A-HSG: Neural Attentive Service Recommendation based on High-order Social Graph

Chunyu Wei
Beijing National Research Center for Information Science and Technology (BNRist)
Department of Automation
Tsinghua University
Beijing 100084, China
cy-wei19@mails.tsinghua.edu.cn

Yushun Fan*
Beijing National Research Center for Information Science and Technology (BNRist)
Department of Automation
Tsinghua University
Beijing 100084, China
fanyus@tsinghua.edu.cn

Jia Zhang
Department of Electrical and Computer Engineering
Carnegie Mellon University
Silicon Valley
Moffett Field
CA 94035, USA
jia.zhang@sv.cmu.edu

Haozhe Lin
Beijing National Research Center for Information Science and Technology (BNRist)
Department of Automation
Tsinghua University
Beijing 100084, China
linhz16@mails.tsinghua.edu.cn

***Abstract*—With the widespread application of Service-Oriented Architecture, the quantity of web services keeps increasing rapidly over the Internet. Providing personalized service recommendation to users remains to be an important research topic. Recent studies have proved social connections helpful for modeling users' potential preference thus improving the performance of service recommendation. To date, however, one special type of social relation, called *high-order social relation*, has not been thoroughly studied. In reality, a user's preference may not only be affected by the user's direct neighbors, but also indirect ones. Furthermore, such influences may not remain static in the context of various attentions. To tackle such issues, we have developed a novel neural Attentive network based on High-order Social Graph (A-HSG) toward offering social-aware service recommendation. First, a graph convolution-based, multi-hop propagation module is devised to extract the high-order similarity signals from users' local social networks, and inject them into the users' general representations. Second, a neighbor-level attention module is constructed to adaptively select informative neighbors to model the users' specific preference. Extensive experiments over a real-life service dataset show that A-HSG outperforms baseline methods in terms of prediction accuracy.**

***Keywords*-Service Recommendation; Social Network; Attention; Graph Convolution; High-order Connectivity**

I. Introduction

With the rapid advancement of the Service Oriented Computing (SOC) and Cloud Computing techniques [1], numerous services have been published onto the Internet. Such services offer users a broad range of choices for fulfilling their demands without building everything from scratch. In this context, service recommendation technology is widely recognized as an important instrument to help users select suitable web services among vast amounts of candidates. In recent years, due to the prosperity of social media, increasingly more service-oriented systems have seamlessly integrated social features, such as Epinions, Yelp, Steam and Amazon. Meanwhile, traditional service repositories, such as ProgrammableWeb, also allow users to join communities and establish friendships with other users [2]. On these service platforms, users usually share their preference of services with their friends. For example, on the largest game service platform Steam, users can post comments about the game services on their profile pages or share game experiences with their friends through instant messages, which can potentially affect the game service selection of their friends. Thus, to accurately recommend web services to users, not only their historical preference should be considered, the social connections between the users and their friends also demand concerns.

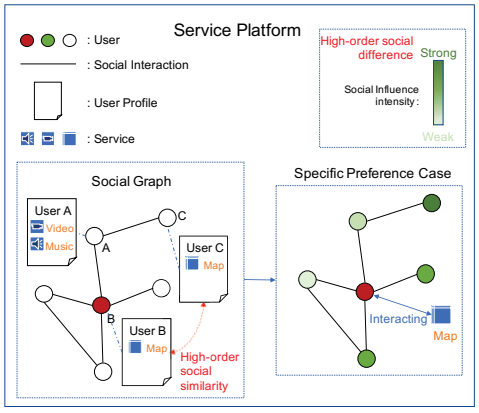

Figure 1. **An illustration of users' high-order social relation**. The left part shows the high-order similarity among users in service system, in which the profiles of user B and C indicate similar general preference between them. The right part describes the high-order difference existing in the service system. When interacting with Map service, the user's specific preference is getting influence of different intensities from its neighbors.

However, integrating social information into service recommendation is not a trivial task, especially when it comes to high-order influence, because users' preference may not

* Corresponding Author

978-1-7281-8786-0/20/$31.00 ©2020 IEEE
DOI 10.1109/ICWS49710.2020.00051

only be influenced by their own friends, but also result from their friends' social network. According to the social correlation theory [3], we adopt the term **high-order social relation** to describe such a common phenomenon in most service systems, which embodies two concepts. Figure 1 shows an example on a service platform to illustrate the definitions of these two concepts and how they may influence users' preference:

- **High-order social similarity (general preference).** High-order social similarity refers to a common case when a user and the friends of his friends tend to share similar general preference. For example, in the left part of Figure 1, User A loves to stay indoor playing games and watching tourism videos, which makes him become a social friend of two vloggers B and C on a service platform. So intuitively users B and C may share many common characters, which leads to the similar general preference for them. For example, both users B and C will use map service when travelling. However, such relationship cannot be reflected by only considering the first-order social relation. Therefore, when learning a user's representation, considering the similarity between the user and his high-order neighbors (like the similarity between B and C) may lead to significant gain in the accuracy of the user's representation.
- **High-order social difference (specific preference).** High-order social difference reflects users' specific preference influenced diversely by the users in their social networks. In other words, for a specific requirement, different users in the social network may contribute different influence intensities. The right part in Figure 1 shows an example when choosing a map service, a user tends to follow those neighbors who love traveling. This means that when modeling a user's specific preference on a service, it is necessary to treat the influence of the user's neighbors differently, which helps to distill significant preference signal towards the specific service among all the user's neighbors.

