1 time in a given tag cloud. Consequently, all of the users were related to the tag cloud in our experiments. Table 2 presents statistic characteristics of these datasets. Note that a user was considered to be interested in the resources that were tagged by that user. The TRS task here was to recommend resources to the users based on their tagging records. For each dataset, 80% of the whole data was randomly selected as the training set, and the remaining 20% of the data constituted the test set. Recommendations were generated based on the known training set, and then the test set was used to evaluate the performance of recommendation algorithms.

5.2. Evaluation Metrics

Users are usually interested in the top-most recommended resources. Hence, we only considered the top-N results in the recommendation list when we applied evaluation metrics. Moreover, we measured both the accuracy of the entire recommendation list and the ranking accuracy. Thus, we used recall@N, precision@N and their harmonic mean, $F1$-measure@N, to evaluate the recommendation accuracy. Recall@N represents the proportion of relevant resources found in the top-N recommendations, while precision@N is the proportion of recommended resources in the top-N set that are relevant. $F1$-measure@N is a harmonic mean of recall@N and precision@N and becomes a comprehensive indicator. Therefore, we have

$$\begin{aligned} precision@N &= \frac{\#\ of\ recommended\ resources\ @N\ that\ are\ relevant}{\#\ of\ recommended\ resources\ @N}, \\ recall@N &= \frac{\#\ of\ recommended\ resources\ @N\ that\ are\ relevant}{total\ \#\ of\ relevant\ resources}\ and \\ F1@N &= \frac{2 \cdot precision@N \cdot recall@N}{precision@N + recall@N}, \end{aligned}$$

where N is a number between 5 and 30 with an interval of 5 in our settings. Note that the three metrics take values from $[0, 1]$, and a large value indicates better quality of recommendations.

Furthermore, because users are generally interested in the top-ranked resources, we used DCG@N to conduct the comparison in terms of the top-N recommendation performance, i.e., the ranking accuracy. For instance, when it comes to precision@10, a relevant resource at position 1 in the recommendation list is considered as useful as another at position 10. On the other hand, DCG@10 penalizes relevant resources appearing close to the tail of a recommended list, and its formula is

$$DCG = \sum_{i=1}^{N} \frac{rel_{u,r}}{\log_2 (i+1)},$$

where N is the length of a recommendation list, and $rel_{u,r}$ is the relevance value of the user u to resource r. If r in the recommendation list satisfies $r \in R(u)$, we set $rel_{u,r} = 1$; otherwise, it is 0.

5.3. Competing Methods

We compared STEM to four other models involving a user/resource profile expansion model in this recommendation: CUM [18], TE-PR [45], CFA [48] and RGD-TR [21]. CUM, which stands for collaborative user modelling, is a collaborative filtering method and enriches an individual user model with collaboration from other similar users. CFA stands for collaborative filtering auto-encoder; it is also a collaborative filtering method in which users' abstract features are extracted from tag-based user profiles by a sparse auto-encoder. In addition, TE-PR, which stands for tag-expansion-based personalized recommendation, is a content-based TRS that adopts a relevance feedback approach, namely the Rocchio algorithm, for both user and resource profile expansions. RGD-TR is also a content-based TRS, although it extracts abstract features for users and resources by a disentangling network and subsequently reconstructs new user and resource profiles under an adversarial protocol. Moreover, we compared our methods to two baselines: ordering resources randomly and ordering resources according to their universal popularity. For each of the competing methods, we used the parameter settings that achieved the best performance according to the following settings. For CUM and TE-PR, there was only one parameter (neighbourhood size S) that is adjusted in their experiments. For CUM, we used the best $S = 10$ because the authors concluded that the neighbourhood with a small size was sufficiently equipped to enrich topics for each user. For TE-PR, we used the best $S_u = 0$ for user profile expansion because the authors concluded that profile expansion did not have a positive contribution for users and the best $S_r = 6$ for resource profile expansion because we could not get a bell curve like the original article. Instead, we used their best parameters. For CFA and RGD-TR, we used the same datasets as their authors; therefore, we needed to use only the original parameter settings. In addition to our STEM, we implemented BN and MB learning, based on the most proper parameter settings, as suggested by the original paper [1, 7].

5.4. Experimental Results

We present in this part the experimental results in terms of the recommendation accuracy and the algorithm runtime on each dataset. We also present the significance test on the recommendation accuracy along with the number of the extended tags. Each result was obtained by computing the average performance of the test set. For the purpose of the comparison, we recorded the best performance of each method for the same training set and test set.

