Partial Relationship Aware Influence Diffusion via a Multi-channel Encoding Scheme for Social Recommendation

Bo Jin
Dalian University of Technology
Dalian, China
jinbo@dlut.edu.cn

Ke Cheng
Dalian University of Technology
Dalian, China
ckpassenger@gmail.com

Liang Zhang*
Dongbei University of Finance and Economics
Dalian, China
liang.zhang@dufe.edu.cn

Yanjie Fu
University of Central Florida
Orange City, FL
yanjie.fu@ucf.edu

Minghao Yin
Northeast Normal University
Changchun, China
ymh@nenu.edu.cn

Lu Jiang
Northeast Normal University
Changchun, China
jiangl761@nenu.edu.cn

ABSTRACT

Social recommendation tasks exploit social connections to enhance recommendation performance. To fully utilize each user's first-order and high-order neighborhood preferences, recent approaches incorporate influence diffusion process for better user preference modeling. Despite the superior performance of these models, they either neglect the latent individual interests hidden in the user-item interactions or rely on computationally expensive graph attention models to uncover the item-induced sub-relations, which essentially determine the influence propagation passages. Considering the sparse substructures are derived from original social network, we name them as partial relationships between users. We argue such relationships can be directly modeled such that both personal interests and shared interests can propagate along a few channels (or dimensions) of latent users' embeddings. To this end, we propose a partial relationship aware influence diffusion structure via a computationally efficient multi-channel encoding scheme. Specifically, the encoding scheme first simplifies graph attention operation based on a channel-wise sparsity assumption, and then adds an InfluenceNorm function to maintain such sparsity. Moreover, ChannelNorm is designed to alleviate the oversmoothing problem in graph neural network models. Extensive experiments on two benchmark datasets show that our method is comparable to state-of-the-art graph attention-based social recommendation models while capturing user interests according to partial relationships more efficiently.

CCS CONCEPTS

• **Human-centered computing** → **Social recommendation**; • **Information systems** → **Recommender systems**.

*Corresponding Author.

CIKM '20, October 19–23, 2020, Virtual Event, Ireland

ACM ISBN 978-1-4503-6859-9/20/10…$15.00
https://doi.org/10.1145/3340531.3412016

KEYWORDS

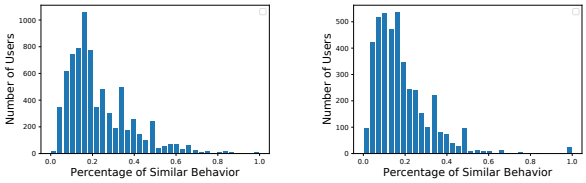
Social Network; Recommendation; Graph Neural Network

ACM Reference Format:
Bo Jin, Ke Cheng, Liang Zhang, Yanjie Fu, Minghao Yin, and Lu Jiang. 2020. Partial Relationship Aware Influence Diffusion via a Multi-channel Encoding Scheme for Social Recommendation . In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management (CIKM '20), October 19–23, 2020, Virtual Event, Ireland*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3340531.3412016

1 INTRODUCTION

Discerning user preference with sparse user behavior data is a key issue in a recommendation task. The social recommendation has emerged as a pioneering direction based on the social influence theory which states connected people would show similar interests pattern [30, 33, 35, 39]. Ideally, a successful social recommendation system should capture temporal or evolutionary individual interests and shared interests. Hence uncovering self interests [2] hidden in user-item interactions while identifying the shared interests according to social influence become the most essential. However, unlike single-label graph learning tasks, a recommendation system based on social network is usually a multi-label graph learning task with extreme label (or user-item interaction) sparsity. In fact, it is the sparse label-induced social relationship substructures [4, 37] that determine the social influence channels. As a result, the traditional assumption that linked users in a social network tend to share the same interests no longer fits for all the users. For instance, Figure 1 illustrates the frequency distribution of similar behaviors between each user and all neighbors according to social network structure of *Yelp* and *Flickr*. The right-skewed distributions demonstrate though users in social network show some similar behaviors with neighborhoods, they largely maintain their own interests. Therefore, it is of utmost importance to capture social influence according to sparse substructures rather than the original social network structures.

Previous studies for social recommendation attempt to model social effects in various ways, such as by trust propagation, regularization loss, matrix factorization, network embedding and deep neural networks [30, 31, 38]. However, these studies have some limitations. First, most works model friends' influence statically or dynamically without differentiating influence importance, such as



(a) *Yelp* (b) *Flickr*

Figure 1: Frequency distributions of similar behaviors between each user and all neighbors according to social network structure on *Yelp* and *Flickr*.

spatial Graph Neural Network (GNN) approaches [11, 30, 36], which apparently ignore the differences of social effects, and will further lead to oversmoothing problems. Second, even though attention-based models [29–31] have been proposed to distinguish social effects, they mainly calculate the node-wise similarities, which are computationally expensive in large-scale social recommendation tasks. In essence, *Diffnet++* [29] and *DANSER* [31] achieve superior performance because they model the label-substructures [4] by incorporating user-item interactions in the information diffusion process of the social network. In our work, we name the label-induced substructures as partial relationships. Moreover, the light-weight convolution [28] inspires us that node-wise computational complexity can be mitigated by splitting the node representation into chunks. Hence, we argue that such partial relationships can be directly reflected in each dimension of user embedding without using burdensome node-wise attention-based models.

In light of this, we propose MEGCN (**M**ulti-channel **E**ncoding **G**raph **C**onvolutional **N**etwork), a partial relationship aware influence diffusion model, for social recommendation. The key idea behind this model is a feature-wise message computation with sparse regularization in the influence diffusion process. More specifically, the layer-wise diffusion process starts with an initial embedding for each user, which can be a free vector that captures the latent interest or any explicit features. Then a GCN-like information aggregation is conducted in each layer, which can help capture neighborhood contexts. At its core, under the assumption of channel-wise graph sparsity, the traditional node-wise message computation is changed into feature-wise computation, such that user interests and shared interests will be simultaneously captured in the layer-wise information diffusion process.

We summarize the contributions of this paper as follows:

- We propose MEGCN, a partial relationship aware influence diffusion model with channel-wise sparsity, to efficiently capture both self interests and share interests in the social recommendation tasks. In the information diffusion process, MEGCN realizes feature-wise message computation under the graph channel-wise sparsity assumption.
- We design two normalization schemes, InfluenceNorm and ChannelNorm, to guarantee channel-wise sparsity, such that they can model partial relationships and alleviate the over-smoothing problem in graph convolution networks.
- We demonstrate the effectiveness of our model on two real world datasets. Experimental results show that MEGCN can achieve comparable recommendation performance to graph attention-based models with much less computational cost.

