Time-Aware Service Recommendation With Social-Powered Graph Hierarchical Attention Network

Chunyu Wei, Yushun Fan, and Jia Zhang, *Senior Member, IEEE*

Abstract—Driven by Service-Oriented Computing techniques, time-aware service recommendation aims to support personalized mashup development, adapting to the rapid shifts of users' dynamic preferences. Recent studies have revealed that users' social connections may help better model their dynamic preferences. However, two phenomena exist to influence users' dynamic preferences of service selection. First, users and their friends may only share preferences in certain services, which means not every service in the friends' consumed mashups has the same impact on a target user's dynamic preference. Second, for a target user, friends in his social network with similar interests and behaviors may contribute more influence intensities. To cover the above phenomena synergistically, this paper proposes a Social-powered Graph Hierarchical Attention Network (SGHAN), as a deep learning model capable of learning similar behaviors from proper friends during mashup development. SGHAN is powered by the reciprocity between its two core components: a service-level attentional encoder captures users' interested services in friends' mashups, while a friend-level graph attention network selects informative friends and propagates the friends' social influences. Extensive experiments show that the SGHAN model consistently outperforms the state-of-the-art methods in terms of prediction accuracy for mashup creation.

Index Terms—Service recommendation, graph neural networks, social network, time-aware, mashup creation

1 Introduction

With the extensive adoption of the Service-Oriented Computing (SOC) and Cloud Computing techniques, a huge number of software services have been published onto the Internet. Users benefit from reusing and integrating these reusable services from different providers as components, in order to dynamically create value-added mashups (i.e., a sequence of services working in proper order in a certain time period) to fulfill user demands.

User interests may change overtime [1]. For example, a user may be interested in Pop music services for some time and switch to Jazz music services afterwards. Thus providing fixed mashups for users cannot adapt to the rapid shifts of users' dynamic interests. In order to respond to user needs in a timely manner, *time-aware service recommendation* [2], [3], [4] focuses on a user's service invocation behaviors within a period of time and predict the next service for the user in order to form a dynamic mashup. Some recent studies establish time-aware service recommendation frameworks for mashup creation, which conduct a joint analysis of temporal information, content description, and historical mashup-service usage [2], [5].

Owing to the recent prosperity of social media, increasingly more service-oriented systems have synergistically integrated social features to help users discuss and choose services [6], such as Epinions, Gowalla, and ProgrammableWeb. Since the online communities on these platforms often promote sharing of service experiences among friends, users are likely to be influenced by their friends. As shown in previous studies on social-aware service recommendation [6], [7], the consideration of friends' service invocation behaviors can help generate interest-adaptive service mashups for target users.

How to effectively integrate social information to support time-aware service recommendation remains a challenge. Most existing studies focus on analyzing different impacts in social networks based on friendships or human popularity [8]. Some recent works study high-order social impacts using neural networks [6]. However, we have observed two interesting phenomena, which may influence target users' dynamic preferences of service selection and have not been fully studied.

- *Service-level Differences.* At a certain time, a user may be interested in searching certain type of services. This means that not every service in a friend's consumed mashup has the same impact on the target user's dynamic preference. As illustrated in Fig. 1, user A's four friends' (B, C, D, and E) mashup experiences may all have influences on his next service selection. However, it is obvious that A will be more interested in travel-related services, after invoking the flight ticket reservation and hotel booking services. Thus in his

- *Chunyu Wei and Yushun Fan are with the Beijing National Research Center for Information Science and Technology (BNRist), Department of Automation, Tsinghua University, Beijing 100190, China. E-mail: cy-wei19@mails.tsinghua.edu.cn, fanys@tsinghua.edu.cn.*
- *Jia Zhang is with the Department of Computer Science, Southern Methodist University, Dallas, TX 75205 USA. E-mail: jiazhang@smu.edu.*

Manuscript received 20 March 2022; revised 8 July 2022; accepted 2 August 2022. Date of publication 9 August 2022; date of current version 12 June 2023.
This work was supported by the National Natural Science Foundation of China under Grant 62173199.
(Corresponding author: Chunyu Wei.)
Digital Object Identifier no. 10.1109/TSC.2022.3197655

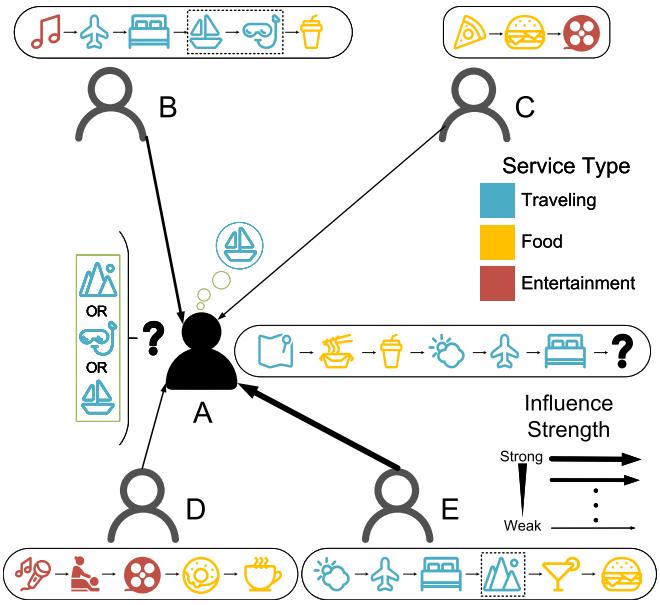


Fig. 1. *An illustration of how user A's dynamic preference is influenced by two levels of differences.* Five users consumed different types of services in their corresponding mashups, while User A may be influenced by his friends when choosing his next service in the mashup.

friends' mashups, the travel-related services will likely have more influence on user A's next service selection. This motivating example implies that when modeling a friend's influence on a target user, it is necessary to treat some relatively important services in the friend's mashup differently, which will better characterize the target user's main focus in his friend's mashup.

- *Friend-level Differences.* For a target user, different friends in his social network may contribute different influence intensities. As shown in Fig. 1, user A is more likely to follow user E when choosing the next service, since A's current mashup pattern is much more aligned with that of E. This motivating example implies the necessity to study friend-level differences, which can ensure the friends with similar interests and behaviors have significant influence over the target user.

Based on the above motivating examples, we argue that hierarchically modeling both service-level differences and friend-level differences in a user's social relationships and their mashup history may further improve time-aware service recommendation.

To realize the goal, this article proposes a Social-based Graph Hierarchical Attention Network (SGHAN s'kæn) to make time-aware service recommendation for dynamic mashup creation. Leveraging the recent advances in neural attention mechanisms [9], [10] and graph attention network [11], SGHAN explores a hybrid framework to model both the service-level and the friend-level differences existing in social influence hierarchically. Our prerequisites are that, a user's social friends are available in a service ecosystem, and their mashup history has been recorded.

SGHAN comprises two major modules to model the social influence: a service-level attentional encoder and a friend-level graph attention network. First, the target user's behavior within a mashup is modeled using a recurrent neural network (RNN) [12] to characterize his current interest. Based on the extracted representation, a *service-level attentional encoder* is employed to capture the user's main interested services in a friend's mashup by aligning them with the target user's current interest. This encoder learns to attend differentially to more and less important services when modeling the friends' social influence. Second, a *friend-level graph attention network* is created to automatically select informative friends for user preference modeling, and propagate the friends' social influence along the relationships to the target user. Finally, the two modules are fused to mutually enhance each other via an end-to-end training process. Over the real-world local service dataset, a series of experimental results demonstrate that our SGHAN model consistently excels baseline methods in terms of prediction accuracy for mashup creation.