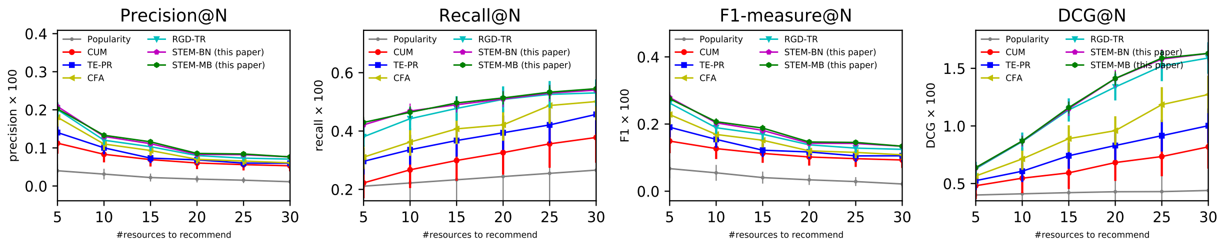

Fig. 4. Performance of various methods on MovieLens

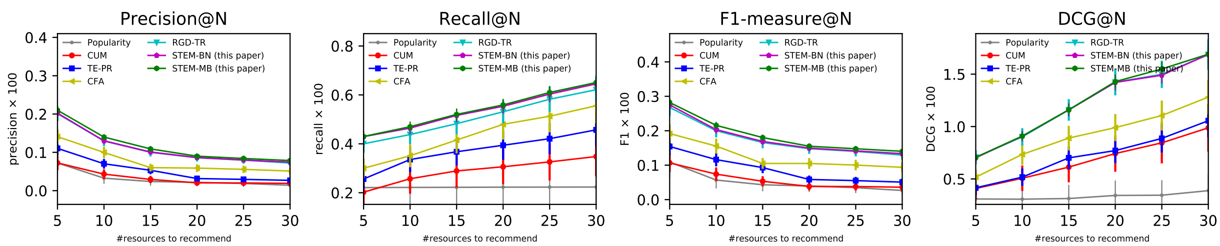

Fig. 5. Performance of various methods on Del.icio.us

5.4.1. *Accuracy*

Figs. 4 - 7 show the precision, recall, and $F1$ and DCG measurements of our methods and their competing methods on four real-world datasets, respectively. The x-axis of each of these figures gives the length of the recommendation list, whereas the y-axis represents these various evaluation indexes we used. Moreover, since it is not clear enough to see the comparative results of the STEM-BN versus STEM-MB from the above figures, Fig. 8 illustrates the comparison between STEM-BN and STEM-MB in terms of $F1$@10 on various datasets when $N = 10$. (We have not illustrated the random baseline because it performed much worse than all other methods.) We concluded the following from our observations:

- As recommendation list size increases, precision and F1-measure tend to decrease, while recall and DCG tend to increase, which was in line with our expectations.
- Overall on the four datasets, the proposed STEM-MB performed a little better than STEM-BN. For example, from Fig. 8, in terms of $F1$-score@10 on Last.fm dataset,

