This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/MIS.2020.3040046, IEEE Intelligent Systems

Neural Graph for Personalized Tag Recommendation

Yonghong Yu, *Member, IEEE,* Xuewen Chen, Li Zhang, *Senior Member, IEEE,* Rong Gao, *Member, IEEE,* Haiyan Gao

***Abstract*—Traditional personalized tag recommendation methods cannot guarantee that the collaborative signal hidden in the interactions among entities is effectively encoded in the process of learning the representations of entities, resulting in insufficient expressive capacity for characterizing the preferences or attributes of entities. In this paper, we firstly propose a graph neural networks boosted personalized tag recommendation model, namely NGTR, which integrates the graph neural networks into the pairwise interaction tensor factorization model. Specifically, we exploit the graph neural networks to capture the collaborative signal, and integrate the collaborative signal into the learning of representations of entities by transmitting and assembling the representations of neighbors along the interaction graphs. In addition, we also propose a light graph neural networks boosted personalized tag recommendation model, namely LNGTR. Different from NGTR, our proposed LNGTR model removes feature transformation and nonlinear activation components as well as adopts the weighted sum of the embeddings learned at all layers as the final embedding. Experimental results on real world datasets show that our proposed personalized tag recommendation models outperform the traditional tag recommendation methods.**

***Index Terms*—Personalized Tag Recommendation Algorithm, Graph Neural Networks, Collaborative Signal**

I. Introduction

As a branch of the recommendation systems [1], tag recommendation systems automatically recommend a list of tags for users to annotate an item. Personalized tag recommendation systems (PTR) [2], [3], [4] provide personalized tag recommendation for each user by taking users' tagging preferences into account, which makes personalized tag recommendation more challenging than non-personalized tag recommendation. Recently, deep learning techniques have shown great potential in various fields, such as natural language processing and computer vision. Among them, the graph neural networks (GNNs) [5] is an effective graph representation learning framework, which learns the representations of nodes or subgraphs that preserve the structures of graphs. In the field of recommendation systems, some researchers incorporate the GNNs into traditional recommendation models to improve the recommendation performance [6], [7], [8]. As shown in the existing studies [6], [7], [8], GNNs could provide great potential to advance the item recommendation models. However, few works have employed the GNNs techniques to boost the PTR. In addition, the collaborative signal is intuitively beneficial to PTR, whose effectiveness is verified in item recommendation [8], [6]. In fact, the collaborative signal can be viewed as the behavior similarity among interacted entities. However, traditional PTR methods obtain the embeddings of entities only based on entities' IDs, and ignore the collaborative signal in the process of embedding. Hence, this scheme limits the expressive capacity of the embeddings.

In this paper, inspired by [8], we firstly propose a GNNs boosted PTR model (NGTR), which integrates the GNNs into the classic pairwise interaction tensor factorization model. Specifically, we consider two bipartite interactions derived from the user-item-tag assignment information, i.e. the user-tag interactions and item-tag interactions. Then, for each type of interactions, we exploit the GNNs to enrich the representations of entities by aggregating the messages of their neighbors, which are propagated over the corresponding interaction graph. In this way, we explicitly inject the collaborative signal into the process of learning representations of entities. In addition, as reported in [9], [10], the GCNs inherit considerable complexity from their deep learning lineage, which can be burdensome and unnecessary for many downstream applications. Moreover, in PTR, each node (i.e. user, item and tag) of the user-tag and item-tag interaction graph only has an ID as input which has no concrete semantics. In this case, performing multiple nonlinear transformation will not be beneficial for GNNs-based PTR model to learn better representations of users, items and tags. Even worse, it may increase the difficulty for training the GNNs-based PTR model and degrade the performance of PTR. Hence, we also propose a light GNNs boosted PTR algorithm, named LNGTR, which removes the feature transformation and nonlinear activation and adopts the weighted sum of the embeddings learned at all layers as the final embedding. Finally, we adopt the BPR criterion [11] to optimize the model parameters of NGTR and LNGTR.

The key contributions of our work are summarized as follows:

- For the task of PTR, we propose to take two types of interactions into account, i.e. the user-tag interactions and the item-tag interactions, and integrate the collaborative signal into the process of embedding by leveraging the embedding propagation layers.
- We propose a GNNs boosted PTR model, namely NGTR, which boosts the classic pairwise interaction tensor fac-

Yonghong Yu, Xuewen Chen and Haiyan Gao are with TongDa College, Nanjing University of Posts and Telecommunications, China. E-mail:yuyh@njupt.edu.cn.

Li Zhang is with Department of Computer and Information Sciences, Northumbria University, UK. E-mail:li.zhang@northumbria.ac.uk

Rong Gao is with School of Computer Science, Hubei University of Technology, China. E-mail:gaorong@hbut.edu.cn

Manuscript received April 19, 2005; revised August 26, 2015.

torization (PITF) model by utilizing the GNNs.

Note that this study is an extension of a previous conference paper in IJCNN2020 [12]. This extension makes the following new contributions:

- Inspired by [9], [10], we propose a light graph neural networks boosted PTR model, namely LNGTR, which removes the feature transformation and nonlinear activation from NGTR.
- We conduct comprehensive experiments to evaluate the effectiveness of LNGTR. The empirical results indicate that the LNGTR is superior to the NGTR in most cases.