2 PRELIMINARIES

2.1 Problem Formalization

Definition 1 (**partial relationship**). *Unlike graph-based single-label learning tasks, a social recommendation task fundamentally involves a social network as well as a user-item interaction graph. Then, each user from the user-item interaction graph can be regarded as a node with multiple labels (i.e. items), which forms various label-substructures when combined with social network structures. Consequently, self interests and shared interests can simultaneously propagate along the label-induced sub social networks structures, which are named as partial relationships between users.*

Definition 2 (**graph with channel-wise sparsity**). *A graph has channel-wise sparsity when element-wise embedding multiplication between two neighbors results in a sparse embedding, of which non-zero dimensions form information passing channels. As a result, the sparse partial relationships can be modeled via using the sparse channels.*

In general, a social network can be described as a set of nodes and links, with nodes representation matrix X and adjacency matrix A containing link weights. The degree matrix $D_{ii} = \sum_j A_{ij}$ represents the sum of all link weights to node i. In consistent with most studies, identity matrix I is used in the graph Laplacian operation and help realize self-loop operation on the graph.

In a social recommendation system, there are two sets of entities (a user set U and an item set V) and two graphs (a user-item interaction graph R and a social network S). Note that the interaction graph R can be an explicit or implicit feedback-based rating graph. In particular, in an implicit user-item interaction scenario (e.g. purchasing an item, voting for a song), we let r_{ai} denote the link weight between user a and item i, and r_{ai} is 1 if user a interacted with item i, otherwise r_{ai} is 0. We let d represent the number of channels or dimensions of an entity embedding, M denote the number of users in social network and N denote the total number of links between users.

Besides, the associated feature matrix P and Q of users (e.g, user profile) and items (item text representation, item visual representation)) are usually provided. Then, the recommendation problem can be formalized as a link prediction task on the graph R: given R, S, P and Q, a social recommendation aims to predict users' unknown interests to items in R: $\tilde{R} = f(R, S, P, Q)$. Given the problem definition, we introduce the preliminaries that are closely related to our proposed model.

Note that social recommendation is a typical graph-based multi-label learning task with one user interacts with multiple items, which needs to model label-induced sub-relations among users. As explained before, our work aims to model such sparse relationships via channel-wise sparsity. After splitting the embedding of user u into unit chunks [28], each chunk or channel of latent vector X_u can be regarded as an independent information passage in user u's ego network S_u, thus modeling the label-substructure relationships in the social network. Our work models such relationships in the aforementioned function f. In the rest of the paper, we use $\odot$ to represent element-wise multiplication.

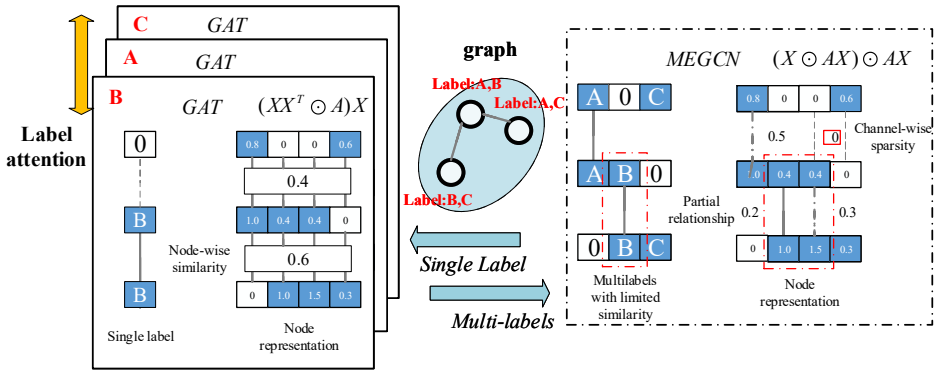


Figure 2: The MEGCN model versus GAT model. The motivation of MEGCN comes from two aspects: 1) GAT-based models are mostly suitable for graph learning tasks with single label. 2) GAT-based models calculate node-wise similarities by dot product, which are computationally expensive. On the left, we show GAT-based models need multiple stacking in multi-label learning tasks. On the right, we show the channel-wise sparsity of MEGCN realizes a lightweight version of GAT models while being applicable to multi-label learning tasks via modeling label-substructures. Channel-wise similarity is calculated according to element-wise multiplication.

2.2 Graph Convolutional Network

Graph Convolutional Network [3] models the graph message aggregation process as spectral convolution, in which node representations $g_\theta = diag(\theta)$ are transformed into Fourier domain with a filter parameterized by $\theta \in R^d$:

$$g_\theta * x = U g_\theta U^T x, \tag{1}$$

where U is formed by eigenvectors of normalized graph Laplacian matrix $L = I_N - D^{-\frac{1}{2}} A D^{-\frac{1}{2}} = U \Lambda U^T$. However, eigenvalue decomposition is computationally expensive. Therefore, it was suggested that g_θ could be approximated by a truncated expansion in terms of Chebyshev polynomials $T_k(x)$ up to K-th order [12]:

$$g'_\theta * x \approx \sum_{k=0}^{K} \theta'_k T_k(\hat{\Lambda}), \tag{2}$$

where $\hat{\Lambda} = \frac{2}{\lambda_{max}} \Lambda - I_N$, λ_{max} denotes the maximum eigenvalue of L, and θ'_k is the vector of Chebyshev coefficients. In particular, when $\lambda_{max} \approx 2$ and $K = 1$, we obtain the first-order form of GCN [3]:

$$g'_\theta * x \approx \theta'_0 x - \theta'_1 (L - I^N) x = \theta'_0 x - \theta'_1 D^{-\frac{1}{2}} A D^{-\frac{1}{2}} x, \tag{3}$$

where free parameters θ'_0 and θ'_1 are shared over the whole graph. In effect, GCN constrains the number of parameters to prevent overfitting and normalizes the updated node vector to alleviate numerical instability problems (i.e. the renormalization trick $I_N + D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \rightarrow \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}}$, with $\hat{A} = A + I_N$ and $\hat{D}_{ii} = \sum_j \hat{A}_{ij}$). The final update function for GCN can be formulated as:

$$X_{updated} = \hat{D}^{-\frac{1}{2}} \hat{A} \hat{D}^{-\frac{1}{2}} X \Theta, \tag{4}$$

which can be regarded as an average on the embeddings of first-order neighbors and the central node. This can easily make nodes indistinguishable and thus cause the oversmoothing problem.