In summary, our main contributions are three-fold:

1) We move one step forward on social-powered time-aware service recommendation, by investigating the effect and significance of both service-level differences and friend-level differences.
2) we propose SGHAN, a deep learning-powered time-aware service recommendation framework for dynamic mashup creation, which highlights two core modules: a *service-level attentional encoder* and a *friend-level graph attention network*. For a target user, the former module adaptively focalizes interested services in his friends' mashups, while the latter module identifies and learns from friends with similar interests and behaviors. The two modules jointly learn from mashup history and reciprocally enhance each other, in order to learn representations of users' dynamic preferences for service selection.
3) Through extensive experiments conducted on a real-life local service dataset, we show that SGHAN consistently outperforms the state-of-the-art models and justify the effectiveness of our proposed framework in capturing both the service-level and friend-level differences.

The remainder of this article is organized as follows. Section 2 formally defines the problem. Section 3 introduces our SGHAN model framework in details. Section 4 presents conducted experiments with analyses. Section 5 discusses related work. Finally, Section 6 draws conclusions.

2 Problem Definition

In this section, we formally define the problem of time-aware service recommendation, as well as some essential notations.

Definition 1. *(Service Ecosystem). In a service ecosystem,* $\mathbb{U} = \{u_1, u_2, \ldots, u_M\}$ *and* $\mathbb{S} = \{s_1, s_2, \ldots, s_N\}$ *denote its comprising set of* ***users*** *and the set of* ***services****, respectively.*

Definition 2. *(Social Graph). Let* $\mathbb{G} = (\mathbb{U}, \mathbb{E})$ *represent an undirected* social graph*, where* $\mathbb{E} \subseteq (\mathbb{U}, \mathbb{U})$ *is the edge set in the graph denoting the social relationships between the comprising users. Note that we do not differentiate between directed graphs and undirected graphs in this work.*

Traditional service recommender systems suggest relevant services to users based on the users' historical behaviors, in

TABLE 1
Notations and Explanations

Notation	Explanation
$\mathbb{U}$	user set
$\mathbb{S}$	service set
$\mathbb{G}$	social graph
$\mathbb{E}$	edge set of the social graph
u_i	user i
s_i	service i
$s^u_{t,p}$	the p-th service consumed by user u at his t-th mashup
M^u_t	user u's t-th mashup
$N(u)$	the set of user u's friends
p^u	target user u's current interest representation
g^k	friend k's preference influence to the target user
$q^{(l)}_u$	target user u's representation at l-th convolution layer

which the temporal order of their consumed services is ignored. However, in most service ecosystems, users' service preferences change rapidly, and thus the order of the users' recent involved services is of great importance for modeling the users' dynamic interests. In practice, users tend to consume several services in a given time window for a specific purpose and these consumer-centric services can be composed as a **Mashup**, which we define as follows:

Definition 3. *(Mashup). The behavior history of a user $u \in \mathbb{U}$ is segmented into a series of mashups chronologically $\{M^u_1, M^u_2, \ldots, M^u_t\}$. User u's t-th **mashup** M^u_t represents a sequence of services $\{s^u_{t,1}, s^u_{t,2}, \ldots, s^u_{t,n}\}$ invoked by user $u \in \mathbb{U}$ in chronological order, where $s^u_{t,p}$ represents the p-th service consumed by user u at his t-th **mashup**, and n is the amount of services contained in user u's t-th **mashup**.*

To avoid confusion, throughout the article, $1, 2, \ldots, t$ denote the order of user's consumed mashups, while $1, 2, \ldots, n$ denote the time steps of services in a mashup.

To provide accurate service recommendation according to a user's current purpose to form a proper mashup, we formally define a *Time-aware Service Recommendation* problem as follows:

Definition 4. *(Time-aware Service Recommendation). Given user u's ongoing mashup $M^u_{t+1} = \{s^u_{t+1,1}, s^u_{t+1,2}, \ldots, s^u_{t+1,n}\}$, the goal of time-aware service recommendation is to recommend a set of services from $\mathbb{S}$ that u is likely to be interested in during the next time step $n+1$, i.e., $s^u_{t+1,n+1}$, which can work in conjunction with the former services as a mashup.*

Recent trends in social media have motivated a number of services systems to synergistically integrate social features. Users on these services systems usually spread their preferences over services to their social connections. Thus, a user's interests are not only correlated to his historical usage records, but also can be further influenced by his social connections. Thus we model both the user's dynamic interests in a mashup and social influences with different intensities and formally define the resulting problem as follows:

Problem Formulation. Given a part of user u's recent mashup $M^u_{t+1} = \{s^u_{t+1,1}, s^u_{t+1,2}, \ldots, s^u_{t+1,n}\}$, our task is to recommend a set of services from $\mathbb{S}$ that u is likely to be interested in during the next time step $n+1$, i.e., $s^u_{t+1,n+1}$, by utilizing information from both his dynamic interests (i.e., information from M^u_{t+1}) and the social influences (i.e., information from $\cup_{k=1}^{N(u)} M^k_t$, where $N(u)$ is the set of user u's friends). Note that for simplicity, in this work we model the social influences as all friends' interests right before the target user starts to develop the current mashup M^u_{t+1}.

The notations used throughout the article are summarized in Table 1.

3 SGHAN Model Framework

In this section, we first outline the overall architecture of our SGHAN framework and give detailed descriptions of its main components, then analyze its learning process including the design of the loss function, followed by discussing the computation complexity of SGHAN.

Fig. 2 illustrates the overall architecture of the SGHAN framework comprising three core modules. First, a dynamic interest encoder applies a recurrent neural network (RNN) to model the sequence of services in the target user's current mashup, and extracts the user's current interests. Second, a service-level attentional influence encoder determines which services of the hidden representations in a friend's mashup should be emphasized or ignored based on the target user's

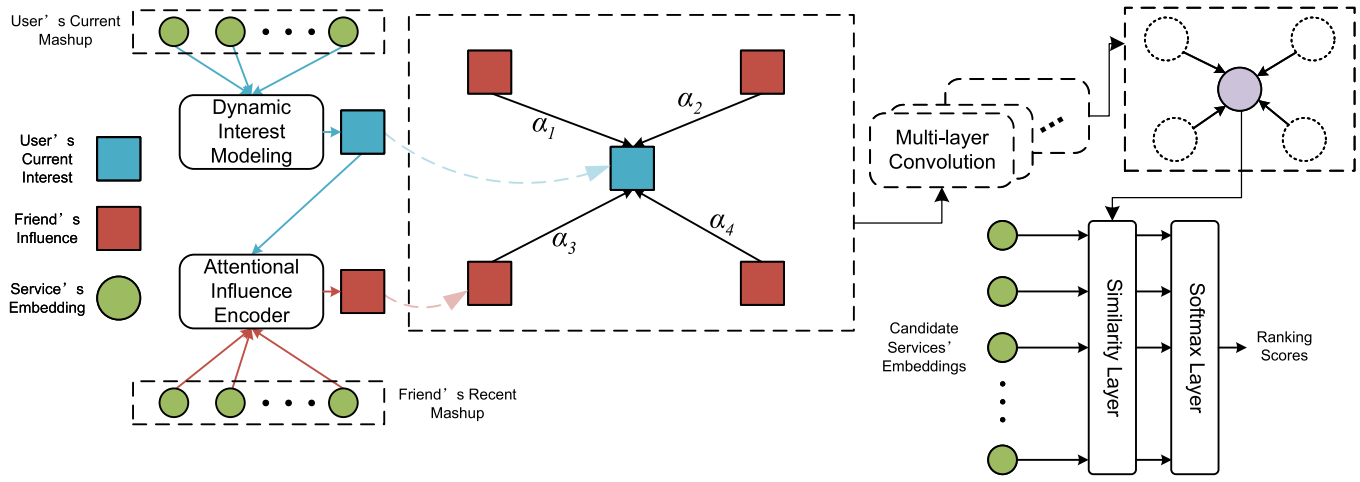


Fig. 2. *Model Framework.* SGHAN contains three components: (1) a dynamic interest encoder, which captures a user's fast-changing and comprehensive preference in an ongoing mashup; (2) an attentional influence encoder, which captures the main focuses of his friends' latest mashups; and (3) a graph attention network, which fuses related neighbors' influence representations to update the target user's preference representation.

current interests, and learns the friend's influences towards the target user. Third, a friend-level graph attention network assigns different weights to the target user's friends based on the target user's current interests, and absorbs the representations of his friends to obtain the target user's mashup representation. Finally, our SGHAN obtains the matching scores by modeling the similarity between each candidate service and the current mashup.