II. Related Work

A. Personalized Tag Recommendation Methods

The representatives of PTR methods include HOSVD [13], RTF [14], and PITF [3].

In [13], Symeonidis et al. applied the Higher Order Singular Value Decomposition (HOSVD) technique to reveal the latent semantic associations among entities. By contrast, Rendle et al. [14] proposed the ranking with tensor factorization (RTF), which learns personalized ranking of user preferences for tags. The computation cost of HOSVD and RTF makes them infeasible for large-scale PTR systems. In [3], Rendle et al. proposed the PITF model, which explicitly models the pairwise interactions among entities. Fang et al. [4] proposed a non-linear tensor factorization method, named NLTF, which exploits the Gaussian radial basis function to capture the complex relations among entities. Recently, Yuan et al. [15] proposed a deep-learning-based method, called ABNT, which utilizes the multi-layer perceptron to model the non-linearities of interactions among entities.

B. The GNN-based Item Recommendation Methods

Typical GNN-based item recommendation algorithms include GraphRec [6], SR-GNN [7], and NGCF [8].

Fan et al. [6] presented a GNN framework, namely GraphRec, for social recommendation. The GraphRec coherently models the user-user social graph, the user-item interact graph as well as the heterogeneous strengths. In [7], Wu et al. proposed the SR-GNN model for session recommendation, which utilizes GNN to capture complex item transitions. Wang et al. [8] proposed a recommendation model based on GNNs, which exploits the user-item graph structure by propagating embeddings on it. Wu et al. [16] proposed an influence diffusion neural network based model, namely DiffNet, which leverages the GCN for recursive social diffusion in social networks. Recently, Zheng et al. [17] proposed the price-aware preference-modeling that employs the GCN to learn price-aware and category-dependent user representations. Wang et al. [18] proposed a multi-task multi-view graph representation learning framework for web-scale recommender systems. Hu et al. [19] proposed a graph neural news recommendation model based on unsupervised preference disentanglement.

In comparison with the above methods, the major difference is that our proposed models focus on PTR, while the above existing studies focus on item recommendation. In addition, unlike the GraphRec that only aggregates the representations of one-hop neighbors to learn the representation of the target node, our proposed models take high-hop neighbors into account, which may integrate high-order collaborative signal in the process of embedding.

III. Preliminaries

A. Formalization

PTR systems usually consist of three types of entities: the set of users U, the set of items I and the set of tags T. We use $S \subseteq U \times I \times T$ to denote the interactive behavior records among three entities, i.e. users' historical tagging information. A ternary $(u, i, t) \in S$ indicates that the user u has annotated the item i with the tag t.

The goal of PTR systems is to recommend a ranked list of tags to users for annotating an item. Formally, the ranked list of Top-N tags given the user-item pair (u, i) is defined as,

$$Top(u, i, N) = \underset{t \in T}{\overset{N}{argmax}}\ \widehat{y}_{u,i,t} \tag{1}$$

where N denotes the number of recommended tags. And $\widehat{y}_{u,i,t}$ indicates the probability of the user u annotates the item i with the tag t.

IV. The GNNs Boosted Personalized Tag Recommendation Models

A. The Framework of PTR Method Based on GNNs

Figure 1 presents the architecture of our proposed model, which mainly consists of three layers: the embedding layer, the embedding propagation layer and the prediction layer.

1) Embedding Layer: In the embedding layer, we project users, items and tags into a low-dimensional space according to their IDs. Specifically, we get the embedded representations of the user u, the item i, the positive tag t and the negative tag t' by the lookup operation over the embedding matrices. Formally,

$$\begin{aligned} \mathbf{e}_u &= \mathbf{U}.onehot(u), & \mathbf{e}_i &= \mathbf{I}.onehot(i), \\ \mathbf{e}_t^U &= \mathbf{T}^U.onehot(t), & \mathbf{e}_{t'}^U &= \mathbf{T}^U.onehot(t'), \\ \mathbf{e}_t^I &= \mathbf{T}^I.onehot(t), & \mathbf{e}_{t'}^I &= \mathbf{T}^I.onehot(t'), \end{aligned} \tag{2}$$

where $onehot(.)$ denotes the one-hot encoding operation. $\mathbf{U} \in \mathbb{R}^{|U| \times d}, \mathbf{I} \in \mathbb{R}^{|I| \times d}, \mathbf{T}^U \in \mathbb{R}^{|T| \times d}, \mathbf{T}^I \in \mathbb{R}^{|T| \times d}$ (d is the factorization dimension) are the latent user feature matrix, the latent item feature matrix, the latent user-specific tag feature matrix and the latent item-specific tag feature matrix, respectively.

2) Embedding Propagation Layers: Generally, in PTR systems, there are three types of interactions, i.e. user-tag interactions, item-tag interactions and user-item interactions. Similar to [3], we only consider user-tag interactions and item-tag interactions. For each type of interactions, we employ the message-passing mechanism to capture the collaborative signal along the corresponding bipartite, which is derived from their interaction information. Taking the user-tag interactions as an example, the propagated messages include the information that propagates from tag node to user node as well as information that propagates from user node to tag node.