2.3 Graph Attention Network

Graph Attention Network [20] changes the operations of graph aggregation and combination in GCN to a weighted summation form by performing self-attention on each connected node pair. The weight between a center node u and a neighbor node s can be measured by:

$$c_u^s = sim(\mathbf{W} X_u, \mathbf{W} X_u^s), \tag{5}$$

where sim is the similarity measure function, such as cosine similarity in most cases. The coefficients are then normalized by $softmax$ function to constrain the transformed value to avoid numerical instability problems:

$$\alpha_u^s = \frac{exp\ c_u^s}{\sum_{\hat{s}=0}^{S} exp\ c_u^{\hat{s}}}. \tag{6}$$

Then, the updated node embedding in a graph attention layer can be formulated as:

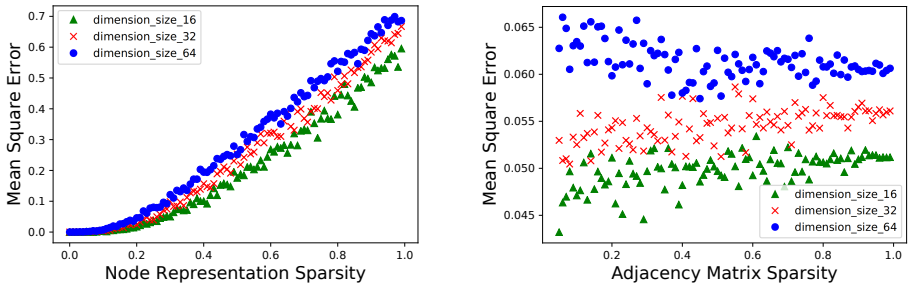
$$X_{updated} = softmax(XX^T \odot A) X \Theta. \tag{7}$$

2.4 Multi-channel Encoding Scheme

Graph Attention Network utilizes cosine similarity metric to measure the importance weight of the first order neighbors. More formally, given central user node u's representation X_u and one of its first-order neighbor node s's representation X_u^s, the similarity score can be calculated as:

$$sim(X_u, X_u^s) = X_u \cdot X_u^s = \sum_{i}^{d} (X_u(i) X_u^s(i)), \tag{8}$$

where s denotes one of its neighbors and d is the number of dimensions or channels. After applying the above operation to all first-order neighbors, the similarity values of all neighbor nodes are normalized by $softmax$ activation function to obtain the final node-wise linking weight.



(a) (b)

Figure 3: Mean squared error comparisons between MEGCN and GAT model on different dimension sizes. As shown in (a), when a graph has channel-wise sparsity, the output of the two models are indistinguishable. As for (b), when channels carry enough sparse information to distinguish different neighbors, the output of the two models are indistinguishable.

However, the method is computationally expensive due to multiple times of pairwise similarity computations. For instance, for those nodes with hundreds of neighbors, the computational cost is unacceptable. With channel-wise sparsity assumption, we could simplify GAT while modeling label-substructure relationships on the social network structure. When graph attention operations are performed on nodes of graph with sparse channel representations, pairwise node similarity can be approximated as:

$$sim(X_u, X_u^s) \approx \sum \{X_u(i)X_u^s(i), i \in \{topk\}\}$$
$$\forall j \notin \{topk\} \quad |X_u(j)X_u^s(j)| << |X_u(i)X_u^s(i)|, \tag{9}$$

where the channels set $\{topk\}$ contains the channels with the largest element-wise multiplication values between u and s. This approximation mainly depends on channel-wise sparsity between two adjoint nodes, for example, when channel is sparse enough, the equation can be simplified into a single multiplication operation without summation parts. Under such circumstances, linking weight between user u and s can be approximated by:

$$\alpha_u^s \approx \frac{exp\ X_u(i)X_u^s(i)}{\sum_{\hat{s}=0}^{S} exp\ (X_u(i)X_u^{\hat{s}}(i))}. \tag{10}$$

Then, the node-wise multiplication between nodes are simplified into element-wise multiplication, and weighted aggregation in GAT can be transformed into:

$$X_{updated} = normalization(X \odot AX) \odot AX\Theta. \tag{11}$$

Note that the above normalization operation is performed in a channel-wise way while original graph attention mechanism realizes message computation, aggregation and update in a node-wise way. In essence, they have the same mathematical expression. However, the channel-wise sparsity ensures that a few channels are sufficient for importance measurement, such that we can perform message computation after message aggregation instead of before it, which largely decreases the computational complexity of original GAT-based models. Besides, the equation can be regarded as controlling influence propagation on only some of the channels, which reflects partial relationships among users. In particular, it can be regarded as a chunk-based aggregation with chunk size 1, where only limited chunks (channels) transfer information. The framework of MEGCN is shown in Figure 2.

2.5 Sanity Check and Discussions

To show our model's motivations, we test the channel-wise sparsity assumption on handcrafted datasets. First, we generate 100 nodes with each embedding dimension following normal Gaussian distribution. Without the loss of generality, the dimension size is set in the range [16, 32, 64]. Then, we form the connected network structure by generating a random variable that follows the uniform distribution. As long as the value of the variable is larger than a sparse rate, which is defined as the percentage of non-zero dimensions of the original node embedding, we add a link between two nodes. Note that channel-wise sparsity in our work means sparsity along a few channels after element-wise multiplication of connected nodes embeddings. Therefore, it is hard to directly generate sparsity after multiplication operation. Instead, we use node representation sparsity to approximate channel-wise sparsity. In particular, we test the effects of different node representation sparsity and adjacency matrix sparsity to show how they affect the results.

As shown in Figure 3(a), the mean squared error difference between MEGCN output and GAT output with the same sparse node representation is minimal when node sparsity percentage is very low. The graph and node embeddings are generated with adjacency matrix sparsity of 0.2, and each node has around 20 neighbors. As for the number of neighbors, when the number of linking channels in each pair is larger than the number of neighbors, GAT and MEGCN achieve the similar performance. The results are shown in Figure 3(b) with node representation sparsity rate of 0.2. However, in most cases, the number of channels can be far less than the number of neighbors, which makes it hard to distinguish node in each channel. In response, we could use neighborhood sampling techniques to help limit the channel-wise information confusion.

To this end, we could adopt the same ideas on the social network-based recommendation system, which is a multi-label learning task with extreme label sparsity. The label-substructures induce the partial relationships among users in the original social network. We need develop extra some normalization approaches to ensure the channel-wise sparsity such that personal interests and shared interests can simultaneously propagate along the sparsity channels. We hope the normalization function in Eq. (11) can generate sparse channel links, which requires normalization functions shrink most channel values to zeros while enlarging the non-zero ones. Hence,