Considering these two unique features, we believe the social connections among users are beneficial to improve the quality of service recommendation.

To the best of our knowledge, there is not yet a recommendation approach considering the high-order social relation in web service domain. However, we notice that in some advanced machine learning technology, the social relation has been partially utilized to grain strides. For example, SBPR [4] and SocialMF [5] utilize users' social connections to improve the recommendation accuracy. But they do not consider both the similarity and the difference of the high-order social relation, which might easily lead to incompleteness of the representation but still provide good bases. Some methods like Deepinf [6] explicitly encode the high-order social similarity but with the social influence set equal or relied on predefined static functions. Some other methods like SAMN [7] only model the social difference of direct social friends, failing to exploit the high-order neighbors and integrate their features. Therefore, the social relation in service ecosystem still cannot be fully exploited.

In this work, we propose a neural Attentive network based on High-order Social Graph (A-HSG), which utilizes the recent advances in graph convolution [8][9] and neural attention mechanism [7] to simultaneously model both the general preference and the specific preference. Applying the idea of Graph Neural Network (GNN), we first devise an **social embedding propagation layer**, which refines the user's representation by aggregating the embeddings of its connected friends. By stacking multiple embedding propagation layers, the model is able to extract the high-order similarity signals from the social network and encode them into the user's general preference representation. Afterwards, to obtain the user's specific preference towards some service, a **neighbor-level attention module** places higher weights on the representations of the neighbors sharing similar preference on the same service, and then aggregates their weighted sum with the user's representation to get the final representation. Finally, a linear interaction between the user's final representation and the service embedding derives a ranking score. Over the real-life Steam game service dataset, our extensive experimental results show that our A-HSG model consistently outperforms the state-of-the-art methods in terms of prediction accuracy.

Our main contributions are three-fold:

1) We propose A-HSG, a novel service recommendation framework, which highlights the noteworthy significance of the high-order social relation in most service systems carrying both **high-order social similarity** and **high-order social difference**.
2) We employ a multi-hop propagation module to explicitly integrate the high-order social similarity into a user's representation, which enhances the capability of encoding his **general preference**.
3) We develop a neighbor-level attention module to adaptively measure the dynamic social influence strength in the context of different services, which enriches a user's representation with the **specific preference**.

The remainder of this paper is organized as follows. Sections II, III, and IV introduces preliminaries, describes our model framework and discusses experiments respectively. Section V compares with related work, and Section VI draws conclusions.

II. Preliminaries

A. Notation Definitions

In a service system, we use $\mathbb{U} = \{u_1, u_2, \ldots, u_M\}$ and $\mathbb{S} = \{s_1, s_2, \ldots, s_N\}$ to denote the set of **users** and the set of **services**, respectively. Let $\mathbb{G} = (\mathbb{U}, \mathbb{E})$ represent a static

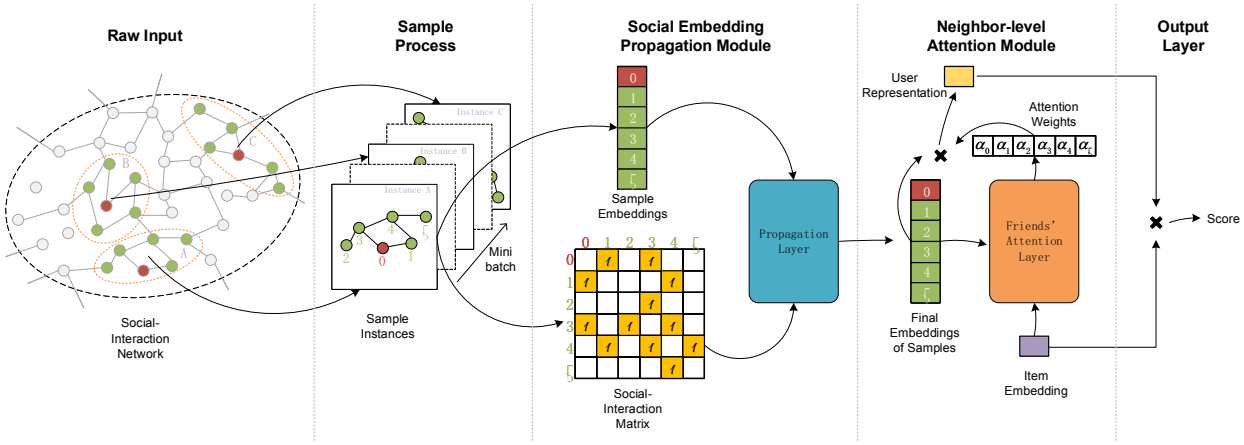

Figure 2. **Model Framework.** Prior to the model framework, our model firstly samples important neighbors from the social network $\mathbb{G}$ to form a sub-network for every user in the batch. Following the sampling, the data process framework comprises two main components: (1) a multi-hop propagation module to enforce the sampled users' embeddings to capture the high-order social signal; and (2) a neighbor-level attention module to assign non-uniform weights to the user and his sampled neighbors.