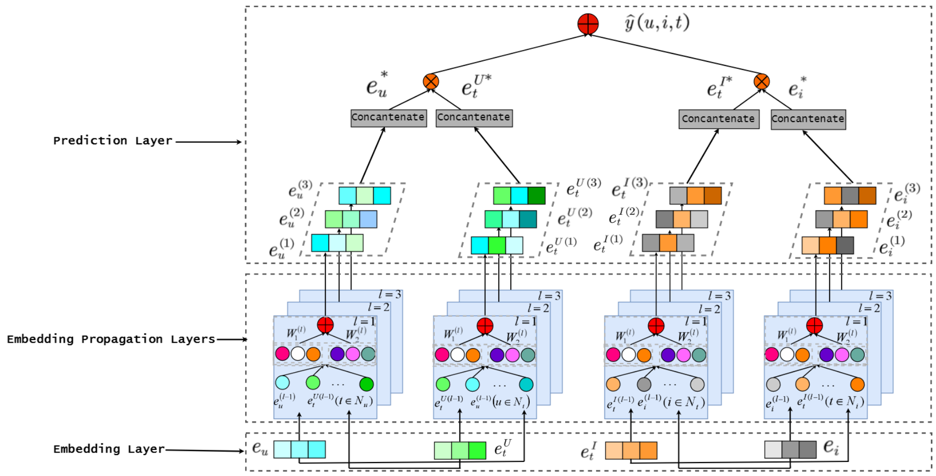

Fig. 1. The framework of our proposed personalized tag recommendation algorithm

Given a user-tag pair (u, t), the propagated messages between the user u and the tag t are defined as follows:

$$
\begin{aligned}
m_{u \leftarrow t} &= p_{ut}\left(\mathbf{W}_1 e_t^U + \mathbf{W}_2\left(e_u \odot e_t^U\right)\right) \\
m_{t \leftarrow u} &= p_{tu}\left(\mathbf{W}_1 e_u + \mathbf{W}_2\left(e_t^U \odot e_u\right)\right)
\end{aligned} \tag{3}
$$

where $m_{u \leftarrow t}$ and $m_{t \leftarrow u}$ indicate the messages that are transmitted from the tag t to the user u and from the user u to the tag t, respectively. And $\odot$ indicates the element-wise product. The p_{ut} and p_{tu} are decay factors that are used to control each message propagation. Formally, p_{ut} and p_{tu} are defined as the Laplacian norm $\frac{1}{\sqrt{|N_u||N_t|}}$, where N_u and N_t represent the first-hop neighbors of the user u and tag t, respectively. The $\mathbf{W}_1, \mathbf{W}_2 \in \mathbb{R}^{d' \times d}$ are training weight matrices, where d' is the transformation size. In Eq. (3), we first follow the principle of classic GNNs and take the messages propagated from neighbors into account. Then, we additionally encode the interaction between target node and its neighbor (e.g. $e_u \odot e_t^U$ and $e_t^U \odot e_u$) into the propagated message. And the interaction reflects the affinity between target node and its neighbor to some extent. Intuitively, neighbors that are more similar to the target node may pass more messages to the target node. Hence, integrating the additional message is able to increase the expressive capacity, resulting in better recommendation performance.

By assembling the messages that are transmitted by the direct neighbors, the assembled representations for the user u and the tag t are represented as follows:

$$
\begin{aligned}
e_u^{(1)} &= \sigma\left(m_{u \leftarrow u} + \sum_{t \in N_u} m_{u \leftarrow t}\right) \\
e_t^{U(1)} &= \sigma\left(m_{t \leftarrow t} + \sum_{u \in N_t} m_{t \leftarrow u}\right)
\end{aligned} \tag{4}
$$

where $\sigma(.)$ is the LeakyReLU activation function [20] and the $m_{u \leftarrow u}$ and $m_{t \leftarrow t}$ consider the self-connections of the user u and the tag t, respectively. Hence, the assembled representations $e_u^{(1)}$ and $e_t^{U(1)}$ explicitly take the first-order connectivity information into account.

In order to further enrich the representations, we inject the high-order connectivity information into the embedded representations of nodes by stacking more embedding propagation layers. Specifically, with l embedding propagation layers, the assembled representations of the user u and the tag t are formulated as:

$$
\begin{aligned}
e_u^{(l)} &= \sigma\left(m_{u \leftarrow u}^{(l)} + \sum_{t \in N_u} m_{u \leftarrow t}^{(l)}\right) \\
e_t^{U(l)} &= \sigma\left(m_{t \leftarrow t}^{(l)} + \sum_{u \in N_t} m_{t \leftarrow u}^{(l)}\right)
\end{aligned} \tag{5}
$$

where $m_{* \leftarrow \star}^{(l)}$ denotes the messages that are propagated from their corresponding l-hop neighbors. Formally,