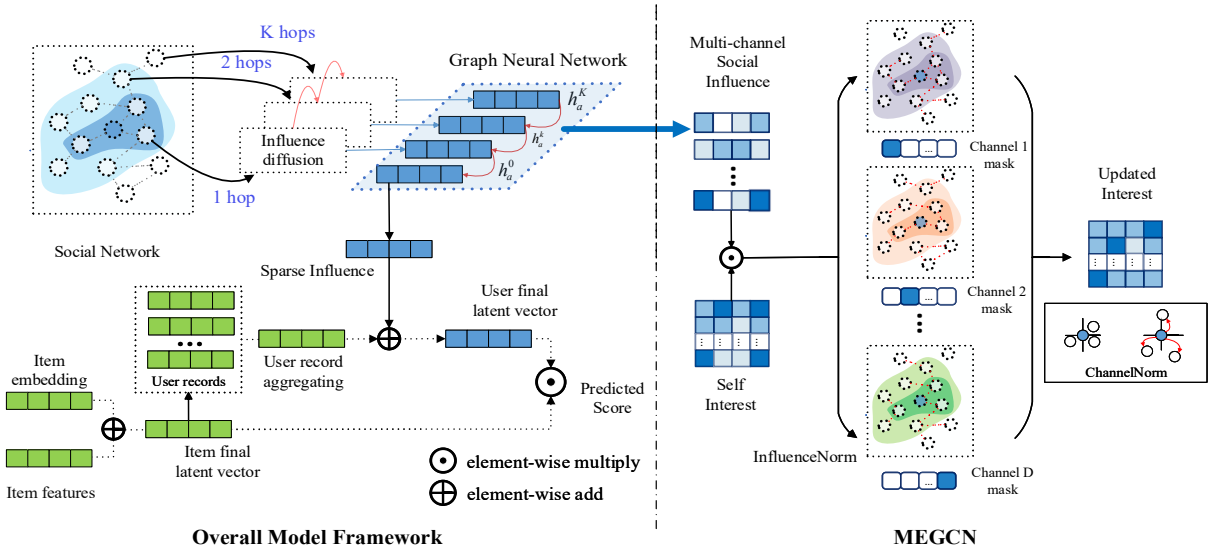


Figure 4: Illustration of the proposed model. The left subfigure shows the overall model framework and the right subfigure presents MEGCN in the influence diffusion process. Especially for the MEGCN, sparse influence is utilized to transform multi-channel influence with channel-wise sparsity from users. Note that normalization mask can be regarded as a kind of interests separation on social network to model partial relationships, which divides social network into substructures of interest with sparse links in each channel. In particular, MEGCN captures multi-channel feature similarity with element-wise product operation. InfluenceNorm is proposed to guarantee the sparsity of social influence. Moreover, the ChannelNorm is designed to balance channel values, which prevents numerical problems and increases node-wise distinction to alleviate the oversmoothing problem.

the equation can generate a sparse update on user representation and further lead to channel-wise sparsity. In general, *softmax* function is regarded as useful normalization function. However, after updating a user's representation, values among channels will be imbalanced due to the nonlinear normalization nature of *softmax* function. As a result, it can further lead to gradient explosion or vanishing on some linking channels, as well as overfitting problems. Therefore, we should normalize the values in each channel of the node representation after each update step.

3 THE PROPOSED MODEL

3.1 Model Framework

We introduce MEGCN into the social recommend system and the overall framework is shown in Figure 4. The model is composed of five Layers: Embedding Layer, Fusion Layer, User Action Aggregating Layer, Influence Diffusion Layer, Prediction Layer. We detail each parts as follows:

Embedding Layer. For each user or item, the original discrete one-hot encoding generates extreme feature sparsity. Hence, we use an embedding layer to encode users and items with corresponding continuous values. Formally, given the one-hot representations of a user or an item, the embedding layer performs an index operation on free user embedding matrix X or item embedding matrix Y. For instance, user a's free latent embedding is obtained as:

$$X_a = Embedding(ID_a) = W_{emb} ID_{onehot}. \tag{12}$$

Fusion Layer. Besides free embedding, items and users have their associated side information, e.g. text description, which can be regarded as feature embedding. Fusion layer takes free embedding X, Y (X_a for user a or Y_i for item i) and features embedding P, Q (P_a for user a or Q_i for item i) as input, and outputs an initial user fusion embedding h_a^0 or an item fusion embedding v_i via a fully connected neural network. For instance, the initial fusion embedding for user a is calculated as:

$$h_a^0 = \sigma\left(\mathbf{W}\left[X_a, P_a\right] + \mathbf{b}\right), \tag{13}$$

where $\mathbf{W}$ and $\mathbf{b}$ are the parameters that need to be learned, and σ is the activation function. Obviously, by setting $\mathbf{W}$ and $\mathbf{b}$ as an identity matrix and zero vector, respectively, we could simplify the fusion layer into a concatenation operation. Similarly, we can obtain the item fusion embedding v_i for item i.

User Action Aggregating Layer. We capture a user's behavior encoding by mean pooling on the fusion embedding of items the user has interacted with in the past. Note that the method was firstly proposed in SVD++ [10]. For instance, we obtain user a's behavior encoding w_a by:

$$w_a = \text{Pool}\left(\{\mathbf{v}_b | b \in R_a\}\right), \tag{14}$$

where v_b is the item fusion vector for item b in user action history R_a, and w_a denotes behavior encoding of user a learned from history actions.

Influence Diffusion Layer. By feeding user interest into the influence diffusion layer, we model the propagation dynamics of users' interest in social network S. For each user a, we use h_a^k to represent interest after k hops of dynamic influence diffusion in the social network. Each diffusion layer contains three operations:

selecting neighbors to diffuse information, aggregating neighbor influence and combining self interest with neighbors' influence [30]. To capture partial relationships among users in the social network, we propose MEGCN in the information diffusion process. In MEGCN, we use channel-wise similarity to measure the importance of social influence in each channel from a neighbor. Each dimension of h_a can be regarded as a channel of an independent interest. For instance, the interest of color and shape could be stored in the first and second channel respectively. Besides, we assume users' interests on some channels are inherently stable, which means users will selectively absorb information from neighbors with partial relationships. Therefore, with sparse channel representation, we could capture interest similarity on different channels between connected nodes. The proposed model puts the message computation part after message aggregation part instead of before it in other GNN models.

- **Aggregating**. The model can be regarded as a subgraph information diffusion on each channel, where nodes distinguish from each other by controlling the sparse linking rate. We perform a linear transformation on aggregated information from neighbors $\tilde{a}$ in user's ego network S_a to obtain neighbor influence vector $\boldsymbol{c}^k$:

$$c^k = \mathbf{W}_a^k \text{ aggregate } \left(\{ h_{\tilde{a}}^k | \tilde{a} \in S_a \} \right) + \mathbf{b}_a^k. \tag{15}$$

- **InfluenceNorm**. InfluenceNorm first conducts element-wise multiplication to identify important channels from the neighborhood influence vector. Then, after normalization, InfluenceNorm generates a sparse channel-wise mask that determines importance weight of each channel in different label-substructure relations. In effect, inspired by graph attention network, which uses *softmax* to normalize the importance weight between each node pair, we use *softmax* as the normalization function. The sparse influence is computed as:

$$h_a^k = \sigma \left(c^k \odot softmax \left(h_a^{k-1} \odot c^k \right) + \alpha c^k + h_a^{k-1} \right). \tag{16}$$

  We add a residual part αc^k to avoid gradient vanishing problem and a hyperparameter α to control the weight.
- **ChannelNorm**. After performing *softmax* operation in InfluenceNorm, we obtain node embeddings with highly imbalanced channel values. Hence, the model may face numerically unstable problems in some channels, which will lead to a local minimum. Meanwhile, channel-wise sparsity will divide the original social network into multi-channel label-substructures, i.e. partial relationships, which can be better modeled with a larger substructure diversity or channel-wise distinction. Hence, we adopt the ideas from PairNorm [41] to control the value imbalance between channels, and propose ChannelNorm:

$$h_a^k = \gamma \odot \delta \odot h_a^k / \text{var}(h_a^k) + \beta, \tag{17}$$

  where γ and β represent trainable channel-wise scale and shift vectors in ChannelNorm operation, and δ is the hyperparameter that controls the normalized variance. We add such parameters for a better channel-wise distinction. After updating sparse influence with ChannelNorm, users preserve their own interests while accepting sparse influence from neighbors, which can preserve necessary distinction between each node pair, thus alleviating the oversmoothing problem.