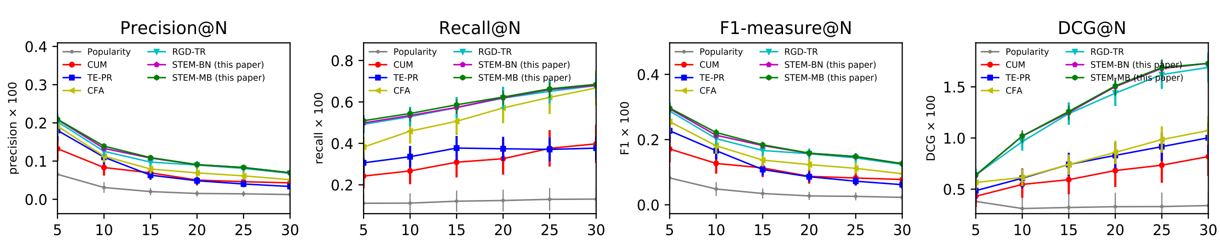

Fig. 6. Performance of various methods on Last.fm

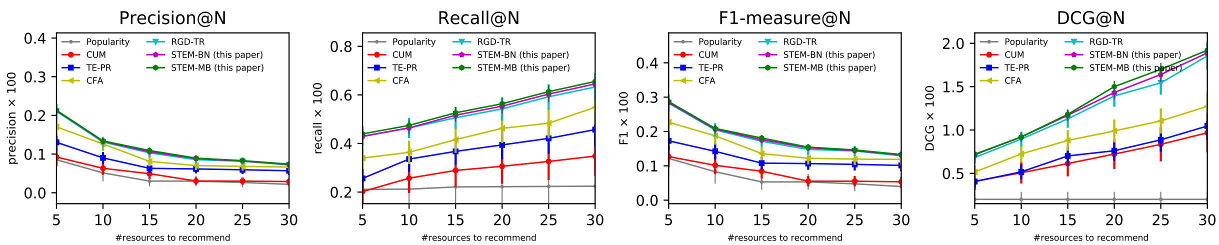

Fig. 7. Performance of various methods on Bibsonomy

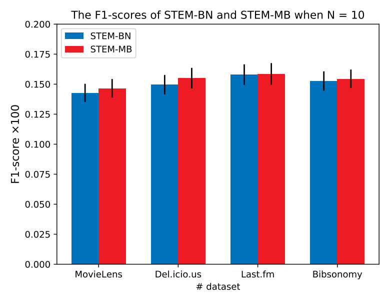

Fig. 8. STEM-BN vs. STEM-MB in terms of F1-score on four dataset when $N = 10$

STEM-MB outperformed STEM-BN by 0.6%, which was partially due to the fact that applying HITON to directly discovering a tag's MB is sounder and more effective than learning BN first and subsequently finding MB for the target tag. Learning a large BN tends to be difficult, therefore leading to an approximate Markov blanket.

- On all four datasets, the proposed STEM-BN significantly outperformed both of the two baseline algorithms, as measured by the four metrics. For example, the improvement of precision@5 over the popularity baseline was 406%, on MovieLens dataset. This was mainly due to the fact that STEM can expand personal tags by relational tags, and STEM-based TRS is more personalized than the popularity method in the recommendation process.

- The proposed STEM-BN attained dramatic improvement over the other two competing approaches without a tag expansion - i.e., CUM and CFA. For instance, in terms of recall@20, STEM-BN outperformed CUM and CFA by 40.4% and 19.7%, respectively, on the BibSonomy dataset. This was mainly because CFA adopts an auto-encoder neural network to extract abstract features for user/resource profiles, in which the auto-encoder is generally used to handle the reduction of data dimensions, whereas our STEM benefited from exploiting relational tags to enrich individual user profiles. This was a completeness of the tags for comprehensively describing user profiles. Thus, the accuracy of resource recommendation could be largely improved by adopting a tag expansion.

- Compared to TE-PR, STEM-BN achieved much better results on all datasets for all the metrics. For example, in terms of $F1$-measure@10, STEM-BN outperformed TE-

PR by 26.6%, 23.4%, 27.5% and 24.8%, respectively, on the four datasets. The results clearly demonstrated the effectiveness of the proposed model. The main reason was that the proposed STEM aimed to expand the user's original tags with probabilistic relations that fully captured users' interest, whereas TE-PR only used similar profiles, and in TE-PR the accuracy of the Racchio algorithm itself was not high enough to accomplish this.

- The proposed STEM-BN performed a little better than RGD-TR in terms of four metrics on all datasets. For example, in terms of recall@20 and DCG@10, STEM-BN outperformed RGD-TR by 1.3% and 1.1%, respectively. As we know, RGD-TR can not only output abstract presentations of users and resources but can also generate reconstructive profiles for them. However, it was hard for RGD-TR to deal with those users who had few tags, wherea our STEM could handle this problem, as long as the users had any tag in the domain tag cloud.

- STEM outperformed its competing methods in terms of DCG on all datasets; as we can see, the lines in green were always at the tops of the four figures. The results verified the effectiveness of exploiting relational tags and endorsed the correctness of Theorem 4.1.

5.4.2. Runtime

We evaluated STEM versus TE-PR because they are content-based recommender models that involve two different tag expansion models. CUM, on the other hand, is a collaborative filtering approach, and both CFA and RGD-TR are two neural networks for tag-aware recommendations. Fig. 9 shows the runtime of STEM-BN, STEM-MB and TE-PR on four datasets. Note that we only recorded the time it took for the model to complete a recommendation online. We found that STEM-MB took the least time and that TE-PR took the most time, regardless of the datasets. Furthermore, TE-PR took about seven times as long as STEM-MB, which was partially due to the fact that

- both the tag cloud and the BN/MB learning can be offline, obtained in advance of forming recommendations,

- TE-PR has to spend time reconstructing the user and resource profiles online and

- HITON is sample-efficient and highly-scalable algorithm for tag variable selections for tag expansions.