$$
\begin{aligned}
&\begin{cases}
m_{u \leftarrow t}^{(l)} = p_{ut}\left(\mathbf{W}_1^{(l)} e_t^{U(l-1)} + \mathbf{W}_2^{(l)}\left(e_u^{(l-1)} \odot e_t^{U(l-1)}\right)\right) \\
m_{u \leftarrow u}^{(l)} = \mathbf{W}_1^{(l)} e_u^{(l-1)}
\end{cases} \\
&\begin{cases}
m_{t \leftarrow u}^{(l)} = p_{ut}\left(\mathbf{W}_1^{(l)} e_t^{U(l-1)} + \mathbf{W}_2^{(l)}\left(e_u^{(l-1)} \odot e_t^{U(l-1)}\right)\right) \\
m_{t \leftarrow t}^{(l)} = \mathbf{W}_1^{(l)} e_t^{U(l-1)}
\end{cases}
\end{aligned} \tag{6}
$$

where $W_1^{(l)}, W_2^{(l)} \in R^{d_l \times d_{l-1}}$ are the weight transformation matrices, and the d_l is transformation size. The $e_u^{(l-1)}$ and $e_t^{U(l-1)}$ are the embedded representations that are obtained at the $(l-1)$-th embedding propagation layer.

It is worth noting that we formulate the layer-wise propagation rule defined by Eqs. (5 and 6) as the matrix-form propagation rule to reduce the computation overhead. Formally,

$$
\mathbf{E}_u^{(l)} = \sigma\left((\mathbf{L}_u + \mathbf{I})\mathbf{E}_u^{(l-1)}\mathbf{W}_1^{(l)} + \mathbf{L}_u \mathbf{E}_u^{(l-1)} \odot \mathbf{E}_u^{(l-1)}\mathbf{W}_2^{(l)}\right) \tag{7}
$$

where $\mathbf{E}_u^{(l)}$ is the set of embeddings for users and tags, which is obtained after propagating embeddings with l layers. $\mathbf{I}$ is the identity matrix. And $\mathbf{L}_u$ denotes the laplacian matrix for the user-tag interactions, which is defined as:

$$
\mathbf{L}_u = \mathbf{D}_u^{-\frac{1}{2}} \mathbf{A}_u \mathbf{D}_u^{-\frac{1}{2}} \qquad and \qquad \mathbf{A}_u = \begin{bmatrix} \mathbf{0} & \mathbf{R}_u \\ \mathbf{R}_u^T & 0 \end{bmatrix} \tag{8}
$$

where $\mathbf{R}_u \in \mathbb{R}^{|U|\times|T|}$ is the interaction matrix between users and tags, and $\mathbf{0}$ is all-zero matrix. $\mathbf{A}_u$ is the adjacency matrix and $\mathbf{D}_u$ is the diagonal matrix, where the value of the t-th diagonal element is $|N_t|$. Hence, with the matrix-form propagation rule, we can simultaneously update all user representations and user-specific tag representations.

Similarly, we adopt the similar architecture to deal with the item-tag interactions, and capture the collaborative signal between items and tags.

3) Prediction Layer: By stacking multiple embedding propagation layers, we obtain the set of embedded representations of users, items and tags:

$$\begin{aligned}&\left\{e_u^{(1)}, e_u^{(2)}, \cdots, e_u^{(l)}\right\}\\&\left\{e_i^{(1)}, e_i^{(2)}, \cdots, e_i^{(l)}\right\}\\&\left\{e_t^{U(1)}, e_t^{U(2)}, \cdots, e_t^{U(l)}\right\}\\&\left\{e_t^{I(1)}, e_t^{I(2)}, \cdots, e_t^{I(l)}\right\}\end{aligned} \tag{9}$$

For each entity, the element $e_*^{(l)}$ is the output of embedding propagation layer that assembles messages propagated from the l-hop neighbors. Hence, different element of one set focuses on different order of connectivity information, and characterizes different aspect of users' preferences, items' and tags' characteristics. For each entity, we concatenate all elements to get the final representation for the entity,

$$\begin{aligned}e_u^* &= e_u^{(1)}||e_u^{(2)}||\cdots||e_u^{(l-1)}||e_u^{(l)}\\e_i^* &= e_i^{(1)}||e_i^{(2)}||\cdots||e_i^{(l-1)}||e_i^{(l)}\\e_t^{U*} &= e_t^{U(1)}||e_t^{U(2)}||\cdots||e_t^{U(l-1)}||e_t^{U(l)}\\e_t^{I*} &= e_t^{I(1)}||e_t^{I(2)}||\cdots||e_t^{I(l-1)}||e_t^{I(l)}\end{aligned} \tag{10}$$

where $||$ is the concatenation operation.

Finally, given a triplet (u, i, t), the predicted score $\hat{y}_{u,i,t}$ is computed as:

$$\hat{y}_{u,i,t} = \sum_{f=1}^{K} e_{u,f}^* \cdot {e_{t,f}^U}^* + \sum_{f=1}^{K} e_{i,f}^* \cdot {e_{t,f}^I}^* \tag{11}$$

where K is the dimension of the final representations of entities.

B. A Light GNNs Boosted Personalized Tag Recommendation

In order to further improve NGTR, we also propose a light GNNs boosted PTR algorithm, named LNGTR. The LNGTR also includes three types of layers: the embedding layer, the embedding propagation layers and the prediction layer. Similar to NGTR, we adopt the same scheme to obtain the embedded representations of entities based on their IDs in the embedding layer. We present the details of the embedding propagation layer and prediction layer in the following sections.