Prediction Layer. After K hops of social influence propagation, we obtain each user's final embedding by combining information via influence diffusion and user action aggregating. The final predicted rating is measured by the multiplication of the two latent vectors.

$$\tilde{\mathbf{r}}_{ai} = \mathbf{v}_i^T u_a = v_i^T \left(h_a^K + w_a \right). \tag{18}$$

Loss Function. We use the same pair-wise ranking-based loss function in [16, 30]:

$$\min_{\Theta} \mathcal{L}(R, \tilde{R}) = \sum_{a=1}^{M} \sum_{(i,j) \in D_a} \sigma \left(\tilde{r}_{ai} - \tilde{r}_{aj} \right) + \lambda \left(\|X\|^2 + \|Y\|^2 \right), \tag{19}$$

where $D_a = \{(i, j) | i \in R_a \wedge j \in V - R_a\}$ denotes the pairwise training data for user a with user history action set R_a and $\sigma(x)$ is the sigmoid function,

3.2 Training

In this section, we introduce some implementation details in the model training part.

3.2.1 Mini-Batch Training. In practice, to avoid huge computational consumption, we divide training data into batches following the procedure that each user's training records are ensured in the same mini-batch, which helps avoid the repeated computation of each user a's latent embedding h_a^k in the iterative influence diffusion layers [30]. For every epoch, we shuffle the training set to generate different batches.

3.2.2 Negative Sampling. Since we only observe positive feedbacks in the original two datasets, we use the negative sampling technique to obtain pseudo negative feedbacks at each iteration in the training process, with the assumption that all items without implicit feedbacks in training data will have equal probability to be selected as a negative sample.

3.2.3 Dropout and Regularization. Since the MEGCN uses sparse influence to update users' interest, the model can easily run into a local minima. Hence, we use feature-wise dropout between different diffusion layers. To reduce the effects of channel information mixture when the number of neighbors is huge, we also apply dropout on adjacency matrix, which can be regarded as a sampling among neighbors, such that users update their interests based on a subset of neighbors. As for the regularization, we use L2-regularization on both user and item embeddings with the same regularization rate.

3.3 Discussion

3.3.1 Time Complexity Analysis. Compared with traditional recommendation models, the main additional time costs of the social recommendations occur in the layer-wise influence diffusion process. More specifically, linear transform operation in GCN only costs $O(MKd^2)$, where M is the number of users in graph, K is the step of influence diffusion, and d is the length of features or channels. MEGCN uses sparse influence to capture multi-channel distinction

Table 1: The statistics of the two datasets.

Dataset	*Yelp*	*Flickr*
Users	17237	8358
Items	38342	82120
Total Links	143765	187273
Ratings	204448	314809
Link Density	0.048%	0.268%
Rating Density	0.031%	0.046%

between social influence and self interests, which costs an additional time complexity $O(MKd)$ compared with GCN. However, GAT needs to computs an attention score between each pair of connected nodes, thus it is the most time consuming algorithm with computational complexity $O(M^2Kd^2)$ in full version and $O(MNKd^2)$ in sparse version (N denotes links number), which is around ten times larger than other models.

4 EXPERIMENTS

In this section, we evaluate our methods via the experiments on two real-world recommendation tasks.

4.1 Data Preparation

We evaluate the performance of our model on two recommendation datasets with social network: Yelp and Flickr. These two datasets are provided in DiffNet [30] and have been explained in detail. In the data preparation step, we filter out users with less than 2 historical action records and 2 social neighbors in both datasets. Then the datasets for experiments are built as follows. We randomly divide 10% of datasets as test set, 10% as validation set, and the rest 80% as the training set. The detail statistics of the two datasets after preprocessing are shown in Table 1.

4.2 Baselines

Since the social recommendation task involves social networks, we compare MEGCN with various state-of-the-art social network-based graph neural networks:

SVD++ [13]. SVD++ only utilizes user-item interaction graph without any other side information, and it can be seen as the baseline model for recommendation systems without social networks.

TrustSVD [10]. TrustSVD is the first model to use social network on recommendation systems. The model only aggregates one hop neighborhood information without other similarity assumption.

DiffNet [30]. Compared with TrustSVD, DiffNet incorporates multi-hop dynamic diffusion layers. The method uses first-order approximation of graph spectral network.

GCN [3]. Compared with Diffnet, GCN adds a renormalization trick on the aggregating operation.

GAT [20]. GAT leverages an attention layer to calculate the correlation score between two neighbor nodes and sets it as the rate of information aggregation.

DualGAT [31]. DualGAT extends social effects from user domain to item domain and leverages dual graph attention mechanism to collaboratively learn representations for static and dynamic social effects. Similar works can be found in [29] and [7]. In essence, their work can model item (label)-substructure by incorporating user-item interactions in the information diffusion process of the social network. Other works [42, 43] that utilize attention mechanism without social network effects are not shown in our experiments.

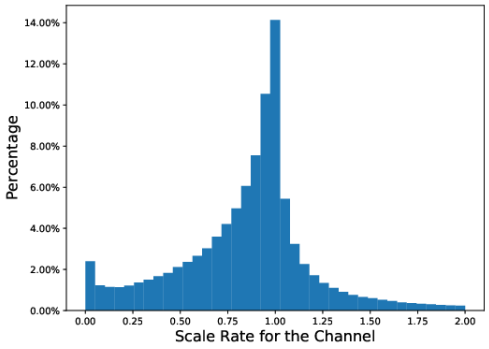


Figure 5: Channel-wise sparsity in *Flickr*. Most of channels are sparse to model partial relationships.