5.4.3. Significance Test

It was necessary to explore the impact of extending nodes other than the Markov blanket in terms of recommendation accuracy. We illustrated the trend of recommendation accuracy on the length of the extended tag set in Fig. 10. Note that the $F1$-score almost obeyed a Gaussian distribution, and the $F1$-score had a small increment if we expanded more tags. Therefore, we needed to explore whether such an increment in accuracy was significant enough. Then we used one-way analysis of variance (ANOVA1) to evaluate whether the extended tags outside the MB had a significant effect on the observed $F1$. We first assumed

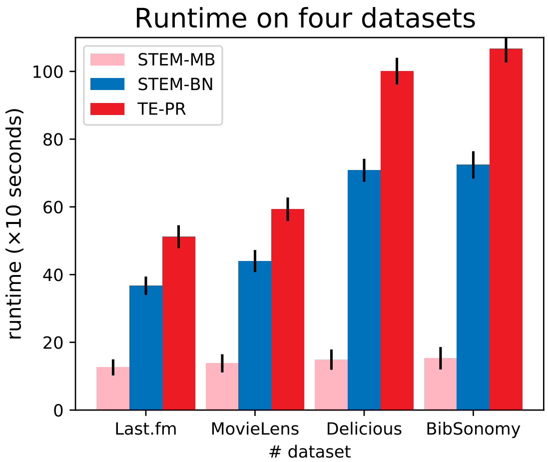

Fig. 9. Runtime in Recommendation

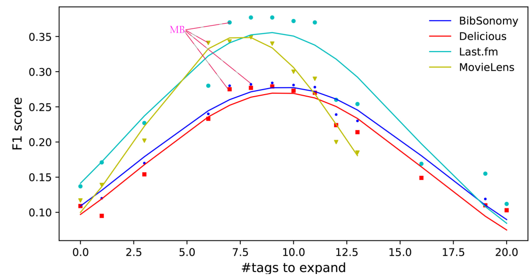

Fig. 10. Recommendation accuracy for the extended tags beyond MB on four datasets

Method	β	p	F_1 [a]	F_2 [b]	Significance?
ANOVA1	0.05	0.54	0.4	4.8	No
		0.53	0.4	4.8	No
		0.78	0.1	4.8	No
		0.59	0.3	5.1	No

[a] F_1 is the F in ANOVA table.
[b] F_2 is obtained from the F distribution table.

Table 3. Significance tests for the $F1$ scores in Fig. 10

that such an increment in accuracy was significant with 95% confidence. The results of this significance test are shown in Table 3, in which β represents the threshold (i.e., the confidence) to accept the current hypothesis, p and F_1 are the return values of the ANOVA1 test, and F_2 is obtained by way of being found on the F-distribution table. We observed that p was always larger than β regardless of datasets, which indicated that we had 95% (i.e., $1-\beta$) certainty that there was no correlation between this increment and the recommendation accuracy. Therefore, this slight accuracy improvement was not significant at all, as it resulted from the statistical tests. This verified the completeness of the Markov blanket for improving the recommendation performance.

5.5. Impact of a Tag Cloud Size K

STEM requires the choice of the size of a domain tag cloud K to learn Bayesian networks of K numbers of nodes. We also needed a tag cloud to discover relational tags in an application domain. Hence, we explored the sensitivity of the choices on the four datasets. Fig. 11 shows that the average $F1$-scores when K varied from 20, 40, 80 and 120. The performance variance did not seem to be high, and the STEM performed the best on every dataset when $K = 40$, which was mainly due to the fact that the BN learned using only the top 20 tags had not completely covered the preferences of all users, which produced an inadequate mining of dependencies between tags, and the BN learned using 80 or more tags may obtain too many