1) Simple Embedding Propagation Layer: Unlike the embedding propagation mechanism of NGTR, in order to simplify the tag recommendation model and speed up the training, we discard the feature transformation and nonlinear activation function from NGTR. Taking the user-tag interactions as an example, with l embedding propagation layers, the representations of the user u and the tag t are formulated as:

$$\begin{aligned}e_u^{(l)} &= \sum_{t\in N_u} \frac{1}{\sqrt{|N_u|}\sqrt{|N_t|}} e_t^{U(l-1)}\\e_t^{U(l)} &= \sum_{u\in N_t} \frac{1}{\sqrt{|N_u|}\sqrt{|N_t|}} e_u^{(l-1)}\end{aligned} \tag{12}$$

2) Prediction Layer: With l layers of embedding propagation, we combine the embeddings obtained at each layer to form the final representation for each entity. Formally,

$$\begin{aligned}e_u^* &= \alpha_1 e_u^{(1)} + \alpha_2 e_u^{(2)} + \cdots + \alpha_{l-1} e_u^{(l-1)} + \alpha_l e_u^{(l)}\\e_t^{U*} &= \alpha_1 e_t^{U(1)} + \alpha_2 e_t^{U(2)} + \cdots + \alpha_{l-1} e_t^{U(l-1)} + \alpha_l e_t^{U(l)}\end{aligned} \tag{13}$$

where α_l denotes the weight of embedded representations of entities learned with l embedding propagation layers. In order to simplify the LNGTR as much as possible, we empirically set the weight parameters as $1/(l+1)$, which demonstrates a better tag recommendation performance.

Similarly, we adopt the similar scheme to deal with the item-tag interactions.

C. Model Parameters Learning

We adopt the BPR [11] criterion to learn the model parameters of our proposed tag recommendation models. Both the objective functions of NGTR and LNGTR are defined as follows:

$$\mathcal{L} = \sum_{(u,i,t,t')\in D_S} -\ln\sigma(\widehat{y}_{u,i,t} - \widehat{y}_{u,i,t'}) + \lambda_\Phi||\mathbf{\Phi}||_F^2 \tag{14}$$

where D_S is the training set. For the NGTR, the model parameter is $\mathbf{\Phi} = \{\mathbf{U}, \mathbf{I}, \mathbf{T}^U, \mathbf{T}^I, \mathbf{W}_1^{(i)}, \mathbf{W}_2^{(i)}, i = 1, 2, ..., l\}$. For the LNGTR, the model parameter is $\mathbf{\Phi_L} = \{\mathbf{U}, \mathbf{I}, \mathbf{T}^U, \mathbf{T}^I\}$. λ_Φ denotes regularization coefficient that controls the effect of the regularization terms. In addition, we adopt the mini-batch Adam optimizer to optimize the objective function $\mathcal{L}$.

V. Empirical Analysis

A. Datasets and Evaluation Metrics

We choose two public available datasets, i.e. Last.fm and ML10M [1], to evaluate the performance of our proposed tag recommendation algorithms. Similar to [3], [14], all datasets are 5-core and 10-core in our experiments. The general statistics of datasets are summarized in Table I.

We adopt the common evaluation protocol, which is widely used in [3], [14]. Specifically, for each user, we randomly select one post and remove the triples that are related to the

[1] Two datasets can be found in https://grouplens.org/datasets/hetrec-2011/

TABLE I
STATISTICS OF DATASETS

Dataset	#Users	#Items	#Tags	#Tag Assignments
lastfm	1892	12523	9749	186479
ml-10m	4009	7601	16529	95580

selected post from S to S_{test}. The remaining observed user-item-tag triples are the training set $S_{train} := S \backslash S_{test}$. We employ two widely used ranking metrics to measure the tag recommendation performance of all compared methods, i.e., Precision@N and Recall@N. For both metrics, we set $N = 3, 5, 10$.

B. Experimental Settings

We choose the following traditional tag recommendation algorithms as baselines:

- NGCF: NGCF [8] utilizes the GNNs to boost item recommendation. In our experiments, we use the user-tag interactions as inputs for NGCF.
- PITF: PITF [3] explicitly models the pairwise interaction among entities, and is a strong competitor.
- NLTF: NLTF [4] enhances the PITF by exploiting the Gaussian radial basis function to capture the non-linear interaction relations among entities.
- ABNT: ABNT [15] utilizes the multi-layer perceptron to model the non-linearities of interactions among entities.

For all compared methods, the dimension of latent factor vector d is tuned amongst $\{8, 16, 32, 64, 128, 256, 512, 1024\}$. The mini-batch size is selected from $\{512, 1024, 2048\}$ and the learning rate is tuned amongst $\{0.001, 0.005, 0.01\}$. The regularization coefficient is chosen from $\{0.001, 0.005, 0.01, 0.05\}$. For most datasets and baselines, we empirically set the dimension of latent factor vector $d = 64$, the size of batch = 512, the learning rate = 0.001, the regularization coefficient of latent factor vector = 0.01, the number of negative instances = 1 and the number of iterations = 3000. For the ABNT, the structure of multi-layer perceptron follows the tower structure, i.e. the dimension of hidden layer is half of that of the previous hidden layer. Moreover, for the weighted matrices of multi-layer perceptron, the regularization coefficient = 1. For NGCF, NGTR and LNGTR, we set the number of embedding propagation layers $l = 3$.