social network, where $\mathbb{E}$ is the edge set in a graph that denotes the social interactions between the users, such as friends in an undirected social network and followers in a directed social network. Note that we do not focus on the differences between these two types of networks in this project. For each user u_i, its **neighbors** are denoted as $u_{i1}, u_{i2}, ...$, with whom user u_i is directly or indirectly connected. We also define $\mathbf{F}(i)$ for each user u_i, which denotes the set of users with whom u_i is directly connected. According to the graph theory, $\mathbf{D}(i)$ denotes the "degree" of user u_i in a social graph, which is calculated by $\mathbf{D}(i) = |\mathbf{F}(i)|$ representing the number of users in the set $\mathbf{F}(i)$. To simplify the notation, we use the above notations to denote the same terms in the local social interaction graphs in the remainder of this paper.

B. Construction of local sub-network

For a specific user u_i, to represent his social features, the best way is to extract his sub-network from the social network $\mathbb{G}$. So we preform Breadth-First-Search (BFS) starting from the user u_i to sample a fixed number of neighbors $\mathbf{N}(i) = \{u_{i1}, u_{i2}, ...u_{iL}\}$ with a fixed length L, thus ensuring that the most influential neighbors are included in the sample list. For the ease of notation, we denote the user u_i itself as u_{i0} and also put it in the sample list. Thus the final sampled users in the local sub-network become $\mathbf{N}(i) = \{u_{i0}, u_{i1}, u_{i2}, ...u_{iL}\}$.

III. Model Framework

In this section, we will first introduce our A-HSG framework, then discuss its optimization. As illustrated in Figure 2, the architecture of A-HSG contains two consecutive components: a social embedding propagation module, and a neighbor-level attention module. The input of the A-HSG is a social interaction network, divided into a batch of local sub-networks through the sample process. The social embedding propagation module learns embedding matrices for users, and stacks multiple propagation layers to learn high-order social similarity signals and merge them into the user's general preference. The attention module learns high-order social difference signals and merge them into the user's specific preference. The final output of A-HSG is the learned user preference representation.

A. Social Embedding Propagation Module

Following the recent emergence of representation learning technique, we encode the user u_i and the target service s_j into a low-dimension latent space with embedding vectors $\mathbf{e}_{u_i} \in \mathbb{R}^{d_0}$ and $\mathbf{e}_s \in \mathbb{R}^{d_0}$, where d_0 is the initial embedding size. The embedding layer learns an embedding matrix $\mathbf{E} \in \mathbb{R}^{(L+1)\times d_0}$ as a look-up table, with each row corresponding to the representation of a user in the sample list:

$$\mathbf{E} = [\mathbf{e}_{u_{i0}}, \mathbf{e}_{u_{i1}}, \mathbf{e}_{u_{i2}}, ...\mathbf{e}_{u_{iL}}], \tag{1}$$

where $\mathbf{e}_{u_{ik}}$ represents the embedding of the user $u_{ik} \in \mathbf{N}(i)$.

According to the social correlation theory [3], users' general preference is influenced by their high-order neighbors. We thus build a multi-hop propagation module to model the social information flow. Figure 3 shows a two-hop propagation process. Here we will first show the information propagation of one single user in a sub-network. Then we extend the propagation to a matrix-form, which can operate layer-wise propagation over the entire sub-network. Finally, we generalize the layer-wise propagation to multiple stacked layers.

*1) **Propagation Process of a single user**:* For a user u_i and one of his connected friends u_k, we define the social

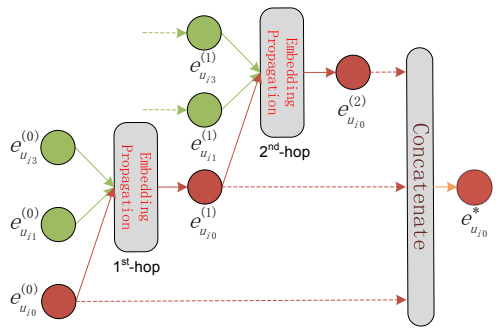

Figure 3. **Illustration of second-order embedding propagation for user** u_{i0}. Each node in the figure denotes the corresponding embedding of each user. After the 1st-hop propagation layer, u_{i0} and its neighbors integrate the social similarity of their 1st-order neighbors into their embedding. Then after the 2nd-hop propagation layer, u_{i0} again integrates its neighbors' new embeddings and thus indirectly encodes the high-order social similarity signals. By doing so, the 1st and the 2nd order social information diffuses to u_{i0} during the corresponding embedding propagation operation. We concatenate all the embeddings of each hop to get the final representation.