3.1 Dynamic Interest Modeling

To learn a user's fast-changing and comprehensive preference in a specific mashup, we use his current interest to characterize his preference. Specially, we define *current interest* as a user's latent preference hidden in his current ongoing mashup (i.e., the current preference after invoking a sequence of services).

Following the recent advances in session-based recommendations [13], we adopt recurrent neural networks (RNNs) to model the service invocations of the target user in his current mashup. Given the target user's current mashup $M^u_{t+1} = \{s^u_{t+1,1}, s^u_{t+1,2}, \ldots, s^u_{t+1,n}\}$, the RNN infers the user's current state by combining his previous state and the latest service, i.e., $h^u_n = f(s^u_{t+1,n}, h^u_{n-1})$, where h^u_n represents user u's current interest after invoking the n-th service in the mashup. In this article, we adopt a long-short term memory (LSTM) structure [14] as the combination function $f(\cdot,\cdot)$, which helps for alleviating exploding and vanishing gradient. We formulate an LSTM-based model as:

$$\begin{aligned} z^u_n &= \sigma(\mathbf{W}_z[h^u_{n-1}, s^u_{t+1,n}] + b_z) \\ f^u_n &= \sigma(\mathbf{W}_f[h^u_{n-1}, s^u_{t+1,n}] + b_f) \\ o^u_n &= \sigma(\mathbf{W}_o[h^u_{n-1}, s^u_{t+1,n}] + b_o) \\ \tilde{c}^u_n &= \tanh(\mathbf{W}_c[h^u_{n-1}, s^u_{t+1,n}] + b_c) \\ c^u_n &= f^u_n \odot c^u_{n-1} + z^u_n \odot \tilde{c}^u_n \\ h^u_n &= o^u_n \odot \tanh(c^u_n) \end{aligned} \tag{1}$$

where $\sigma(\cdot)$ is the element-wise logistic sigmoid function. We use the last hidden state h^u_n to represent the target user's current interest representation in the current mashup, which can be denoted as p^u.

3.2 Service-Level Attentional Influence Encoder

The main purpose of the service-level attention encoder is to learn the user's main focus in his friend's latest mashup. In other words, the encoder aims to identify the services that may have the most influence on the target user. Meanwhile, the temporal dependency among services in the friend's mashup is also important to maintain. Thus, the architecture of the attentional influence encoder is similar to the LSTM structure described in Section 3.1. As shown in Fig. 3, we again apply an RNN with LSTM units as the basic component, which can be represented as:

$$h^k_n = f(s^k_{t,n}, h^k_{n-1}), \tag{2}$$

where $f(\cdot)$ is the LSTM unit illustrated in Equation 1 and h^k_n is the hidden state of the n-th service in friend k's latest mashup. Intuitively, if a service in the mashup is more relevant to the user's current interest, the user will pay more attention to this service. In order to simulate such intuition between the target user and the services in his friends' mashups, we compute the similarity between the target user's current interest and the service's hidden state in its mashup as the attention score. Such a score will help determine whether the service's influence should be emphasized or ignored when obtaining the friend's preference influence to the target user. Formally, it can be represented as:

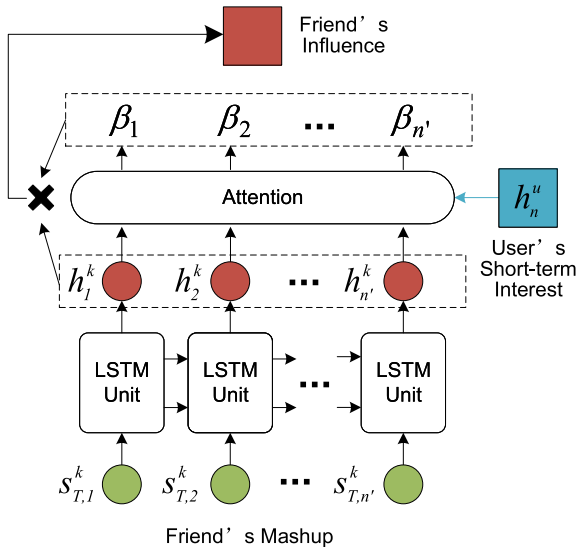


Fig. 3. A schematic view of the attentional influence encoder.

$$\alpha^*_i = q(p^u, h^k_i) = \mathbf{h}^T ReLU(\mathbf{W}_1 p^u + \mathbf{W}_2 h^k_i), \tag{3}$$

where h^k_i is the hidden state of the i-th service in friend k's latest mashup. $\mathbf{W}_1$ and $\mathbf{W}_2$ are matrices that can be learned to project the above representation into a shared latent space.

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

$$\alpha_i = \frac{\exp(\alpha^*_{(i)})}{\sum_{j \in M^k_t} \exp(\alpha^*_{(j)})}. \tag{4}$$

Upon we obtaining the weight factor of each service in a friend's mashup, the friend k's preference influence to the target user can be represented as follows:

$$g^k = \sum_{j \in M^k_t} \alpha_j h^k_j, \tag{5}$$

in which the user's main interest in the k's latest mashup is adaptively focused.

3.3 Friend-Level Graph Attention Network

After obtaining the target user's dynamic interest and his friends' potential influences, we describe the process how the user's preference gets influenced by his friends. In this section, We first introduce how we construct an influence graph, and then describe how we use a graph convolutional network (GCN)-based model to fuse related neighbors' influence representations to update the target user's preference representation.

3.3.1 Influence Graph

For each target user, we build his influence graph where each node corresponds to the user and his direct friends, in which an edge indicates there exists social relation between the user and the corresponding friend. For the target user node, we use his dynamic interest representation p^u constructed in Section 3.1 as the node's initial representation $q_u^{(0)}$. For those neighbors of the target user node, we use the preference influence g^k obtained in Section 3.2 as their initial representations $q_k^{(0)}$, which already encodes the user's main focus in the corresponding friend's latest mashup.

Note that for simplicity, in this work, we only consider influences coming from the direct friends of a target user. Influences from high-order social friends will be considered in our future work.

3.3.2 Graph Attentional Convolution

With all nodes' initial representations defined in the target user's influence graph, in this section, we will first formulate the social influence propagation of one single friend in his influence graph. Then we will extend the single user propagation equation to a matrix form, which helps accelerate the calculation.

Recently, researches on graph neural networks [15], [16], [17] propose to define convolutions directly on a graph, operating on groups of spatially close neighbors. However, most of them utilize the fixed symmetric normalized Laplacian as the propagation strategy, in which the weights of neighbors are relied on predefined static functions without considering friend-level differences existing in social influence. Inspired by the graph attention networks (GATs) [11], we propose to apply the attention mechanism to guide the social influence propagation. Different from Equation 4, we use the inner product followed by a softmax function to represent the similarity between the target user u's dynamic interest and his friend k's influences:

$$\beta_{uk}^{(l)} = \frac{\exp({q_u^{(l)}}^T q_k^{(l)})}{\sum_{j \in N(u)} \exp({q_u^{(l)}}^T q_k^{(l)})}. \tag{6}$$

where $q_u^{(l)}$ is the representation of user u at the l-th convolution layer and $\beta_{uk}^{(l)}$ is the friend-level attention weight of friend k's preference influence to the target user u at the l-th convolution layer.