C. Performance Comparison

Tables II, III, IV, V present the tag recommendation qualities of all compared methods on four datasets.

From Table II to Table V, we have the following observations: (1) On four datasets, PITF achieves a better performance than those of NTLF and ABNT, which demonstrates the strong competitiveness of PITF model. On the other hand, the observation also indicates that integrating the multi-layer perceptron into PITF framework cannot guarantee improvements of tag recommendation quality, although ABNT is built upon the PITF. One possible reason is that the ABNT involves more trainable parameters, whereas train data available are insufficient for learning its model parameters. Except for ABNT, other methods are superior to NGCF. This is owing to the fact that NGCF only considers the user-tag interactions and ignores the item-tag interactions when making tag recommendations. (2) For each compared method, its recommendation performance is better on the core-10 datasets than that on the corresponding core-5 datasets. This observation indicates that increasing the density of datasets could boost the tag recommendation performance. (3) Our proposed NGTR model consistently outperforms other methods, which demonstrates the effectiveness of NGTR. In terms of precision@3, our proposed NGTR model improves the PITF by 9.3% and 4.1% on last.fm-core5 and ml-10m-core5, respectively. In terms of precision@5, the improvements of NGTR over PITF are 2.7% and 18.6% on last.fm-core10 and ml-10m-core10, respectively. To some extent, the improvements are considerable. Hence, this observation confirms that integrating the collaborative signal into the process of embedding in an explicitly manner is beneficial for the PTR model. (4) In addition, our proposed LNGTR model outperforms the NGTR model in all test cases.

TABLE II
PERFORMANCE COMPARISONS ON LASTFM-CORE5

Method	NGCF	PITF	NLTF	ABNT	**NGTR**	**LNGTR**
Pre@3	0.16790	0.21266	0.19486	0.15628	**0.23244**	**0.28709**
Pre@5	0.13947	0.17893	0.16780	0.13531	**0.19125**	**0.22938**
Pre@10	0.10230	0.12737	0.11907	0.10178	**0.13272**	**0.15393**
Rec@3	0.21913	0.25711	0.22753	0.15691	**0.32444**	**0.38627**
Rec@5	0.29070	0.34786	0.32389	0.21940	**0.41697**	**0.48899**
Rec@10	0.40126	0.48138	0.45230	0.32984	**0.54541**	**0.61167**

TABLE III
PERFORMANCE COMPARISONS ON LASTFM-CORE10

Method	NGCF	PITF	NLTF	ABNT	**NGTR**	**LNGTR**
Pre@3	0.17391	0.25132	0.24431	0.16046	**0.26467**	**0.32298**
Pre@5	0.14679	0.20875	0.20642	0.13665	**0.21429**	**0.25776**
Pre@10	0.11398	0.14577	0.12493	0.09431	**0.14617**	**0.17567**
Rec@3	0.21800	0.32035	0.28488	0.15792	**0.34791**	**0.41993**
Rec@5	0.28778	0.41583	0.40170	0.21895	**0.45288**	**0.53992**
Rec@10	0.42892	0.56539	0.55412	0.30336	**0.58738**	**0.68393**

TABLE IV
PERFORMANCE COMPARISONS ON ML-10M-CORE5

Method	NGCF	PITF	NLTF	ABNT	**NGTR**	**LNGTR**
Pre@3	0.09495	0.13976	0.13232	0.08215	**0.14545**	**0.19226**
Pre@5	0.06828	0.10206	0.09717	0.06283	**0.10545**	**0.13960**
Pre@10	0.04384	0.06414	0.05960	0.04000	**0.06717**	**0.08586**
Rec@3	0.24631	0.32077	0.29738	0.20888	**0.33312**	**0.43114**
Rec@5	0.28195	0.39096	0.35602	0.25378	**0.39653**	**0.50241**
Rec@10	0.34948	0.46230	0.42697	0.30388	**0.48516**	**0.60131**

TABLE V
PERFORMANCE COMPARISONS ON ML-10M-CORE10

Method	NGCF	PITF	NLTF	ABNT	**NGTR**	**LNGTR**
Pre@3	0.12438	0.16986	0.14357	0.08955	**0.19332**	**0.27150**
Pre@5	0.08955	0.11725	0.11429	0.07591	**0.13902**	**0.18891**
Pre@10	0.05714	0.07443	0.07143	0.05011	**0.08422**	**0.11023**
Rec@3	0.31983	0.37704	0.33881	0.22100	**0.46023**	**0.62775**
Rec@5	0.38369	0.45230	0.43344	0.30174	**0.54606**	**0.71876**
Rec@10	0.46876	0.52050	0.53408	0.38579	**0.63980**	**0.80117**

We attribute the improvement of recommendation performance of LNGTR to the simple structure of underlying GNNs, which removes the feature transformation and nonlinear activation from NGTR. The light GNNs make the LNGTR more concise and are appropriate for effectively learning the embedded representations of entities in PTR, which results in better tag recommendation performance.