information flow from u_k to u_i as:

$$\mathbf{m}_{u_i \leftarrow u_k} = f(\mathbf{e}_{u_k}, \mathbf{e}_{u_i}, p_{ik}), \tag{2}$$

where p_{uk} is the decay factor of the propagation. Intuitively, the more connected friends $\mathbf{e}_{u_i}$ or $\mathbf{e}_{u_k}$ has, the smaller p_{uk} should be, since they will have less influence on their every single friend. Applying the concept of neural graph collaborative filtering [9], we set $f(\cdot)$ as:

$$\begin{aligned}\mathbf{m}_{u_i \leftarrow u_k} =& \frac{1}{\sqrt{|\mathbf{D}(i)||\mathbf{D}(k)|}}(\mathbf{W}_{self}\mathbf{e}_{u_k} + \\ & \mathbf{W}_{inter}(\mathbf{e}_{u_i} \odot \mathbf{e}_{u_k})),\end{aligned} \tag{3}$$

where $\frac{1}{\sqrt{|\mathbf{D}(i)||\mathbf{D}(k)|}}$ is the coefficient p_{uk}, $\mathbf{W}_{self}, \mathbf{W}_{inter} \in \mathbb{R}^{d_0 \times d_1}$ are independent weight matrices of this propagation process, and d_1 is the transformation size. We use $\mathbf{e}_{u_i} \odot \mathbf{e}_{u_k}$ to represent the interaction between both u_i and u_k, where $\odot$ denotes the element-wise product. This operation ensures that the connected friends with higher similarity to user u_i will pass more information to u_i.

By exploiting the social information flows from user u_i's all connected friends, we can update the representation of u_i after one round of propagation process as:

$$\mathbf{e}_{u_i}^{(1)} = ReLU(\mathbf{m}_{u_i \leftarrow u_i} + \sum_{u_k \in \mathbf{F}'(i)} \mathbf{m}_{u_i \leftarrow u_k}), \tag{4}$$

where $ReLU$ is a nonlinear activation function. In order to keep the original information of u_i, we insert the term $\mathbf{m}_{u_i \leftarrow u_i}$ in the function, which can be represented as:

$$\mathbf{m}_{u_i \leftarrow u_i} = \mathbf{W}_{self}\mathbf{e}_{u_i}. \tag{5}$$

Here $\mathbf{W}_{self} \in \mathbb{R}^{d_0 \times d_1}$ shares the same values as the weight matrix in Equation(3).

*2) **Matrix form of the Propagation Process**:* To carry out the propagation process for all users in the network, we transform Equation (3), (4), and (5) into a matrix form as:

$$\mathbf{E}^{(1)} = ReLU((\mathcal{L} + \mathbf{I})\mathbf{E}\mathbf{W}_{self} + \mathcal{L}\mathbf{E} \odot \mathbf{E}\mathbf{W}_{inter}) \tag{6}$$

where $\mathbf{E}^{(1)}$ is the propagation result of the initial embedding matrix $\mathbf{E}$, and $\mathcal{L}$ is the Laplacian matrix of the sub-graph, which can be calculated as follows:

$$\mathcal{L} = \mathbf{D}^{-\frac{1}{2}}\mathbf{R}(i)\mathbf{D}^{-\frac{1}{2}}, \tag{7}$$

where $\mathbf{D}$ is the diagonal degree matrix with each of its diagonal elements $\mathbf{D}_{ii} = \mathbf{D}(i)$. The diagonal element $\mathcal{L}_{ii} = 0$ and the off-diagonal element $\mathcal{L}_{ik} = \frac{1}{\sqrt{|\mathbf{D}(i)||\mathbf{D}(k)|}}$.

By implementing the matrix form, we can not only update all the user representations in the same sub-network simultaneously, but also facilitate the batch calculation. Furthermore, it enables us to easily stack multiple propagation processes to extract the high-order social signals, which will be discussed in the next part.

*3) **Multi-hop Propagation**:* By stacking more propagation layers, the module can explore the high-order social similarity information. In the l-th step of propagation, Equation (6) can be reformulated as:

$$\begin{aligned}\mathbf{E}^{(l)} =& ReLU((\mathcal{L} + \mathbf{I})\mathbf{E}^{(l-1)}\mathbf{W}_{self}^{(l)} + \\ & \mathcal{L}\mathbf{E}^{(l-1)} \odot \mathbf{E}^{(l-1)}\mathbf{W}_{inter}^{(l)}),\end{aligned} \tag{8}$$

where $\mathbf{E}^{(l)} \in \mathbb{R}^{(L+1) \times d_l}$ are the representations of the users in $\mathbf{N}(i)$ after l times of propagation steps, and $\mathbf{E}^{(l-1)}$ is the representation from previous steps. $\mathbf{W}_{self}^{(l)}, \mathbf{W}_{inter}^{(l)} \in \mathbb{R}^{d_{l-1} \times d_l}$ are the transformation matrices for the l-th step. We assign different weight matrices for the transformation in each step, and thus the dimension of $\mathbf{E}^{(l)}$ is changed after every propagation. $\mathbf{E}^{(0)}$ is set as $\mathbf{E}$ in Equation (1).