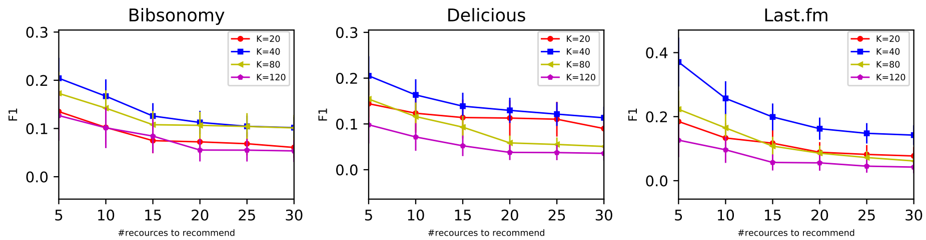

Fig. 11. Performance of generating K-TagCloud, varying K=20,40,80,120

unexpected indirect dependencies. Note that here our STEM currently does not possess the function of selecting whether to expand the current tag based on the size of the expansion probability. Hence, exploring a clever way of selecting K would require a future study on tag clouds.

6. Discussion and Conclusion

In this article, we presented STEM, an MB-based relational tag expansion model that incorporates a user's latent preferences for resources with the latent coexistence of her/his tags. In STEM, a tag cloud technique was proposed to shrink the current domain for an application purpose in TRS. Moreover, a novel idea of dynamic user profile generation was presented to make recommendations more personalized and appropriate. To the best of our knowledge, this has been the first attempt to adopt a dynamic user profile for recommendation in social TRS. We have also conducted a lot of comparative experiments in terms of both recommendation effectiveness and algorithmic runtime. We demonstrated that STEM improved the accuracy of the obtained recommendations via multiple metrics and datasets. Therefore, we conclude that the proposed STEM has the following properties: 1) It discovers the latent coexistence between tags within a Markov blanket, allowing us to analyse the tagging data dynamics; 2) It provides a user with explainable serendipity by adopting the dynamic user profiles; 3) It improves recommendation accuracy in both theoretical and experimental ways. However, STEM is limited in its simplified setting, which suggests a promising direction for future research. We have adopted a relatively simply way to determine the size K of the tag cloud. However, this would be limited in a large dataset. For example, when STEM encounters a completely new domain in which the tags have not been cleaned, relying on tag frequency alone to generate a tag cloud may create too many useless tags. Hence, future research could investigate an effective way to determine the size so that the learned BN could accurately cover all users' main preferred tags.

We noticed that our model does not account for semantic tags as a plurality of keywords, whereas RGD-TR can capture abstract features and map tags on high-order space in such a way that it can generate reconstructive profiles for them. Besides, CFA that adopts a sparse auto-encoder network to represent semantic tags is also encouraging. Hence, future work could also include incorporating a deep neural network for semantic analysis and virtual representations for tags.

The MB-based STEM approach contributes to addressing the tag sparsity problem and provides an interpretable technique in TRS research. However, the MB computation may become rather heavy in a complex domain since a large network needs to be built. We are planning to develop a light STEM approach by simplifying the network and making efficient recommendations.

The STEM-based TRS may face the problem of dishonest recommendations in an ad hoc network, since it cannot recognize incorrect feedback recommended by other nodes, which leads to security concerns. Moreover, directions for future work could also include investigating these malicious behaviours to discover dishonest nodes or groups through trust or reputation computing [9, 17].

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grants No. 61806089, 61772442, 61836005 and 11671335). It is also funded by the Ministry of Justice of the China National Nomocracy and Jurisprudence Grant No. 17SFB3022.

References

[1] Constantin F Aliferis, Ioannis Tsamardinos, and Alexander Statnikov. Hiton: a novel markov blanket algorithm for optimal variable selection. In *AMIA Annual Symposium Proceedings*, volume 2003, page 21. American Medical Informatics Association, 2003.

[2] Valerio Arnaboldi, Mattia Campana, Franca Delmastro, and Elena Pagani. Tag-based recommender system for context-aware content dissemination in opportunistic networks. *University of Pisa Tech Report*, 2015.

[3] Dominik Benz, Andreas Hotho, Robert Jäschke, Beate Krause, Folke Mitzlaff, Christoph Schmitz, and Gerd Stumme. The social bookmark and publication management system bibsonomy. *The International Journal on Very Large Data Bases*, 19(6):849–875, 2010.

[4] Iván Cantador, Alejandro Bellogín, and David Vallet. Content-based recommendation in social tagging systems. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 237–240. ACM, 2010.

[5] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. Second workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *ACM conference on Recommender systems*. ACM, 2011.

[6] Minmin Chen, Alex Beutel, Paul Covington, Sagar Jain, Francois Belletti, and Ed H Chi. Top-k off-policy correction for a reinforce recommender system. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 456–464, 2019.