After we obtain the friend-level attention weights, we combine the features on the user's influence graph together, followed by a non-linear transformation to acquire the node representation of the next convolution layer:

$$\begin{aligned} \hat{q}_u^{(l+1)} &= q_u^{(l)} + \sum_{j \in N(u)} \beta_{uk}^{(l)} q_k^{(l)}, \\ q_u^{(l+1)} &= ReLU(\mathbf{W}^{(l)} \hat{q}_u^{(l+1)} + \mathbf{b}^{(l)}), \end{aligned} \tag{7}$$

where $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are the learnable and shared parameters at the l-th convolution layer. Note that we include a self-connect term $q_u^{(l)}$ to preserve the target user's own dynamic interest. By stacking the graph attentional convolution layer L times, the target user vertex in the influence graph aggregates information with the farthest reach of his L-degree neighbors, which help the final $q_u^{(L)}$ explicitly encodes the high-order preference influence [6].

3.4 Service Prediction

To avoid the issue of gradient vanishing, we embrace a skip connection to add the user's original dynamic interest p^u and his friends' context influence $q_u^{(L)}$ together:

$$p_{final}^u = p^u + q_u^{(L)} \tag{8}$$

where p_{final}^u is the final representation of the target user's preference on the next service in his current mashup. We obtain the probability of service s becoming the next potential service in the target user u's current mashup on the whole service set $\mathbb{S}$ with a softmax function:

$$\begin{aligned} &prob(s|s_{t+1,1}^u, s_{t+1,2}^u, \ldots, s_{t+1,n}^u; \{M_t^k, k \in N(u)\}) \\ &= \frac{\exp(z_s^T p_{final}^u)}{\sum_{s \in \mathbb{S}} \exp(z_s^T p_{final}^u)}, \end{aligned} \tag{9}$$

where z_s is the embedding of the candidate service s.

3.5 Parameter Learning

In this section, we discuss parameter optimization and the training efficiency of our SGHAN framework.

3.5.1 Optimization

The purpose of our optimization process is to maximize the probability of the positive services consumed by the target user in his current mashup. To this end, we choose the negative log-likelihood as loss function to supervise the training process, which is formulated as follows:

$$\begin{aligned} \mathcal{L} = -\sum_{u \in \mathbb{U}} \sum_{t=1}^{T} log\ prob(s_{t+1,m+1}^u | s_{t+1,1}^u, s_{t+1,2}^u, \ldots, \\ s_{t+1,m}^u; \{M_t^k, k \in N(u)\}), \end{aligned} \tag{10}$$

where m represents the number of services in the mashup.

3.5.2 Time Complexity Analysis

The computation in our SGHAN consists of four parts. (1) When inferring a user's dynamic preference, we adopt an LSTM-based structure for learning his current interest, of which the time complexity is $O(md)$, where m denotes the average length of a user's mashup and d denotes the embedding size. (2) In the service-level attentional influence encoder, we also adopt an LSTM-based structure to obtain a user's k friend's hidden state, of which the time complexity would be $O(kmd)$. As discussed earlier, we compute the attention scores for each service in every friend's recent mashup. The time complexity for this procedure will also be $O(kmd)$. (3) In the friend-level graph attention network, the target user's k neighbors will be re-scaled and aggregated to update the target user embedding. The time complexity of stacking L propagation layers would be $O(kdL)$. Hence, the overall theoretical time complexity for each user is $O(kmd) + O(kdL)$. In practice, we usually have $L \leq 3 < m$. In conclusion, our algorithm is computationally feasible

in practice, and thus will support real-time query in real-world service recommendation system.

4 Experiments

In this section, we present our extensive experiments with analysis.

4.1 Dataset Description

Because there lacks a large web service repository for our study, we turned to offline service repository to construct our testbed. Previous researches have shown that offline services and web services share three common features:

- Temporal dependencies exist in both fields. More specifically, a series of services consumed in a time window usually aim for some major purpose and are performed in a relatively fixed order, which can be composed as a "mashup" [2].
- The social network among users exists and friends can have an influence on a user's preference towards services [6].
- Both offline services and web services are massive and heterogeneous [18].

Gowalla is a known website where users can record and share offline services they consume by checking-in, e.g., a meal in a restaurant or a show in a theater. In addition to its service recording function, Gowalla also allows social networks to connect and coordinate users with people or events that match their interests. Thus, we adopted the Gowalla platform to test and evaluate our SGHAN on time-aware service recommendation for mashup creation. Cho et al. [19] randomly collected a total of 6,442,890 service records of 196,591 users over the time period of February 2009 - October 2010. We constructed the associated social network of these users using their public API. To thoroughly evaluate the effectiveness and generalization of our method, we also conducted comparative experiments on another dataset Yelp[1], which is also a famous offline service rating platform. We collected a total of 10,407 service records with 6,005 users over the period of January 2015 - June 2016. Table 2 summarizes the numerical properties of these two datasets.

In the task of time-aware service recommendation, we aim to provide a suitable service for the target user based on his recent consumed services. Thus the recommended service can cooperate with other services for the same purpose. We treated each check-in in the dataset as a service consumption. Based on the empirical frequency of the offline consumption, we segmented the data into day-long sessions and each session can be considered as a mashup. We chronologically reserved the last six months for testing and filtered out the services that did not appear in the training set. For those services reserved mashups, we randomly and equally split them into validation set and test set.

TABLE 2
Statistics of the Datasets

Item	Gowalla	Yelp
Users	11,459	6,005
Services	16,435	10,470
Invocations	1,098,766	122,526
Avg. invocations per user	95.89	20.41
Avg. services per mashup	6.11	5.79
Social Connections	94,764	111,931
Density(Social Connections)	0.0721%	11,459
Avg. friends per user	8.27	18.64

4.2 Experimental Settings

4.2.1 Baselines

We compared our SGHAN with four classes of baseline methods to evaluate the performance of time-aware service recommendation: (1) classical methods utilizing only users' implicit feedback; (2) social-aware methods that incorporate social influence; (3) time-aware methods that consider a user's actions in his current mashup; (4) graph-based time-aware models that consider both social relations and temporal information of users. Below, we list six baseline models used in our experiments. The class index of each model is indicated beside its name.

- *BPR-MF [20] (1):* This method is a highly competitive method for implicit feedback based recommendation. It improves matrix factorization (MF) with the BPR objective function.
- *SBPR [21] (2):* This is a ranking model that considers social relationships in the learning process, assuming that users tend to assign higher ranks to items that their friends prefer.
- *SoReg [22] (2):* This method utilizes social network to regularize the latent user factors of matrix factorization.
- *GRU4Rec [12] (3):* This is a recent state-of-the-art approach that uses recurrent neural networks for session-based recommendations.
- *NARM [9] (3):* This method employs RNNs with an attention mechanism to capture a user's main purpose and sequential behavior.
- *DGRec [13] (4):* This method models dynamic user behaviors with an RNN, and context-dependent social influence with graph attention neural networks based on the given social networks.

4.2.2 Evaluation Metrics

To evaluate the performance of all algorithms in the experiments, we adopted two popular metrics, namely Normalized Discounted Cumulative Gain@K (NDCG@K) and Hit Ratio@K (HR@K). Both metrics are the higher, the better. The former NDCG@K metric accounts for the position of the hits by assigning higher scores to hits at top ranks and thus is position-aware. The latter HR@K metric measures whether the test item is present on the recommendation list or not. The above two metrics are formulated as:

$$NDCG@K = \frac{1}{R_N} \sum_{i=1}^{N} \frac{2^{rel_i - 1}}{\log_2(1+i)} \tag{11}$$

$$HR@K = \frac{\sum_{i=1}^{K} rel_i}{|y_u^{test}|} \tag{12}$$

1. Data set available from https://www.yelp.com/dataset

TABLE 3
Overall Performance Comparison for Gowalla

Model	NDCG@10	HR@10	NDCG@20	HR@20
BPR-MF [20]	0.0351	0.0672	0.0405	0.0889
SBPR [21]	0.0465	0.0768	0.0509	0.0941
SoReg [22]	0.0442	0.0740	0.0498	0.0963
GRU4Rec [12]	0.0901	0.1574	0.1038	0.2122
NARM [9]	0.0905	0.1568	0.1047	0.2133
S-Walk [27]	0.0967	0.1709	0.1230	0.2365
DGRec [13]	0.1171	0.2051	0.1356	0.2788
SGHAN	**0.1240**	**0.2192**	**0.1436**	**0.2969**

TABLE 4
Overall Performance Comparison for Yelp

Model	NDCG@10	HR@10	NDCG@20	HR@20
BPR-MF [20]	0.0052	0.0114	0.0092	0.0280
SBPR [21]	0.0079	0.0184	0.0114	0.0325
SoReg [22]	0.0114	0.0228	0.015	0.0375
GRU4Rec [12]	0.0142	0.0307	0.0211	0.0580
NARM [9]	0.0150	0.0331	0.0203	0.0545
S-Walk [27]	0.0169	0.0351	0.0230	0.0594
DGRec [13]	0.0181	0.0374	0.0244	0.0623
SGHAN	**0.0208**	**0.0439**	**0.0293**	**0.0780**

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 services used by user u in the test set.