After propagating l times, we receive $l+1$ representations for each user in $\mathbf{N}(i)$. According to above analysis, those representations focus on diverse social information flows and manifold preference. Thus to construct a comprehensive embedding, we add a hidden layer on the concatenated embedding matrix:

$$\begin{aligned}\mathbf{E}^* &= Concat(\mathbf{E}, \mathbf{E}^1, \mathbf{E}^2, ..., \mathbf{E}^l)\mathbf{W}_r \\ &= [\mathbf{e}_{u_{i0}}^*, \mathbf{e}_{u_{i1}}^*, \mathbf{e}_{u_{i2}}^*, ...\mathbf{e}_{u_{iL}}^*],\end{aligned} \tag{9}$$

where $\mathbf{W}_r \in \mathbb{R}^{(d_0+d_1+...+d_l) \times d_0}$ is the transformation matrix regulating the final size, and $\mathbf{e}_{u_{ik}}^*$ denotes the final embedding of user u_{ik} in $\mathbf{N}(i)$ and it is also one row of the matrix $\mathbf{E}^*$.

B. Neighbor-level Attention Mechanism

The main purpose of neighbor-level attention is to assign different weights to a user's sample neighbors when the user

is interacting with a specific service and thus help model the specific preference. First, we use the element-wise product of the final user embedding and the service embedding to represent the "opinion" that the user has on the service:

$$\mathbf{o}_i = \mathbf{e}_i^* \odot \mathbf{q}, \tag{10}$$

where $\mathbf{e}_i^*$ is the final representation of user i. Given the opinion of the user and his neighbors, we implement a concatenate operation to learn the joint opinion of the user and each of his neighbors. Then a two-layer network is applied to compute the attention weights $\alpha_{(j)}$ $(1 \leq j \leq L)$:

$$\alpha_{(j)}^* = \mathbf{h}^T ReLU(\mathbf{W} Concat(\mathbf{o}_0, \mathbf{o}_j) + \mathbf{b}), \tag{11}$$

where $\mathbf{o}_j$ is the opinion of the j-th user in the sample set (the 0-th one is the target user). Note that $\mathbf{W} \in \mathbb{R}^{2d_0 \times d_0}$, $\mathbf{h}^T$ are model parameters.

Afterwards, we normalize the neighbor-level attention scores with a softmax function, which can make the attention network a probabilistic interpretation:

$$\alpha_{(j)} = \frac{\exp(\alpha_{(j)}^*)}{\sum_{1 \leq i \leq L} \exp(\alpha_{(i)}^*)}. \tag{12}$$

Then the final representation of the user can be formulated as:

$$\mathbf{U}_i = \mathbf{e}_{u_{i0}}^* + \sum_{1 \leq j \leq L} \alpha_{(j)} \mathbf{e}_{u_{ij}}^*, \tag{13}$$

where $\mathbf{U}_i$ is final representation of user u_i.

C. Learning

1) Prediction: After obtaining the representation of user u_i, we then apply the Matrix Factorization (MF) technique to model the implicit feedback and rating prediction:

$$\hat{\mathbf{Y}}_{ij} = \mathbf{U}_i^T \mathbf{q}_j \tag{14}$$

where $\hat{\mathbf{Y}}_{ij}$ is the predicted score for the unused services, which will then be ranked in descending order to provide Top-K service recommendation list.

2) Optimization: The core of the optimization process is to learn the relevance-based ranking of services. Existing researches usually apply Bayesian Personalized Ranking (BPR), which compares the score of the positive service with several sampled services. However, in this case, we found that BPR is unstable in optimization. Hidasi et al. proposed **Top1** [10], a regularized approximation of the relative rank of the relevant item. For each positive user-service pair $< u_i, s_j >$, we randomly sample a negative service from the unobserved services of the user, which is denoted as s_k. The **Top1** pairwise ranking loss is as follows:

$$L_{TOP1} = \sum_{(i,j,k) \in \mathfrak{D}} \sigma(\hat{\mathbf{Y}}_{ik} - \hat{\mathbf{Y}}_{ij}) + \sigma(\hat{\mathbf{Y}}_{ik}^2) \tag{15}$$

where $\sigma(x) = \frac{1}{1+\exp(-x)}$ is the logistic sigmoid function and $\mathfrak{D}$ represents the set of pairwise training instances. The second term of equation (15) is a regularization term.

IV. Experiments

A. Experimental Settings

*1) **Dataset Description**:* Steam is a platform providing game and software service, which includes services from over 1,200 publishers and over 75 million active Steam users. In addition to its service store, Steam provides numerous social networking features such as profile pages, friends, groups, instant messaging, and voice chats. According to [1] and [11], the game service shows a high level of similarity to web services with three unique features: (1) The friendship connections are sparse compared to other social networks; (2) There are long tail behaviors in their distributions; and (3) Services on heterogeneous platforms are massive and varied.

To test and verify our A-HSG, we randomly crawled 7,043 Steam accounts, along with the friendship lists, and a total of 16,112 game services to construct a real-world dataset, where the social network is built according to Section II.B. Table I summarizes the numerical properties.