4.3 Evaluation Metrics

As our work focuses on recommending top-N items, we use two ranking-based evaluation metrics, which are Hit Ratio (HR) and Normalized Discounted Cumulative Gain (NDCG). The two metrics are defined as:

Hit Ratio measures the number of items that the user likes in the test data that has been successfully predicted in the top-N ranking list. It can be given as:

$$HR = \frac{HitNumber}{N}. \tag{20}$$

Normalized Discounted Cumulative Gain considers the hit positions of the items and gives a higher score if the hit items in the top positions [19].

$$NDCG = \frac{DCG}{IDCG}. \tag{21}$$

4.4 Results Summary

4.4.1 Parameter Setting. For all the models that are based on the latent factor models, we initialize the latent vectors with small random values. In the model learning process, we use Adam as the optimizing method for gradient descent based methods with an initial learning rate of 0.001. And the batch size is set as 512. For the parameter settings, we set the embedding dimension size as 64. Moreover, we adjust some baseline models to make sure all the models have consistent embedding dimensions. In our model, there are four hyperparameters to be fine-tuned. Learning rate lr is selected from $[1e-1, 1e-2, 1e-3, 1e-4, 1e-5]$, normalization variance δ is searched in the range of $[1, 2, 4, 8]$, l2 normalization rate λ is selected from $[1e-2, 2e-2, 3e-2, 4e-2]$, and residual rate α is searched in the range of $[0.1, 0.2, 0.3, 0.4, 0.5]$. We fine-tune those hyperparameters to get the best results on validation set. The values of two metrics in the test datasets are obtained after 400 epochs.

4.4.2 Overall Comparison. Table 2 compares the average HR and NDCG of the benchmarks with our MEGCN model. First, we can observe that our model outperforms others in all two datasets, which

Table 2: HR@N and NDCG@N comparison on two datasets.

Models	*Yelp*						*Flickr*					
	HR@N			NDCG@N			HR@N			NDCG@N		
	N=5	N=10	N=15	N=5	N=10	N=15	N=5	N=10	N=15	N=5	N=10	N=15
SVD++	0.1868	0.2831	0.3492	0.1389	0.1711	0.1924	0.0827	0.1054	0.1257	0.0753	0.0825	0.0895
TrustSVD	0.1906	0.2915	0.3693	0.1385	0.1738	0.1983	0.1072	0.1472	0.1741	0.0970	0.1085	0.1200
DiffNet	0.2346	0.3497	0.4292	0.1695	0.2109	0.2348	0.1226	0.1622	0.1949	0.1143	0.1284	0.1388
GCN	0.2348	0.3378	0.4124	0.1751	0.2124	0.2350	0.1269	0.1628	0.1972	0.1173	0.1293	0.1406
GAT	0.2478	0.3478	0.4156	0.1918	0.2282	0.2487	0.1306	0.1655	0.199	0.1193	0.1312	0.1422
DualGAT	**0.2591**	0.3666	0.4328	0.1982	0.2371	0.2573	**0.1326**	0.1667	0.2001	**0.1215**	0.1330	0.1443
MEGCN	0.2590	**0.3685**	**0.4332**	**0.2012**	**0.2394**	**0.2590**	0.1302	**0.1688**	**0.2053**	0.1208	**0.1344**	**0.1460**

Table 3: HR@15 and NDCG@15 performance with different diffusion depth k.

Diffusion Depth k	*Yelp*				*Flickr*			
	HR@15	Improve.	NDCG@15	Improve.	HR@15	Improve.	NDCG@15	Improve.
k=2	**0.4332**	-	**0.2590**	-	0.2053	-	0.1460	-
k=3	0.4282	-1.15%	0.2572	-0.69%	0.2055	+0.09%	**0.1465**	+0.34%
k=4	0.4279	-1.22%	0.2569	-0.71%	**0.2068**	+0.54%	0.1458	-0.13%

indicates that channel-wise sparsity-based model performs better than traditional graph attention models. Second, the performance of GCN is better than DiffNet in most cases, which indicates that the stacking of layers with more parameters can enhance the performance of the model. Third, we find GAT models outperform other Laplacian Smoothing-based models, which proves the importance of distinguishing different neighbors. In addition, dualGAT achieves the best performance among baseline models since it considers both static and dynamic social influence.

4.4.3 Partial Relationship Analysis. To find out how channel-wise sparsity in MEGCN helps improve model performance, we conduct extra experiments using *Flickr* dataset. We obtain users' representations after two hops of influence diffusion based on our model. As shown in Figure 5, about 30% of channels perform like GCN with scale weight around 1, and the others are typical MEGCN outputs with sparse influence update. Based on our assumption in equation 10, they model partial relationships with enough channel-wise sparsity. As a result, MEGCN assigns different weights to different channels for a better recommendation performance.

4.4.4 Oversmoothing Problem. We compare models performance after multi-hop influence diffusion. One of the disadvantages of GCN models is the oversmoothing problem, which refers to the phenomenon that as nodes dynamically aggregate information from their neighbors, nodes will become similar with each other after multiple hops of diffusion. We test our model with three ascending diffusion depths, which are shown to have increasingly negative effects on recommendation performance in [1]. Nevertheless, as illustrated in Table 3, our model can achieve consistent prediction performance even after many hops of information aggregation, which shows that our model can alleviate oversmoothing problem in the dynamic influence diffusion process with InfluenceNorm and ChannelNorm. In sum, the model can separate multi-channel interests in the social influence context.

4.4.5 Time Cost and Channel Independence. We list the feature correlation scores and training/testing time costs for different models in Table 4. On one hand, we can observe that MEGCN achieves around 10 times computational efficiency compared with advanced attention-based GNN models, such as GAT and DualGAT, while generating a similar lower correlation score. It indicates MEGCN better models partial relationship in different channels. On the other hand, it displays similar time costs compared with traditional GCN models. In particular, the feature-wise correlation matrix of user interests generated by DiffNet and MEGCN are shown in Figure 6. The lighter color heatmap directly shows the lower correlation score between features via MEGCN model. Better channel independence stands for a more reliable partial relationship. Both MEGCN and DualGAT achieve a good performance on such metric, which indicate that MEGCN can model partial relationship like GAT-based models.

5 RELATED WORK

5.1 Social Recommendation

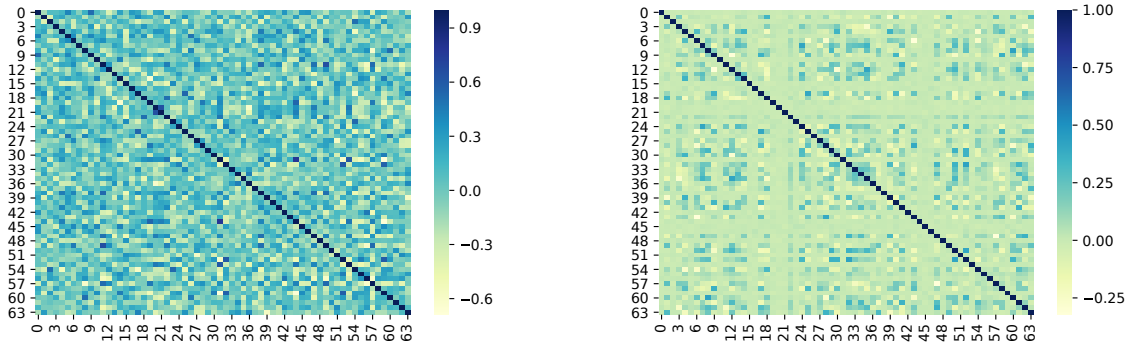
Social effects have been widely exploited in recommendation systems to solve data sparsity problems. Fundamentally, current studies can be divided into similarity-based methods and model-based methods. Especially, most current methods leverage deep neural networks [2, 6, 21, 26, 30, 39] to capture nonlinear information. However, they either assume social influence dominates the user behavior or are too computationally expensive. Note that, besides direct social effects, various peer relations [24, 34] can be constructed to improve representation learning and recommendation performance.