4.2.3 Implementation Details

All of our experiments were conducted on an Ubuntu server [CPU: Intel Xeon E5-2680 v4 * 2, GPU: NVIDIA RTX 3090 * 4, RAM: 384GB]. We implemented SGHAN on the basis of Pytorch [23], a widely used Python library for neural networks.

We initialized the latent vectors with small random values for all the models that are using latent factors. For a fair comparison, the dimension of the service latent factors is fixed to 100. We also set the number of hidden units of the LSTMs or RNNs in all models as 100. We set the batch size of all experiments as 200. We used Adam optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e^{-8}$. The initial learning rate was set as 0.002 and decayed at the rate of 0.9 every five epochs. We applied the Adam optimizer for all baseline models, where the batch size was fixed as 200.

The parameters for baseline methods were initialized as in common practices, and were then carefully tuned to achieve optimal performances. we applied a grid search for hyper-parameters. The learning rate for all models were tuned amongst [0.005, 0.01, 0.02, 0.05]. To prevent overfitting, we added L_2 norm with coefficient tuned from [0.001, 0.005, 0.01, 0.02, 0.1]. All the latent factors in the baseline models were initialized with small random values. We selected the best models by early stopping when the HR@20 on the validation set stops increasing for three consecutive epochs.

In our SGHAN, we used an embedding matrix to represent services' features, which means we projected a service s into a latent space with an embedding vector e_s. There are many embedding methods that can be utilized for service representation learning in our work. For example, Lam et al. [24] analyze service descriptions (i.e., WSDL file) to obtain service embeddings. Some methods propose to learn service embeddings based on their functionality [25] or non-functional features (i.e., QoS) [26]. Since those embedding techniques are not our focus in this article, without losing generality, we only used the ID feature of services to obtain the initial embeddings in our experiments to illustrate the effectiveness of our approach.

4.3 Performance Comparisons

The empirical results of our SGHAN and the baselines, in terms of HR@K and NDCG@K with recommendation size $K = 10, 20$ over both service datasets, are summarized in Tables 3 and 4, respectively. We conducted one-sample t-tests and $p - value < 0.05$ indicates that the improvements of SGHAN over the strongest baseline are statistically significant. Besides, we have the following observations:

- BPR-MF shows very limited performance since it only considers a user's overall invocation histories for inferring his long-term static interests. Based on BPR-MF, the methods leveraging social network information, i.e., SBPR and SoReg, perform better than the ones without it. The experiment results demonstrate that social neighbors' information could help improve service recommendation.
- The methods incorporating temporal information bear better performances than those which do not. For example, in Table 3 for all metrics, GRU4Rec and NARM significantly outperform BPR-MF, SBPR, and SoReg. It might be because the temporal dependency of services consumed by a user implicitly contains the current interest of the user in the current mashup, which is of great help to recommend the next service for the target user.
- The graph-based methods, including our SGHAN, consistently outperform the other methods for all metrics. These results provide trustworthy evidence for incorporating users' dynamic interest and social relations synergistically into time-aware service recommendation.
- SGHAN outperforms DGrec by a large margin. DGrec represents friends' social influence, by directly performing an RNN-based model on their latest mashups. However, the friend's current interest is not equivalent to the influence to the target user, since the user has his own service-level focus inside his friends' latest mashups. The experimental results demonstrate that our service-level attention inside a friend's mashup has more vital expressiveness in the social influences to capture the target user's main focus.

4.4 Study of Service-Level Differences

The service-level attentional influence encoder is one of the key characteristics in our proposed SGHAN, which helps

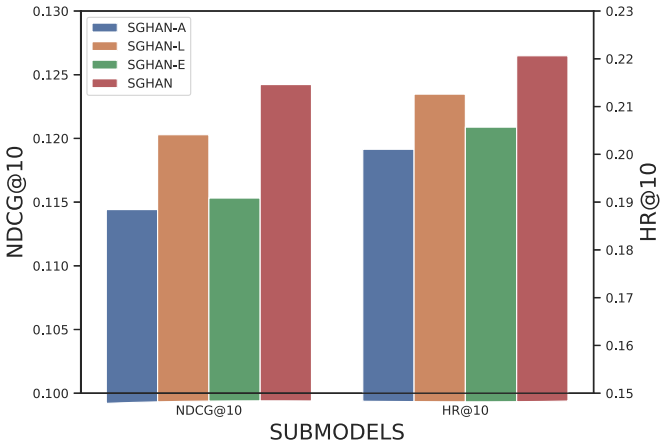


Fig. 4. Performance of variants of SGHAN.

capture target user's main purpose in his friends' latest mashups. We designed experiments to further explore the impacts of the attentional influence encoder in SGHAN and analyze the effectiveness of the adopted service-level attention mechanism, by constructing the following variants of SGHAN:

- *SGHAN-A:* A variant model of SGHAN, in which the user has the same focus on all the comprising services in a friend's mashup, i.e., the weights of the services in Equation 4 are set equal.
- *SGHAN-L:* A variant model of SGHAN, which only considers the last hidden state of a friend's mashup as the social influence to the target user.
- *SGHAN-E:* A variant model of SGHAN, which does not consider temporal dependency in a friend's mashup by averaging the service embeddings in the mashup.

Fig. 4 shows the experimental results of SGHAN and its three variants. For the interest of space, here we only show the results of the NDCG@10 and HR@10 on the Gowalla dataset. From Fig. 4, two observations can be made.

First, SGHAN-A and SGHAN-E perform the worst. SGHAN-E fails to consider the temporal dependencies among services in a mashup, thus may not precisely capture user's dynamic preferences. The mean-pooling operation in SGHAN-A blurs the temporal information captured by the LSTM structure to some extent. Second, SGHAN-L performs better because it leverages LSTM's capacity for capturing the temporal dependency in the last hidden state of the mashup. Third, the performance of SGHAN is significantly better than SGHAN-L, which shows that the service-level attention can help better utilize temporal information in the hidden state and capture the user purpose in the current mashup.

4.5 Study of Friend-Level Differences

To justify and gain further insights of the specifics of SGHAN's friend-level differences, we conducted ablation studies to validate the effectiveness of our proposed friend-level graph attention network.

As shown in Equation 8, SGHAN obtains the user's final representation by combining the user's original dynamic interest p^u and his friends' context influence $q_u^{(L)}$. To tease apart the contribution of both sources of information, we compared SGHAN against two variants as follows:

TABLE 5
Ablation Study Comparing SGHAN and Its Variants

Variation	NDCG@10	HR@10	NDCG@20	HR@20
SGHAN_self	0.0901	0.1574	0.1038	0.2122
SGHAN_equal	0.1227	0.2160	0.1428	0.2957
SGHAN	0.1240	0.2192	0.1436	0.2969

- *SGHAN_equal:* A variant model in which the weights of the neighbors in Equation 7 are set equal.
- *SGHAN_self:* A variant model considering the user's original dynamic interest p^u only. Note that the SGHAN_self is identical to the GRU4Rec method [12], and thus we reuse the result in Table 3 for consistency.