Table I
Statistics of the datasets

Item	Number
User	7,043
Service	16,112
Invocations	6,700
Density(Invocations)	0.0059%
Social Interaction	8,509
Density(Social Interaction)	0.0172%

*2) **Evaluation Metrics**:* To evaluate the performance of all algorithms, we adopted two popular metrics, namely Normalized Discounted Cumulative Gain (NDCG) and Hit Ratio (HR). The NDCG@K metric is position-aware, which accounts for the position of the hits by assigning higher scores to hits at top ranks. The HR@K metric measures whether the test item is present on the recommendation list. Both the adopted metrics can be formulated as follows:

$$NDCG@K = \frac{1}{R_N} \sum_{i=1}^{N} \frac{2^{rel_i - 1}}{\log_2(1+i)} \tag{16}$$

$$HR@K = \frac{\sum_{i=1}^{K} rel_i}{|y_u^{test}|} \tag{17}$$

Where K is the size of the recommendation list, $rel_i = 0$ or 1 denotes whether the service at the rank i is in the test set, and the R_N term indicates the maximum possible cumulative component through ideal ranking. $|y_u^{test}|$ is the number of service used by the user u in the test set.

*3) **Baselines**:* To evaluate the performance of the Top-K recommendation, we compared our A-HSG with the following five baseline methods:

- **BPR** [12]: This method optimizes the matrix factorization (MF) model with the BPR objective function for implicit feedback-based service recommendation.
- **SBPR** [4]: This is a ranking model assuming that users tend to assign higher ranks to items their friends prefer.
- **NCF** [13]: This method is the state-of-the-art deep learning based framework combining matrix factorization (MF) with a multilayer perceptron model.
- **SocialMF** [5]: This is a classical model considering the trust information and its propagation into the matrix factorization model for recommender systems.
- **SAMN** [7]: This is a state-of-the-art deep learning method, which unifies the strengths of memory networks and attention mechanisms to address the problems in social-aware recommendations.

Our method aims to model the social relationship in the service recommendation. However, to the best of our knowledge, few research has explored the effect of the social influence in service recommendation. Thus we adjusted many general social recommendation methods to compare with our A-HSG on the Steam game service dataset.

Table II
OVERALL PERFORMANCE COMPARISON

	HR@10	NDCG@10	NDCG@20
SocialMF	0.4360±0.0059	0.4138±0.0111	0.5575±0.0040
BPR	0.4706±0.0051	0.4360±0.0082	0.6262±0.0027
SBPR	0.4902±0.0070	0.4425±0.0085	0.6538±0.0038
NCF	0.5294±0.0016	0.4753±0.0053	0.6463±0
SAMN	0.5882±0.0086	0.5567±0.0100	0.6617±0.0027
A-HSG	**0.6471±0.0066**	**0.6136±0.0086**	**0.7070±0.0030**

*4) **Experiment Details**:* We implemented A-HSG on the basis of Pytorch [14], a widely used Python library for neural networks. We randomly split the dataset into training set (70%), validation set(20%) and test set(10%). During the tuning process, we found that 0.005 can be a good initial learning rate with an embedding size of 64 respectively. Borrowing the idea of autoencoder, the transformation size sequence should be non-increasing. By shrinking the transformation size, each propagation process can learn more abstractive features. Empirically, we halved the transformation size for each successive propagation layer.

B. Comparative Analysis on Overall Performance

The empirical results of our A-HSG and the baselines on the Steam game service dataset are summarized in Table II. We conducted one-sample t-tests and $p - value < 0.05$ indicates that the improvements of A-HSG over the strongest baseline are statistically significant. From the results, we drew the following conclusions:

- The methods integrating social information perform better than the ones without it. For example, in Table II for most metrics, SBPR shows a better performance than BPR, while the state-of-the-art SAMN and our A-HSG outperform BPR and NCF. The experiment results provide trustworthy evidence for introducing the social information into the service recommendation.
- We noticed that the methods assigning different weights to different neighbors bear better performances than those which do not. Compared to SBPR and SocialMF, the performance of SAMN and A-HSG proves that attention mechanisms on the friend level improve the user representation learning. It might because the influence strength of a user's friends should be different.
- A-HSG consistently yields the best performance on the Steam dataset, which improves over the strongest baseline SAMN with respect to NDCG@10 by 9.54%. In our model, we stack multiple propagation layers to extract the high-order social similarity in an explicit way, while the state-of-the-art SAMN only considers the directed friends. This result strongly supports that the high-order social similarity can improve the user representation learning in the service ecosystem.

C. Effect of Hop time and Transformation Size

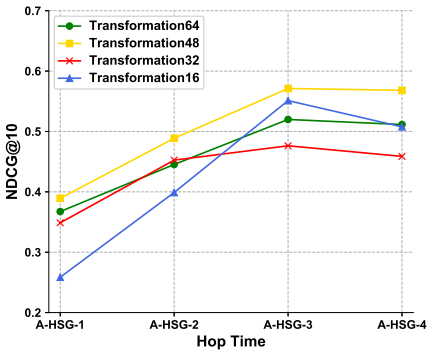

Figure 4. **Performance comparison on models of different hop times w.r.t. different transformation sizes of 1st propagation layer**

Since the multi-hop propagation module plays a key role in A-HSG, we varied the model depth and the size of each layer to investigate how A-HSG benefits from multiple propagation layers. HR and NDCG show similar patterns, thus we focus our analysis on NDCG. The experiment results are shown in Figure 4, wherein the term A-HSG-# indicates the A-HSG method with # propagation layers. Since we set the transformation size of each layer to reduce by half, in Figure 4, we only display the transformation size of the first propagation layer. From Figure 4 and Table II, we have the following observations:

- The general trend shows that the increase of the hop time can improve the service recommendation performance. From Figure 4, we notice that under

different transformation sizes, A-HSG-2 and A-HSG-3 consistently show better performance than A-HSG-1. Combining above-mentioned analysis in section IV-B, we attribute this improvement to the increased level of the ability to extract the high-order social similarity.