5.2 Graph Neural Networks

Traditional graph embedding methods [8, 9, 17] and original Graph Convolution Network [3] mainly focus on embedding nodes from a single fixed graph, thus lacking the ability to generalize to unseen nodes. Then, [11, 22, 36] proposed to aggregate graph information in an inductive way, and the dynamic structure and feature information aggregation mechanism have helped achieve impressive performance in graph-involved tasks. However, in essence, Graph Convolution Network is a special form of Laplace Smoothing [14], which is a spectrum method that constrains neighborhood to be

Table 4: Training time per epoch and average absolute feature correlation score for different models.

Models	DiffNet	GCN	GAT	DualGAT	MEGCN
training time per epoch	**0.8956**s	0.9135s	9.9444s	11.0342s	0.9834s
testing time per epoch	**0.1297**s	0.1306	0.6254	0.6632	0.1496
average absolute feature correlation	0.1791	0.1140	0.0736	**0.0715**	0.0723



(a) correlation matrix of user interest features in DiffNet

(b) correlation matrix of user interest features in MEGCN

Figure 6: Comparison on correlation matrix of user interest features between DiffNet and MEGCN. Lower correlation score stands for higher independence and distinction for feature-wise user interest. In our experiments, user embedding has 64 dimensions.

similar. Therefore, it is not appropriate to directly utilize GCN and its variants [11, 36] directly in social recommendation task since they tend to attach equal importance weights to neighbors. One possible solution is to utilize graph attention network [20] as it can learn the latent embeddings of each node by attending to its neighbors following a self-attention strategy. Nevertheless, it is computationally expensive in large scale networks. Besides, various works have proposed to leverage substructures [22, 23, 25] information in graph-based learning models.

5.3 Multi-label Learning

Multi-label learning [40] has been widely applied to different machine learning tasks, such as text categorization [15, 18], image annotation [27]. However, in most cases, a multi-label learning task faces empty relevant label problem [32] and then model performance can be largely affected when incomplete multi-labels are used for learning. [5] proposes to handle such problems by learning from semi-supervised weak-label data. Inspired by [5], since graph-based models are typical semi-supervised, multi-task learning can be realized according to graph-based models. [37] is one of the earliest works applying multi-label learning on recommendation tasks.

6 CONCLUSION

In this paper, we propose MEGCN, a graph neural network based on channel-wise sparsity. MEGCN simplifies GAT operation and utilizes two models, InfluenceNorm and ChannelNorm, to capture both self interest and shared interest in the influence diffusion process of social recommendation task. Essentially, our work can model the sparse label-induced structures in the original social network, namely partial relationship, without suffering from expensive computational cost of graph attention based models. Finally, experimental results validate the effectiveness of the proposed models. In particular, the MEGCN achieves the highest HR and NDCG score on all datasets. Future research will be conducted from both theoretical and practical perspectives. Theoretically, we will explore how to further improve the performance of the MEGCN models from the optimization point of view. Besides, since graph sub-structure has been validated to be useful for graph modeling [22, 25], we will explore the usefulness of such information in the social recommendation task. In practice, we will apply the method to other graph-based multi-label tasks and test whether it it applicable for all kinds of tasks.

ACKNOWLEDGMENTS

This work was supported by National Key R&D Program of China (No. 2018YFC0910500), National Natural Science Foundation of China (No. 61772110,61877008,71731003), Science Foundation of Ministry of Education of China & China Mobile (No. MCM20170507) and Program of Introducing Talents of Discipline to Universities (Plan 111) (No. B20070)

REFERENCES

[1] James Atwood and Don Towsley. 2016. Diffusion-convolutional neural networks. In *Advances in neural information processing systems*. 1993–2001.

[2] Enhong Chen, Guangxiang Zeng, Ping Luo, Hengshu Zhu, Jilei Tian, and Hui Xiong. 2017. Discerning individual interests and shared interests for social user profiling. *World Wide Web* 20, 2 (2017), 417–435.

[3] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*. 3844–3852.

[4] Kien Do, Truyen Tran, Thin Nguyen, and Svetha Venkatesh. 2019. Attentional multilabel learning over graphs: a message passing approach. *Machine Learning* 108, 10 (2019), 1757–1781.

[5] Hao-Chen Dong, Yu-Feng Li, and Zhi-Hua Zhou. 2018. Learning From Semi-Supervised Weak-Label Data. In *Proceedings of the AAAI Conference on Artificial Intelligence(AAAI)*. 2926–2933.

[6] Wenqi Fan, Qing Li, and Min Cheng. 2018. Deep Modeling of Social Relations for Recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence(AAAI)*. AAAI press, 8075–8076.

[7] Wenqi Fan, Yao Ma, Qing Li, Yuan He, Yihong Eric Zhao, Jiliang Tang, and Dawei Yin. 2019. Graph Neural Networks for Social Recommendation. In *Proceedings of the 2019 World Wide Web Conference(WWW)*. 417–426.

[8] Shanshan Feng, Gao Cong, Arijit Khan, Xiucheng Li, Yong Liu, and Yeow Meng Chee. 2018. Inf2vec: Latent representation model for social influence embedding. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*. IEEE, 941–952.

[9] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable Feature Learning for Networks. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining(KDD)* 2016 (2016), 855–864.

[10] Guibing Guo, Jie Zhang, and Neil Yorke-Smith. 2015. TrustSVD: Collaborative Filtering with Both the Explicit and Implicit Influence of User Trust and of Item Ratings. In *Proceedings of the AAAI Conference on Artificial Intelligence(AAAI)*, Vol. 15. 123–125.

[11] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Proceedings of the Conference on Neural Information Processing Systems(NIPS)*. 1024–1034.

[12] David K Hammond, Pierre Vandergheynst, and Rémi Gribonval. 2011. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis* 30, 2 (2011), 129–150.

[13] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining(KDD)*. 426–434.

[14] Qimai Li, Zhichao Han, and Xiao-Ming Wu. 2018. Deeper Insights into Graph Convolutional Networks for Semi-Supervised Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence(AAAI)*. Association for the Advancement of Artificial Intelligence, 3538–3545.