Table 5 shows the performance of different variants. SGHAN and SGHAN_equal outperform SGHAN_self significantly on all the metrics, which means social influence has essential impacts on the service recommendation quality. Compared to the full SGHAN, the SGHAN_equal has worse performance. The results show that the friend-level difference considered in the graph attention network can make sure that the friends who share common preferences with the target user can have a more prominent influence on the user.

4.6 Effect of Convolution Layer

SGHAN aggregates friends' influence to revise the target user's preference representation and by stacking more layers, SGHAN will pass influences from high-order friends to the target user. To investigate whether SGHAN can benefit from such multiple convolution layers, we experimented the layer numbers from 1 to 5. Fig. 5 shows the results, wherein SGHAN-l denotes the model with l convolution layers. From Fig. 5, we have the following two observations:

First, Fig. 5 shows that our SGHAN with more than one single convolution layer have better performance. This implies that the social influence from a user's high-order friends may further enhance service recommendation.

Second, there is a small improvement from SGHAN-3 to SGHAN-4 and when further stacking more convolution layers on top of SGHAN-4, both metrics start to decrease. This might because in the learning process of a user's representation, a

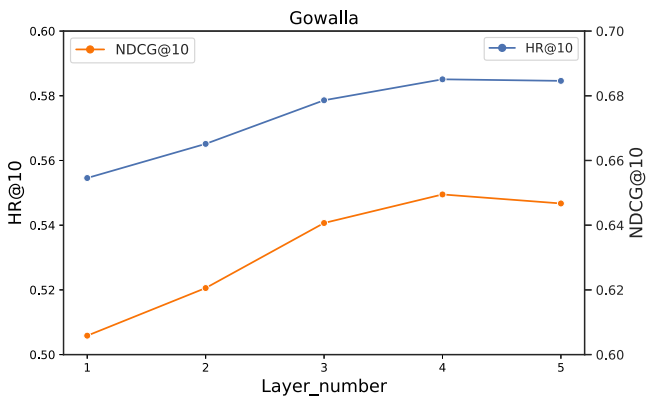


Fig. 5. Performance of SGHAN under different Convolution Layers.

TABLE 6
Computation Cost on Different User Scale

User Scale	Training Time (s/epoch)		Memory Occupation (MiB)	
	SGHAN	DGRec	SGHAN	DGRec
2K	7	5	2387	2337
10K	83	74	4284	4179
20K	295	287	7575	7315
50K	1900	1833	19286	18345

Some results are infeasible with limited resources.

too deep architecture might introduce noise and the neighbors far from the user have little marginal improvement to offset the noise. This finding confirms that 3 or 4 convolution layers are optimal for our scenario.

4.7 Computational Cost

Beyond the theoretical analysis of time complexity in Section 3.5.2, we compared the empirical computation cost of our SGHAN with another graph-based time-aware model DGRec on the synthetic datasets with varied numbers of users. We constructed the synthetic datasets by randomly sampling 1,000 users alongside their invocation history from the *Gowalla* dataset and replicated them different times to create datasets of different user scales.

Table 6 shows the training time and the memory occupation with one single GTX3090. The training time is greatly affected by the number of training instances (i.e., the number of invocations), while the number of users more influences the memory occupation. From Table 6, it can be observed that in terms of training time and memory occupation, both our SGHAN and the SOTA DGRec perform in the same order of magnitude. This shows that our method can achieve significant performance improvement on the time-aware service recommendation, while the amount of computation resource is effectively controlled. Meanwhile, the experiment verifies that our model has high scalability to be deployed in a real-world service recommendation system, since it can cover the dynamic preference modeling of 50,000 users with limited memory on a single RTX3090.

5 Related Work

In this section, we discuss related work from three aspects: mashup creation, dynamic recommendation, and graph neural networks.

5.1 Mashup Creation

Service discovery has been one key topic in the field of services computing. Semantics-aware methods [28], [29] mainly focus on information retrieval and similarity calculation, which usually extract semantic information, e.g., keywords and labels, and calculate relevance scores represented by semantic distance. Quality-of-Service (QoS) [18], [26] is widely employed to represent nonfunctional performance of web services and has been adopted as a key factor in service selection. These traditional methods of service recommendation are insufficient to mashup creation, because most of them focus on suggesting individual services separately based on a hypothesis that user demands are independent.

In recent years, increasingly more research works focus on service recommendation for mashup creation. Elmeleegy et al. [30] propose MashupAdvisor, which utilizes a semantic matchmaking algorithm and a metric planner to modify a mashup toward producing suggested output. Bianchini et al. [31] present a recommendation system to provide proactive suggestions to mashup designer, relying on the semantic descriptions of mashup components. In [32], a novel category-aware service clustering and distributed recommending method is proposed for automatic mashup creation. To alleviate the cold-start problem, Bai et al. [5] modify the generative process of a mashup and recommend services through both content information and historical usage analysis. To capture the deep relationships among services, Cao et al. [33] formalize service package recommendation as a Quadratic Knapsack Problem and solve it using Branch and Bound algorithm.

With the rapid development of Deep Learning (DL), researchers have started to investigate the potential of DL for mashup creation and recommendation. For example, Yao et al. [34] propose a probabilistic matrix factorization approach with implicit correlation regularization, to solve the recommendation for mashups with enhanced recommendation diversity. Wu et al. [35] propose a neural framework (MTFM) based on multi-model fusion and multi-task learning for accurate mashup creation. Gu et al. [36] propose a compositional semantics-based service bundle recommendation model (CSBR) to tackle the semantic gap between mashup descriptions and service descriptions. Cao et al. [37] propose an integrated content and network-based service clustering and recommendation method for mashup development.

Note that the aforementioned works on service discovery assume that all service users are independent. With the prevalence of social media, social influence is pervasive, not only in our daily physical life but also in the virtual webspace. Under such a context, some recent approaches [6], [38] propose to integrate users' social connections. To model cross-service dependency under social influence, Cao et al. [8] propose a mashup service recommendation approach based on user interests and relations among services. Xu et al. [39] propose a coupled matrix model to describe the multi-dimensional relationships among users, mashups, and services.

Existing studies, though inspiring, are limited to providing a fixed set of services centered on a specific functionality as a mashup. They typically neglect the fact that users' functional requirements change dynamically over time. Moreover, the social influence strengths in most of the works are assumed equal among friends or with a simple metric from other sources (e.g., the strengths between their interactions in the past). In contrast, our work proposes to capture target user's main interests in his friends' consumed mashup in a more fine-grained manner.

5.2 Dynamic Recommendation

Modeling users' dynamic interests that change over time has drawn increasing attention in recent years [40], [41], [42]. For example, Wang et al. [40] reveal that the concept drifts of users' temporal preferences are commonly seen during a period of time. They propose CD-APIR, a concept drift-aware temporal cloud service APIs recommendation approach for building composite cloud systems. Such kind

of temporal dependencies commonly exist in interaction data, but cannot be well captured by the conventional content-based recommender systems or collaborative filtering technique [43]. This gap motivates the development of session-based recommendation or sequential recommendation.

Existing methods usually extract users' current interest by exploring multiple user-service interactions that happen successively in a sequence in a continuous time period, which usually lasts for several minutes to several hours [2]. Related works can be further divided into three categories, i.e., conventional approaches, latent representation-based approaches, and neural network (NN)-based approaches. Conventional approaches leverage conventional data mining or machine learning techniques. For example, Linden et al. [44] propose an item-to-item collaborative filtering method to personalize the online store for each customer. Sarwar et al. [45] look into different techniques for computing item-item similarities and compare their results with basic k-nearest neighbor approaches. Shani et al. [46] propose a Markov Decision Processes (MDP) aiming to provide recommendations in a session-based manner. The simplest MDP boils down to first-order Markov chains, where the next recommendation can be computed through the transition probabilities between items. Latent representation-based approaches, like [47], make recommendations with low-dimensional latent representations for each interaction within sessions with shallow models.