[7] Gregory F Cooper and Edward Herskovits. A bayesian method for the induction of probabilistic networks from data. *Machine learning*, 9(4):309–347, 2002.

[8] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 191–198, 2016.

[9] Gianni DAngelo, Francesco Palmieri, and Salvatore Rampone. Detecting unfair recommendations in trust-based pervasive environments. *Information Sciences*, 486:31–51, 2019.

[10] Lianhong Ding, Bingwu Liu, and Qi Tao. Hybrid filtering recommendation in e-learning environment. In *2010 Second International Workshop on Education Technology and Computer Science*, volume 3, pages 177–180. IEEE, 2010.

[11] Mohammed El-Absi, Ali Al-Haj Abbas, Ashraf Abuelhaija, Klaus Solbach, and Thomas Kaiser. Distance and tag aware localization in indoor terahertz systems. In *First International Workshop on Mobile Terahertz Systems*, pages 1–5. IEEE, 2018.

[12] Diego Fernández, Vreixo Formoso, Fidel Cacheda, and Victor Carneiro. High order profile expansion to tackle the new user problem on recommender systems. *PloS one*, 14(11), 2019.

[13] F Maxwell Harper and Joseph A Konstan. The movielens datasets: history and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19, 2016.

[14] Ming He, Kaisheng Yao, Peng Yang, and Yuan Yao. Tag2vec: Tag embedding for top-n recommendation. In *The International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, pages 168–175. Springer, 2019.

[15] Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 2333–2338, 2013.

[16] Ruoran Huang, Nian Wang, Chuanqi Han, Fang Yu, and Li Cui. Tnam: A tag-aware neural attention model for top-n recommendation. *Neurocomputing*, 385:1–12, 2020.

[17] M Hasan Islam, Adnan Ahmed Khan, et al. Detection of dishonest trust recommendations in mobile ad hoc networks. In *Fifth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, pages 1–7. IEEE, 2014.

[18] Heung-Nam Kim, Abdulmajeed Alkhaldi, Abdulmotaleb El Saddik, and Geun-Sik Jo. Collaborative user modeling with user-generated tags for social recommender systems. *Expert Systems with Applications*, 38(7):8488–8496, 2011.

[19] Hyunwoo Kim and Hyoung-Joo Kim. A framework for tag-aware recommender systems. *Expert Systems with Applications*, 41(8):4000–4009, 2014.

[20] Ming Li, Ying Li, Wangqin Lou, and Lisheng Chen. A hybrid recommendation system for q&a documents. *Expert Systems with Applications*, 144:113088, 2020.

[21] Zhaoqiang Li, Jiajin Huang, and Ning Zhong. Leveraging reconstructive profiles of users and items for tag-aware recommendation. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1294–1299. IEEE, 2018.

[22] Nan Liang, Hai-Tao Zheng, Jin-Yuan Chen, Arun Kumar Sangaiah, and Cong-Zhi Zhao. Trsdl: Tag-aware recommender system based on deep learning–intelligent computing systems. *Applied Sciences*, 8(5):799, 2018.

[23] Marek Lipczak, Yeming Hu, Yael Kollet, and Evangelos Milios. Tag sources for recommendation in collaborative tagging systems. *ECML PKDD discovery challenge*, 497:157–172, 2009.

[24] Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. In *Recommender systems handbook*, pages 73–105. Springer, 2011.

[25] Tinghuai Ma, Jinjuan Zhou, Meili Tang, Yuan Tian, Abdullah Al-Dhelaan, Mznah Al-Rodhaan, and Sungyoung Lee. Social network and tag sources based augmenting collaborative recommender system. *IEICE Transactions on Information and Systems*, 98(4):902–910, 2015.

[26] Darshita Mittal, Sanyukta Shandilya, Dhruv Khirwar, and Archana Bhise. Smart billing using content-based recommender systems based on fingerprint. In *ICT Analysis and Applications*, pages 85–93. Springer, 2020.

[27] Behzad Soleimani Neysiani, Nasim Soltani, Reza Mofidi, and Mohammad Hossein Nadimi-Shahraki. Improve performance of association rule-based collaborative filtering recommendation systems using genetic algorithm. *Int. J. Inf Technol. Comput. Sci.*, 2:48–55, 2019.