- When we added more propagation layers on A-HSG-3, the NDCG@10 of the model started to decrease, which might be caused in two aspects. First, too deep architecture introduce noises into the learning process. Second, the neighbors far away from the user have little influence on the user. Thus the marginal improvement brought by the far neighbors is insufficient to offset the noises. When further stacking a propagation layer on A-HSG-4, the model becomes extremely unstable due to the gradient explosion. Stated thus, in most service systems, three hops provide sufficient capacity to model the complex general preference of users.
- Figure 4 portrays a steady improvement as the 1st-transformation size increases, where the 1st-transformation size of around 48 shows peak performance followed by a degradation due to overfitting. At 1st-transformation size of 16 shows an unusual rise in performance, which might be the contribution of the layer-aggregation mechanism. Although each size less than 16 seems unable to carry much information, when concatenating all layers' outputs together, the final representation has a strong ability to encode the social similarity of each order and thus improves the general preference modeling.

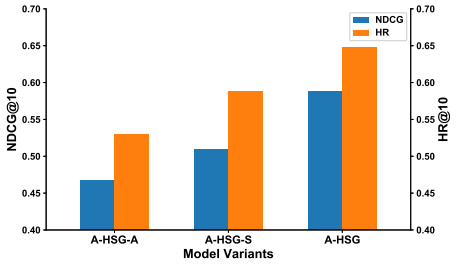

Figure 5. **Performance of variants of A-HSG**

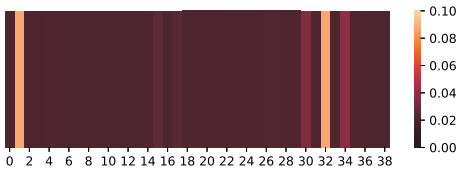

Figure 6. **Case study.** Attention distributions of a user's neighbors in the final representation towards specific service

D. *Effect of the Neighbor-level Attention*

We further evaluated the key component of the model - **Neighbor-level Attention**, by conducting experiments with the following variants of A-HSG:

- **A-HSG-S** A variant model of A-HSG without aggregating the neighbors' representations with the final representation of the user.
- **A-HSG-A** A variant model of A-HSG in which the weights of the neighbors in Equation 13 are set equal.

Figure 5 shows the performances of A-HSG and its two variants. Due to space limitation, we show only the results of the NDCG@10 and HR@10 on our Steam dataset. From Figure 5, two observations are made:

1) When the neighbor-level attention mechanism is applied, the performances achieve significant improvement compared to A-HSG-S and the constant weight method A-HSG-A. This may because the user representation from the propagation module only explicitly encodes the user's general preference without considering the user's specific preference towards the target service. The result also shows that the attention mechanism can learn an adaptive weight to make sure that when interacting with different services, the neighbors who share common preference can be more prominent in the user's final representation.
2) A-HSG-A performs the worst among the variant models since it does not consider the high-order social difference. Even worse, by setting all weights equal and then aggregating the weighted sum with the user's representation from the propagation module, it disturbs the representation of the user's general preference.

To better understand how attention mechanisms help treat the influence intensities of neighbors in a user's social network, in Figure 6, we randomly select 39 neighbors of a user (#6565) to see their contribution in his final representation, when interacting with random service #52. We have the following observations: (1) The attention weights of a user's neighbors are different. For example, the attention weights of neighbors No.1 and No.32 are relatively high, which may because they have both consumed service #52. (2) The neighbors with more invocation histories tend to have high attention weights. For example, neighbors No.30 and No.34 both consume over 10 game services which makes their attention weights also a bit higher then others.

V. Related Work

Our study is closely related to a large and growing body of literature on service recommendation. With the explosive development of service ecosystem, service recommendation is one key issue in the field of service oriented computing due to the information overload problem.

Semantic-aware methods mainly focus on information retrieval and similarity calculation. Existing methods usually

extract semantic information, e.g., keywords and labels, and calculate relevance scores represented by semantic distance. Li et al. represented users and services as vectors of words and calculated the cosine similarity between corresponding vectors [15]. Based on latent dirichlet allocation (LDA), [16] revealed a correlation between services and words extracted from related WSDL documents.

Quality-of-Service (QoS) is widely employed to represent the nonfunctional performance of web services and has been adopted as a key factor in service selection. Zheng et al. [17] proposed a user-collaborative mechanism for collecting historical QoS data of Web services from different service users. Ahmed et al. [18] proposed a model predicting web service's behaviors by predicting the status of underlying hidden states in terms of response time.