NN-based approaches aim to effectively capture users' current interests. For example, GRU4Rec [12] applies gated recurrent units (GRU) to model sequential behaviors and make recommendations. NARM [9], STAMP [48] and DSAN [49] utilize attention mechanisms to distinguish short- and long-term item dependency. Recently, SR-GNN [50], NISER+ [51], and GCE-GNN [52] exploit graph neural networks (GNNs) to further analyze complex item transitions. In these methods, sequential user behaviors are modeled as graph-structured data. To capture high-order relations among items, DHCN [53] propose a dual channel hypergraph convolutional network and S-Walk [27] utilize random walks with restart (RWR) for session-based recommendation.

In contrast to existing work, we model dynamic user behaviors with an RNN-based module and social relationships among users as graph-structured data.

5.3 Graph Neural Networks

Research on graph neural networks (GNNs) has rapidly gained significant momentum in recent years, since many real-world data can be represented by graphs conveniently, e.g., social networks, citation networks, and road maps. It is known that convolutional neural networks (CNNs) have achieved great success in computer vision and several other application fields. Somre recent works focus on modeling general graph-structured data using CNNs [15], [16]. Specifically, Kipf et al. [16] propose graph-convolutional networks (GCNs) for semi-supervised graph classification, which directly propagate node information along edges in the spatial domain. In addition, Bruna et al. [17] propose a spectral-based graph convolutional network by introducing the spectral graph theory into neural networks. However, the weights of all neighbors in these methods rely on predefined static functions when updating the node representations. Graph attention networks (GATs) address this issue by considering the different weights to the central node from different neighbors.

Some recent studies, like DGRec [13], consider the friend-level differences to improve time-aware recommendation. However, they do not exploit the service-level differences in a friend's mashup thus failing to conjecture the target user's main interest in that mashup. In contrast, our research synergistically considers service-level and friend-level differences, and proposes a finer-grained model for a target user to learn from proper friends' behaviors guided by his dynamic preference.

6 Conclusion

To adapt to the rapid shifts of users' dynamic preference, time-aware service recommendation exploits a target user's service invocation behaviors within a period of time and predicts the next service to form a dynamic mashup. With the prevalence of online social networks, increasingly more service platforms tend to leverage the social networks among users to enhance service recommendation quality. However, when modeling the dynamic social influence, two levels of differences that influence the inference of users' dynamic preferences have not been well studied. They are *Service-level Differences* and *Friend-level Differences*.

In this article, we present SGHAN to recommend services for dynamic mashup creation, which unifies a *service-level attentional encoder* and a *friend-level graph attention network* to model social influences hierarchically. As a hierarchical deep learning model, SGHAN learns a target user's dynamic interest, and based on it, studies from his social networks to learn their similar behaviors and social influences. The resulted user preference representation leads to effective service recommendation on-the-fly.

For simplicity, in this work we only model the social influences as all friends' interests right before the target user starts to develop the current mashup. In our future work, we plan to study social influence from target user's friends' richer past behaviors. Furthermore, we try to explore more service attributes, such Qos and prices, and extract more richer relationships among services. Moreover, in a real-world scenario, there may exist users sharing common interests while they do not even know each other. It would be interesting and we plan to extract such relationships to help further enhance service recommendation.

References

[1] G. Fazelnia, E. Simon, I. Anderson, B. Carterette, and M. Lalmas, "Variational user modeling with slow and fast features," in *Proc. 15th ACM Int. Conf. Web Search Data Mining*, 2022, pp. 271–279.
[2] Y. Zhong, Y. Fan, K. Huang, W. Tan, and J. Zhang, "Time-aware service recommendation for mashup creation," *IEEE Trans. Services Comput.*, vol. 8, no. 3, pp. 356–368, May/Jun. 2015.
[3] K. Huang, Y. Fan, and W. Tan, "Recommendation in an evolving service ecosystem based on network prediction," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 3, pp. 906–920, Jul. 2014.
[4] C. Yu and L. Huang, "Time-aware collaborative filtering for QoS-based service recommendation," in *Proc. IEEE Int. Conf. Web Serv.*, 2014, pp. 265–272.
[5] B. Bai, Y. Fan, K. Huang, W. Tan, B. Xia, and S. Chen, "Service recommendation for mashup creation based on time-aware collaborative domain regression," in *Proc. IEEE Int. Conf. Web Serv.*, 2015, pp. 209–216.

[6] C. Wei, Y. Fan, J. Zhang, and H. Lin, "A-HSG: neural attentive service recommendation based on high-order social graph," in *Proc. IEEE Int. Conf. Web Serv.*, 2020, pp. 338–346.

[7] T. Liang, L. Chen, J. Wu, G. Xu, and Z. Wu, "SMS: A framework for service discovery by incorporating social media information," *IEEE Trans. Services Comput.*, vol. 12, no. 3, pp. 384–397, May/Jun. 2019.

[8] B. Cao, J. Liu, M. Tang, Z. Zheng, and G. Wang, "Mashup service recommendation based on user interest and social network," in *Proc. IEEE 20th Int. Conf. Web Serv.*, 2013, pp. 99–106.

[9] J. Li, P. Ren, Z. Chen, Z. Ren, T. Lian, and J. Ma, "Neural attentive session-based recommendation," in *Proc. ACM Conf. Informat. Knowl. Manage.*, 2017, pp. 1419–1428.

[10] A. Vaswani et al., "Attention is all you need," in *Proc. Adv. Neural Informat. Process. Syst.*, 2017, pp. 5998–6008.

[11] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. 6th Int. Conf. Learn. Representations*, 2018.

[12] B. Hidasi, A. Karatzoglou, L. Baltrunas, and D. Tikk, "Session-based recommendations with recurrent neural networks," in *Proc. 4th Int. Conf. Learn. Representations*, 2016.

[13] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, and J. Tang, "Session-based social recommendation via dynamic graph attention networks," in *Proc. 12th ACM Int. Conf. Web Search Data Mining*, 2019, pp. 555–563.

[14] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Informat. Process. Syst.*, 2016, pp. 3837–3845.

[16] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Representations*, 2017.

[17] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proc. 2nd Int. Conf. Learn. Representations*, 2014.

[18] S. Wang, Y. Ma, B. Cheng, F. Yang, and R. N. Chang, "Multi-dimensional QoS prediction for service recommendations," *IEEE Trans. Services Comput.*, vol. 12, no. 1, pp. 47–57, Jan./Feb. 2019.

[19] E. Cho, S. A. Myers, and J. Leskovec, "Friendship and mobility: User movement in location-based social networks," in *Proc. 17th ACM Int. Conf. Knowl. Discov. Data Mining*, 2011, pp. 1082–1090.

[20] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proc. 25th Conf. Uncertainty Artif. Intell.*, 2009, pp. 452–461.

[21] T. Zhao, J. J. McAuley, and I. King, "Leveraging social connections to improve personalized ranking for collaborative filtering," in *Proc. 23rd ACM Int. Conf. Conf. Informat. Knowl. Manage.*, 2014, pp. 261–270.

[22] H. Ma, D. Zhou, C. Liu, M. R. Lyu, and I. King, "Recommender systems with social regularization," in *Proc. 4th Int. Conf. Web Search Web Data Mining*, 2011, pp. 287–296.

[23] A. Paszke et al., "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Conf. Neural Informat. Process. Syst.*, 2019, pp. 8024–8035.

[24] G. Lam and D. Rossiter, "A web service framework supporting multimedia streaming," *IEEE Trans. Services Comput.*, vol. 6, no. 3, pp. 400–413, *Third Quarter* 2013.

[25] A. V. Paliwal, B. Shafiq, J. Vaidya, H. Xiong, and N. R. Adam, "Semantics-based automated service discovery," *IEEE Trans. Services Comput.*, vol. 5, no. 2, pp. 260–275, *Second Quarter* 2012.

[26] Z. Zheng, H. Ma, M. R. Lyu, and I. King, "QoS-aware web service recommendation by collaborative filtering," *IEEE Trans. Services Comput.*, vol. 4, no. 2, pp. 140–152, *Second Quarter* 2011.