[15] Kamal Nigam, Andrew McCallum, Sebastian Thrun, and Tom Michael Mitchell. 2004. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning* 39 (2004), 103–134.

[16] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2012. BPR: Bayesian personalized ranking from implicit feedback. *arXiv preprint arXiv:1205.2618*.

[17] Leonardo FR Ribeiro, Pedro HP Saverese, and Daniel R Figueiredo. 2017. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining(KDD)*. ACM, 385–394.

[18] Naonori Ueda and Kazumi Saito. 2003. Parametric Mixture Models for Multi-Labeled Text. In *Proceedings of the Conference on Neural Information Processing Systems(NIPS)*. 737–744.

[19] Hamed Valizadegan, Rong Jin, Ruofei Zhang, and Jianchang Mao. 2009. Learning to Rank by Optimizing NDCG Measure. In *Proceedings of the Conference on Neural Information Processing Systems(NIPS)*. 1883–1891.

[20] Petar Veli$\check{c}$ković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *Proceedings of International Conference on Learning Representations(ICLR)* (2018).

[21] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2019. Exploring High-Order User Preference on the Knowledge Graph for Recommender Systems. *ACM Trans. Inf. Syst.* 37 (2019), 32:1–32:26.

[22] Pengyang Wang, Yanjie Fu, Hui Xiong, and Xiaolin Li. 2019. Adversarial substructured representation learning for mobile user profiling. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining(KDD)*. 130–138.

[23] Pengyang Wang, Yanjie Fu, Jiawei Zhang, Xiaolin Li, and Dan Lin. 2018. Learning urban community structures: A collective embedding perspective with periodic spatial-temporal mobility graphs. *ACM Transactions on Intelligent Systems and Technology (TIST)* 9, 6 (2018), 1–28.

[24] Pengyang Wang, Yanjie Fu, Jiawei Zhang, Pengfei Wang, Yu Zheng, and Charu Aggarwal. 2018. You are how you drive: Peer and temporal-aware representation learning for driving behavior analysis. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining(KDD)*. 2457–2466.

[25] Pengyang Wang, Yanjie Fu, Yuanchun Zhou, Kunpeng Liu, Xiaolin Li, and Kien Hua. 2020. Exploiting Mutual Information for Substructure-aware Graph Representation Learning. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence(IJCAI)*. 3415–3421.

[26] Xiang Wang, Xiangnan He, Liqiang Nie, and Tat-Seng Chua. 2017. Item silk road: Recommending items from information domains to social users. In *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval(SIGIR)*. ACM, 185–194.

[27] Jason Weston, Samy Bengio, and Nicolas Usunier. 2011. WSABIE: Scaling Up to Large Vocabulary Image Annotation. In *Proceedings of International Joint Conference on Artificial Intelligence(IJCAI)*.

[28] Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. 2019. Pay Less Attention with Lightweight and Dynamic Convolutions. In *Proceedings of International Conference on Learning Representations(ICLR)*.

[29] Le Wu, Junwei Li, Peijie Sun, Richang Hong, Yong Ge, and Meng Wang. 2020. DiffNet++: A Neural Influence and Interest Diffusion Network for Social Recommendation. *ArXiv* abs/2002.00844 (2020).

[30] Le Wu, Peijie Sun, Yanjie Fu, Richang Hong, Xiting Wang, and Meng Wang. 2019. A Neural Influence Diffusion Model for Social Recommendation. In *Proceedings of the 42nd International ACM SIGIR conference on Research and Development in Information Retrieval(SIGIR)*.

[31] Qitian Wu, Hengrui Zhang, Xiaofeng Gao, Peng He, Paul Weng, Han Gao, and Guihai Chen. 2019. Dual Graph Attention Networks for Deep Latent Representation of Multifaceted Social Effects in Recommender Systems. In *Proceedings of the 2019 World Wide Web Conference(WWW)*.

[32] Xi-Zhu Wu and Zhi-Hua Zhou. 2017. A Unified View of Multi-Label Performance Measures. *ArXiv* abs/1609.00288 (2017).

[33] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S. Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph Contextualized Self-Attention Network for Session-based Recommendation. In *Proceedings of International Joint Conference on Artificial Intelligence(IJCAI)*.

[34] Tong Xu, Hengshu Zhu, Hui Xiong, Hao Zhong, and Enhong Chen. 2019. Exploring the Social Learning of Taxi Drivers in Latent Vehicle-to-Vehicle Networks. *IEEE Transactions on Mobile Computing* (2019).

[35] Yanan Xu, Yanmin Zhu, Yanyan Shen, and Jiadi Yu. 2019. Learning Shared Vertex Representation in Heterogeneous Graphs with Convolutional Networks for Recommendation. In *Proceedings of International Joint Conference on Artificial Intelligence(IJCAI)*.

[36] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L. Hamilton, and Jure Leskovec. 2018. Graph Convolutional Neural Networks for Web-Scale Recommender Systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining(KDD)*.

[37] Dong Zhang, Shu Zhao, Zhen Duan, Jie Chen, Yanping Zhang, and Jie Tang. 2020. A multi-label classification method using a hierarchical and transparent representation for paper-reviewer recommendation. *ACM Transactions on Information Systems (TOIS)* 38, 1 (2020), 1–20.

[38] Jiani Zhang, Xingjian Shi, Shenglin Zhao, and Irwin King. 2019. STAR-GCN: Stacked and Reconstructed Graph Convolutional Networks for Recommender Systems. In *Proceedings of International Joint Conference on Artificial Intelligence(IJCAI)*.

[39] Jiawei Zhang, Congying Xia, Chenwei Zhang, Limeng Cui, Yanjie Fu, and S Yu Philip. 2017. BL-MNE: emerging heterogeneous social network embedding through broad learning with aligned autoencoder. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 605–614.

[40] Min-Ling Zhang and Zhi-Hua Zhou. 2014. A Review on Multi-Label Learning Algorithms. *IEEE Transactions on Knowledge and Data Engineering(TKDE)* 26 (2014), 1819–1837.

[41] Lingxiao Zhao and Leman Akoglu. 2020. PairNorm: Tackling Oversmoothing in GNNs. In *Proceedings of International Conference on Learning Representations(ICLR)*.

[42] Guorui Zhou, Na Mou, Ying Fan, Qi Pi, Weijie Bian, Chang Zhou, Xiaoqiang Zhu, and Kun Gai. 2019. Deep interest evolution network for click-through rate prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence(AAAI)*, Vol. 33. 5941–5948.

[43] Guorui Zhou, Chengru Song, Xiaoqiang Zhu, Xiao Ma, Yanghui Yan, Xingya Dai, Han Zhu, Junqi Jin, Han Li, and Kun Gai. 2018. Deep Interest Network for Click-Through Rate Prediction. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining(KDD)* (2018).