Network-aware methods exploit network analysis to make service recommendation. Huang et al. [1] provided both a methodology to quantify a service-mashup ecosystem and an empirical study on ProgrammableWeb, a service repository.

However, the aforementioned works on service recommendation assume that all the service users are independent. They mainly focus on modeling the feedback order by using service users' positive and negative feedback, but do not investigate how the feedback from users' friends can be used to model users' preference on services.

With the prevalence of social media, social influence is everywhere around us, not only in our daily physical life but also on the virtual Web space. Some researchers proposed to enhance the reliability of service recommendation by integrating users' social connections. Tang et al. [2] adapted the conventional CF technique by choosing recommending users for the target user in regard to both similarity and trust between them. Kalai et al. [19] proposed a web service discovery process by taking into account the best social friendships of the current user and the past invocation histories with satisfactory web services of one's friends. Previous published studies, though inspiring, are limited to users' direct friends without considering the **high-order connectivity** from user-user interaction. Moreover, the social influence strength in most of the works is usually set equally for the social connections or relied on a predefined static function, which should be different [7].

VI. Conclusions

Social information has shown a great potential to improve the performance of service recommender systems. In this paper, we have presented a novel model which unifies graph convolutional techniques and attention mechanisms, to seamlessly integrate the high-order social similarity and difference into user representations for more accurate service recommendation. Our main ideas includes: 1) applying graph convolution to model the multi-hop propagation process of the social information; 2) developing a neighbor-level attention instrument to assign dynamic weights to user's neighbors in the context of various services. Extensive experiments have proved that A-HSG outperforms the baseline methods in prediction accuracy on the real-world game service dataset.

In our future work, we plan to extend A-HSG to incorporate the content and context information of service users to deal with the service-side cold-start problem.

Acknowledgment

This research has been partially supported by the National Key Research and Development Program of China (No.2018YFB1402500), the National Natural Science Foundation of China (No.61673230) and High-Tech Ship Research Project of China (17GC26102.01).

References

[1] K. Huang, Y. Fan, and W. Tan, "An empirical study of programmable web: A network analysis on a service-mashup system," in *2012 IEEE 19th International Conference on Web Services*. IEEE, 2012, pp. 552–559.

[2] M. Tang, Y. Xu, J. Liu, Z. Zheng, and X. F. Liu, "Trust-aware service recommendation via exploiting social networks," in *2013 IEEE International Conference on Services Computing*. IEEE, 2013, pp. 376–383.

[3] P. V. Marsden and N. E. Friedkin, "Network studies of social influence," *Sociological Methods & Research*, vol. 22, no. 1, pp. 127–151, 1993.

[4] T. Zhao, J. McAuley, and I. King, "Leveraging social connections to improve personalized ranking for collaborative filtering," in *Proceedings of the 23rd ACM international conference on conference on information and knowledge management*, 2014, pp. 261–270.

[5] M. Jamali and M. Ester, "A matrix factorization technique with trust propagation for recommendation in social networks," in *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 2010, pp. 135–142.

[6] J. Qiu, J. Tang, H. Ma, Y. Dong, K. Wang, and J. Tang, "Deepinf: Modeling influence locality in large social networks," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'18)*, 2018.

[7] C. Chen, M. Zhang, Y. Liu, and S. Ma, "Social attentional memory network: Modeling aspect-and friend-level differences in recommendation," in *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. ACM, 2019, pp. 177–185.

[8] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 974–983.

[9] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," *arXiv preprint arXiv:1905.08108*, 2019.

[10] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," *arXiv preprint arXiv:1511.06939*, 2015.

[11] M. O'Neill, E. Vaziripour, J. Wu, and D. Zappala, "Condensing steam: Distilling the diversity of gamer behavior," in *Proceedings of the 2016 internet measurement conference*. ACM, 2016, pp. 81–95.

[12] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," in *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*. AUAI Press, 2009, pp. 452–461.

[13] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th international conference on world wide web*. International World Wide Web Conferences Steering Committee, 2017, pp. 173–182.

[14] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019, pp. 8024–8035.

[15] X. Dong, A. Halevy, J. Madhavan, E. Nemes, and J. Zhang, "Similarity search for web services," in *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment, 2004, pp. 372–383.

[16] C. Li, R. Zhang, J. Huai, X. Guo, and H. Sun, "A probabilistic approach for web service discovery," in *2013 IEEE International Conference on Services Computing*. IEEE, 2013, pp. 49–56.

[17] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "Qos-aware web service recommendation by collaborative filtering," *IEEE Transactions on services computing*, vol. 4, no. 2, pp. 140–152, 2010.

[18] W. Ahmed, Y. Wu, and W. Zheng, "Response time based optimal web service selection," *IEEE Transactions on Parallel and distributed systems*, vol. 26, no. 2, pp. 551–561, 2013.

[19] A. Kalaï, C. A. Zayani, and I. Amous, "User's social profile–based web services discovery," in *2015 IEEE 8th International Conference on Service-Oriented Computing and Applications (SOCA)*. IEEE, 2015, pp. 2–9.