[28] Athanasios N Nikolakopoulos and George Karypis. Boosting item-based collaborative filtering via nearly uncoupled random walks. *arXiv preprint arXiv:1909.03579*, 2019.

[29] Timur Osadchiy, Ivan Poliakov, Patrick Olivier, Maisie Rowland, and Emma Foster. Recommender system based on pairwise association rules. *Expert Systems with Applications*, 115:535–542, 2019.

[30] Judea Pearl. Causality: models, reasoning, and inference. *IIE Transactions*, 34(6):583–589, 2002.

[31] F John Reh. Pareto's principle-the 80-20 rule. *Business Credit-New York Then Columbia MD*, 107(7):76, 2005.

[32] Andriy Shepitsen, Jonathan Gemmell, Bamshad Mobasher, and Robin Burke. Personalized recommendation in social tagging systems using hierarchical clustering. In *ACM conference on Recommender systems*, pages 259–266. ACM, 2008.

[33] Jieun Son and Seoung Bum Kim. Content-based filtering for recommendation systems using multiattribute networks. *Expert Systems with Applications*, 89:404–412, 2017.

[34] Poonam B Thorat, RM Goudar, and Sunita Barve. Survey on collaborative filtering, content-based filtering and hybrid recommendation system. *International Journal of Computer Applications*, 110(4):31–36, 2015.

[35] Karen HL Tso-Sutter, Leandro Balby Marinho, and Lars Schmidt-Thieme. Tag-aware recommender systems by fusion of collaborative filtering algorithms. In *Proceedings of the 2008 ACM symposium on Applied computing*, pages 1995–1999. ACM, 2008.

[36] Omar Abdel Wahab, Robin Cohen, Jamal Bentahar, Hadi Otrok, Azzam Mourad, and Gaith Rjoub. An endorsement-based trust bootstrapping approach for newcomer cloud services. *Information Sciences*, 2020.

[37] Hao Wang, Binyi Chen, and Wu-Jun Li. Collaborative topic regression with social regularization for tag recommendation. In *International Joint Conference on Artificial Intelligence*, pages 2719–2725, 2013.

[38] Hao Wang, Naiyan Wang, and Dit-Yan Yeung. Collaborative deep learning for recommender systems. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1235–1244, 2015.

[39] Jiaqi Wang and Jing Lv. Tag-informed collaborative topic modeling for cross domain recommendations. *Knowledge-Based Systems*, pages 106–119, 2020.

[40] Suhang Wang, Yilin Wang, Jiliang Tang, Kai Shu, Suhas Ranganath, and Huan Liu. What your images reveal: Exploiting visual contents for point-of-interest recommendation. In *Proceedings of the 26th International Conference on World Wide Web*, pages 391–400, 2017.

[41] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, pages 165–174, 2019.

[42] Zhenghua Xu, Thomas Lukasiewicz, Cheng Chen, Yishu Miao, and Xiangwu Meng. Tag-aware personalized recommendation using a hybrid deep model. In *International Joint Conference on Artificial Intelligence*, pages 3196–3202, 2017.

[43] Zhenghua Xu, Oana Tifrea-Marciuska, Thomas Lukasiewicz, Maria Vanina Martinez, Gerardo I Simari, and Cheng Chen. Lightweight tag-aware personalized recommendation on the social web using ontological similarity. *IEEE Access*, 6:35590–35610, 2018.

[44] Chao Yang, Xinghe Chen, Tingting Song, Bin Jiang, and Qin Liu. A hybrid recommendation algorithm based on heuristic similarity and trust measure. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 1413–1418. IEEE, 2018.

[45] Chin-Sheng Yang and Li-Chen Chen. Enhancing personalized recommendation in social tagging systems by tag expansion. In *International Conference on Information Science, Electronics and Electrical Engineering*, volume 3, pages 1695–1699. IEEE, 2014.

[46] Hong Yu, Bing Zhou, Mingyao Deng, and Feng Hu. Tag recommendation method in folksonomy based on user tagging status. *Journal of Intelligent Information Systems*, 50(3):479–500, 2018.

[47] Ervine Zheng, Gustavo Yukio Kondo, Stephen Zilora, and Qi Yu. Tag-aware dynamic music recommendation. *Expert Systems with Applications*, 106:244–251, 2018.

[48] Yi Zuo, Jiulin Zeng, Maoguo Gong, and Licheng Jiao. Tag-aware recommender systems based on deep neural networks. *Neurocomputing*, 204:51–60, 2016.