D. Impact of The Number of Embeddings Propagation Layers

In this section, we conduct a group of experiments to explore the effect of l on tag recommendation performance by varying the value of l from 1 to 4. Other parameters keep the same settings as described in Section V-B. The experimental results in terms of precision@10 on lastfm-core10 and ml-10-core10 are shown in Figs. 2 and 3. Other measure metrics show similar trends.

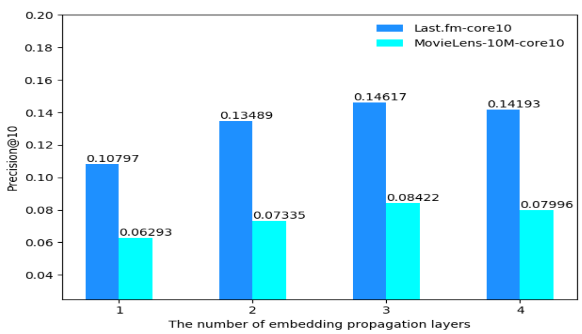

Fig. 2. Impact of l on NGTR

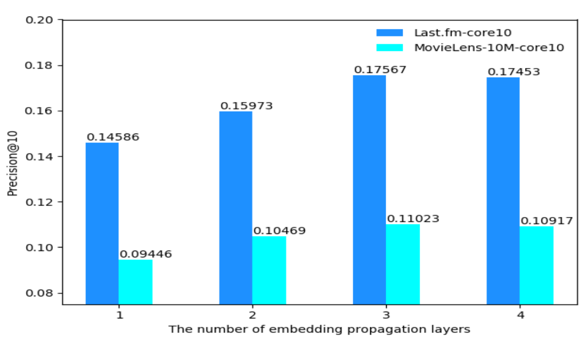

Fig. 3. Impact of l on LNGTR

As shown in Figs. 2-3, our proposed models (NGTR and LNGTR) are sensitive to the value of l. With the number of embedding propagation layers increases, the $Prescision@10$ firstly increases. Then, if the number of embedding propagation layers continues to increase and surpasses a threshold value, the performance of our proposed models begins to degrade. A possible reason is that: a large value of l makes our proposed methods leverage the collaborative signal that is propagated from the relative distant neighbors. Intuitively, the collaborative signal of the distant neighbors may not be helpful for enriching the representation of target entities since the correlations between entity and their distant neighbors are weak. When the number of embedding propagation layers $l = 3$, both NGTR and LNGTR achieve their best performance.

E. Impact of The Dimension of Latent Feature Vectors

In this section, we vary the dimension of the hidden feature vectors d in [16, 32, 64, 128, 256], and investigate the impact of parameter d on tag recommendation quality. Other parameters remain unchanged. We only plot the precision@10 of NGTR and LNGTR on lastfm-core10 and ml-10m-core10 in Figs. 4 and 5. And other measure metrics show similar trends.

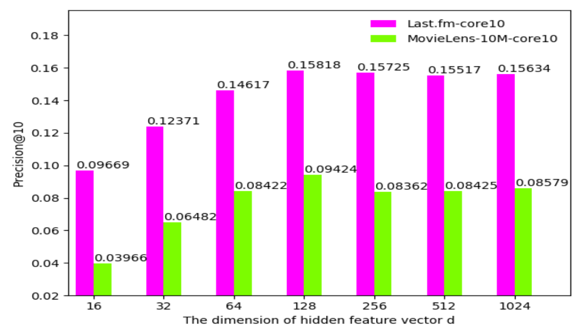

Fig. 4. Impact of parameter d on NGTR.

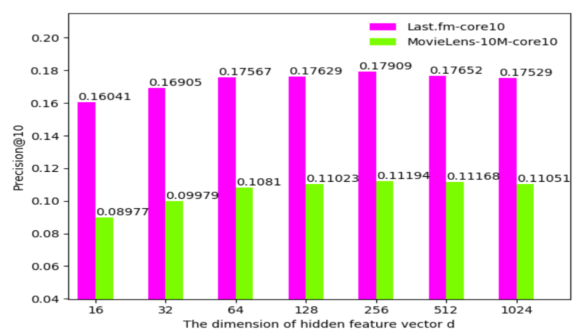

Fig. 5. Impact of parameter d on LNGTR.

As can be seen, the dimension of latent feature vectors d also affects the tag recommendation performance. In the early stage, the recommendation performance of NGTR and LNGTR is constantly improving as the value of d increases. Then, when the value d reaches to 128, the curve of precision@10 remains stable and the tag recommendation performance does not further improve as we further increase the value of d. This is because that if the latent feature vectors can capture the interacted entities' preferences or characteristics effectively, further increasing the value of d could not enhance the representation capacity of our proposed models. On both lastfm-core10 and ml-10m-core10, our proposed NGTR and LNGTR models achieve their best performances when d is around 128 and 256, respectively.