[27] M. Choi, J. Kim, J. Lee, H. Shim, and J. Lee, "S-Walk: Accurate and scalable session-based recommendation with random walks," in *Proc. 15th ACM Int. Conf. Web Search Data Mining*, 2022, pp. 150–160.

[28] C. Li, R. Zhang, J. Huai, X. Guo, and H. Sun, "A probabilistic approach for web service discovery," in *Proc. IEEE Int. Conf. Serv. Comput.*, 2013, pp. 49–56.

[29] L. Chen, Y. Wang, Q. Yu, Z. Zheng, and J. Wu, "WT-LDA: user tagging augmented LDA for web service clustering," in *Proc. 11th Int. Conf. Service-Oriented Comput.*, 2013, pp. 162–176.

[30] H. Elmeleegy, A. Ivan, R. Akkiraju, and R. Goodwin, "Mashup advisor: A recommendation tool for mashup development," in *Proc. IEEE Int. Conf. Web Serv.*, 2008, pp. 337–344.

[31] D. Bianchini, V. D. Antonellis, and M. Melchiori, "A recommendation system for semantic mashup design," in *Proc. Int. Workshops Database Expert Syst. Appl.*, 2010, pp. 159–163.

[32] B. Xia, Y. Fan, W. Tan, K. Huang, J. Zhang, and C. Wu, "Category-aware API clustering and distributed recommendation for automatic mashup creation," *IEEE Trans. Services Comput.*, vol. 8, no. 5, pp. 674–687, Sep./Oct. 2015.

[33] J. Cao, Y. Lu, and N. Zhu, "Service package recommendation for mashup development based on a multi-level relational network," in *Proc. 14th Int. Conf. Service-Oriented Comput.*, 2016, pp. 666–674.

[34] L. Yao, X. Wang, Q. Z. Sheng, B. Benatallah, and C. Huang, "Mashup recommendation by regularizing matrix factorization with API co-invocations," *IEEE Trans. Services Comput.*, vol. 14, no. 2, pp. 502–515, Mar./Apr. 2021.

[35] H. Wu, Y. Duan, K. Yue, and L. Zhang, "Mashup-oriented web API recommendation via multi-model fusion and multi-task learning," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2021.3098756.

[36] Q. Gu, J. Cao, and Y. Liu, "CSBR: A compositional semantics-based service bundle recommendation approach for mashup development," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2021.3085491.

[37] B. Cao, X. F. Liu, M. M. Rahman, B. Li, J. Liu, and M. Tang, "Integrated content and network-based service clustering and web APIs recommendation for mashup development," *IEEE Trans. Services Comput.*, vol. 13, no. 1, pp. 99–113, Jan./Feb. 2020.

[38] M. Tang, Y. Xu, J. Liu, Z. Zheng, and X. F. Liu, "Trust-aware service recommendation via exploiting social networks," in *Proc. IEEE Int. Conf. Serv. Comput.*, 2013, pp. 376–383.

[39] W. Xu, J. Cao, L. Hu, J. Wang, and M. Li, "A social-aware service recommendation approach for mashup creation," in *Proc. 20th IEEE Int. Conf. Web Serv.*, 2013, pp. 107–114.

[40] L. Wang, Y. Zhang, and X. Zhu, "Concept drift-aware temporal cloud service APIs recommendation for building composite cloud systems," *J. Syst. Softw.*, vol. 174, 2021, Art. no. 110902.

[41] A. Zenebe, L. Zhou, and A. F. Norcio, "User preferences discovery using fuzzy models," *Fuzzy Sets Syst.*, vol. 161, no. 23, pp. 3044–3063, 2010.

[42] Q. Zhang, D. Wu, G. Zhang, and J. Lu, "Fuzzy user-interest drift detection based recommender systems," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, 2016, pp. 1274–1281.

[43] S. Wang, L. Hu, Y. Wang, L. Cao, Q. Z. Sheng, and M. A. Orgun, "Sequential recommender systems: Challenges, progress and prospects," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 6332–6338.

[44] G. Linden, B. Smith, and J. York, "Amazon.com recommendations: Item-to-item collaborative filtering," *IEEE Internet Comput.*, vol. 7, no. 1, pp. 76–80, Jan./Feb. 2003.

[45] B. M. Sarwar, G. Karypis, J. A. Konstan, and J. Riedl, "Item-based collaborative filtering recommendation algorithms," in *Proc. 10th Int. World Wide Web Conf.*, 2001, pp. 285–295.

[46] G. Shani, D. Heckerman, and R. I. Brafman, "An MDP-based recommender system," *J. Mach. Learn. Res.*, vol. 6, pp. 1265–1295, 2005.

[47] L. Hu, L. Cao, S. Wang, G. Xu, J. Cao, and Z. Gu, "Diversifying personalized recommendation with user-session context," in *Proc. 16th Int. Joint Conf. Artif. Intell.*, 2017, pp. 1858–1864.

[48] Q. Liu, Y. Zeng, R. Mokhosi, and H. Zhang, "STAMP: short-term attention/memory priority model for session-based recommendation," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2018, pp. 1831–1839.

[49] J. Yuan, Z. Song, M. Sun, X. Wang, and W. X. Zhao, "Dual sparse attention network for session-based recommendation," in *Proc. 35th AAAI Conf. Artif. Intell.*, 2021, pp. 4635–4643.

[50] S. Wu, Y. Tang, Y. Zhu, L. Wang, X. Xie, and T. Tan, "Session-based recommendation with graph neural networks," in *Proc. 33rd AAAI Conf. Artif. Intell.*, 2019, pp. 346–353.

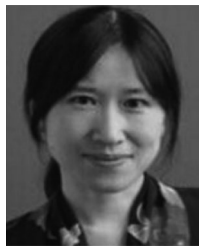
[51] P. Gupta, D. Garg, P. Malhotra, L. Vig, and G. Shroff, "NISER: normalized item and session representations with graph neural networks," 2019, *arXiv: 1909.04276*.

[52] Z. Wang, W. Wei, G. Cong, X. Li, X. Mao, and M. Qiu, "Global context enhanced graph neural networks for session-based recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Informat. Retrieval*, 2020, pp. 169–178.

[53] X. Xia, H. Yin, J. Yu, Q. Wang, L. Cui, and X. Zhang, "Self-supervised hypergraph convolutional networks for session-based recommendation," in *Proc. 35th AAAI Conf. Artif. Intell.*, 2021, pp. 4503–4511.

Chunyu Wei received the BS degree in control theory and application from Tsinghua University, China, in 2019. He is currently working toward the PhD degree with the Department of Automation, Tsinghua University. His research interests include services computing, service recommendation, and social computing.

Yushun Fan received the PhD degree in control theory and application from Tsinghua University, China, in 1990. He is currently a professor with the Department of Automation, director with the System Integration Institute, and director with the Networking Manufacturing Laboratory, Tsinghua University. From September 1993 to 1995, he was a visiting scientist, supported by Alexander von Humboldt Stiftung, with the Fraunhofer Institute for Production System and Design Technology (FHG/IPK), Germany. He has authored 10 books and published more than 300 research papers in journals and conferences. His research interests include enterprise modeling methods and optimization analysis, business process reengineering, workflow management, system integration, object-oriented technologies and flexible software systems, petri nets modeling and analysis, and workshop management and control.



Jia Zhang (Senior Member, IEEE) received the MS and BS degrees in computer science from Nanjing University, China and the PhD degree in computer science from the University of Illinois at Chicago. She is currently the Cruse C. and Marjorie F. Calahan Centennial Chair in engineering, professor with the Department of Computer Science at Southern Methodist University. Her research interests emphasize the application of machine learning and information retrieval methods to tackle data science infrastructure problems, with a recent focus on scientific workflows, provenance mining, software discovery, knowledge graph, and interdisciplinary applications of all of these interests in the area of earth science. She has published more than 170 refereed journal papers, book chapters, and conference papers. She is currently an associate editor of the *IEEE Transactions on Services Computing* (TSC).