VI. Conclusion

In this research, we firstly propose a GNNs boosted personalized tag recommendation model, which exploits the GNNs to capture the collaborative signal between interacted entities as well as integrate the collaborative signal into the process of embedding by performing messages propagation over the entity interaction graphs. We also propose a light GNNs boosted personalized tag recommendation algorithm, which removes the feature transformation and nonlinear activation from NGTR. Experimental results on real world datasets indicate that both of our proposed models outperform the traditional tag recommendation models.

Recently, the generative adversarial networks [21] have been widely applied in natural language processing (NLP) and computer vision (CV), and show great potential in these respective fields. We plan to explore whether the training method of GAN can be applied to enhance the robustness of our proposed models. In addition, another future research direction is to adaptively learn the weights of representations yielded by each embedding propagation layer since their contributions to the learning of final representations of entities may be different. Finally, we also aim to explore the inclusion of the neural attention model to further improve our proposed approaches.

Acknowledgment

This work is supported in part by the Natural Science Foundation of the Higher Education Institutions of Jiangsu Province (17KJB520028), Tongda College of Nanjing University of Posts and Telecommunications (XK203XZ18002), Qing Lan Project of Jiangsu Province and Doctoral Scientific Research Foundation of Hubei University of Technology (BSQD2019026).

References

[1] G. Adomavicius and A. Tuzhilin, "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions," *TKDE*, no. 6, pp. 734–749, 2005.

[2] R. Jäschke, L. Marinho, A. Hotho, L. Schmidt-Thieme, and G. Stumme, "Tag recommendations in folksonomies," in *PKDD*, 2007, pp. 506–514.

[3] S. Rendle and L. Schmidt-Thieme, "Pairwise interaction tensor factorization for personalized tag recommendation," in *WSDM*, 2010, pp. 81–90.

[4] X. Fang, R. Pan, G. Cao, X. He, and W. Dai, "Personalized tag recommendation through nonlinear tensor factorization using gaussian kernel," in *AAAI*, 2015.

[5] W. L. Hamilton, R. Ying, and J. Leskovec, "Representation learning on graphs: Methods and applications," *arXiv:1709.05584*, 2017.

[6] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," in *WWW*, 2019, pp. 417–426.

[7] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *AAAI*, 2019, pp. 346–353.

[8] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *SIGIR*, 2019.

[9] F. Wu, T. Zhang, A. H. de Souza, C. Fifty, T. Yu, and K. Q. Weinberger, "Simplifying graph convolutional networks," in *ICML*, 2019.

[10] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," *arXiv preprint arXiv:2002.02126*, 2020.

[11] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *UAI*, 2009, pp. 452–461.

[12] F. J. L. Z. R. G. Xuewen Chen, Yonghong Yu and H. Gao, "Graph neural networks boosted personalized tag recommendation algorithm," in *IJCNN*, 2020, pp. 1–8.

[13] P. Symeonidis, A. Nanopoulos, and Y. Manolopoulos, "Tag recommendations based on tensor dimensionality reduction," in *RecSys*, 2008, pp. 43–50.

[14] S. Rendle, L. Balby Marinho, A. Nanopoulos, and L. Schmidt-Thieme, "Learning optimal ranking with tensor factorization for tag recommendation," in *KDD*, 2009, pp. 727–736.

[15] J. Yuan, Y. Jin, W. Liu, and X. Wang, "Attention-based neural tag recommendation," in *DASFAA*, 2019, pp. 350–365.

[16] L. Wu, P. Sun, Y. Fu, R. Hong, X. Wang, and M. Wang, "A neural influence diffusion model for social recommendation," in *SIGIR*, 2019, pp. 235–244.

[17] Y. Zheng, C. Gao, X. He, Y. Li, and D. Jin, "Price-aware recommendation with graph convolutional networks," in *ICDE*, 2020, pp. 133–144.

[18] M. Wang, Y. Lin, G. Lin, K. Yang, and X. Wu, "M2grl: A multi-task multi-view graph representation learning framework for web-scale recommender systems," in *KDD*, 2020, pp. 2349–2358.

[19] L. Hu, S. Xu, C. Li, C. Yang, C. Shi, N. Duan, X. Xie, and M. Zhou, "Graph neural news recommendation with unsupervised preference disentanglement," in *ACL*, 2020, pp. 4255–4264.

[20] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *ICML*, 2013.

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014, pp. 2672–2680.

Yonghong Yu is an Associate Professor at TongDa College, Nanjing University of Posts and Telecommunications. He received a PhD degree from Nanjing University. His main research interests include machine learning, recommender systems.

Xuewen Chen is an Undergraduate at TongDa College, Nanjing University of Posts and Telecommunications. His main research interests include machine learning and recommendation algorithms.

Li Zhang received a PhD degree from the University of Birmingham. She is currently an Associate Professor & Reader in Computer Science in Northumbria University, UK. She holds expertise in evolutionary computation and deep learning.

Rong Gao is a assistant professor at School of Computer Science, Hubei University of Technology. His research interests include data mining, intelligent recommendation.

Haiyan Gao is a Lecturer at TongDa College, Nanjing University of Posts and Telecommunications. Her main research interests include data mining and user behavior analysis.