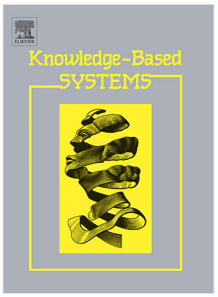
Journal Pre-proof



Motif-based graph attentional neural network for web service recommendation

Guiling Wang, Jian Yu, Mo Nguyen, Yuqi Zhang, Sira Yongchareon, Yanbo Han

PII: S0950-7051(23)00262-9
DOI: https://doi.org/10.1016/j.knosys.2023.110512
Reference: KNOSYS 110512

To appear in: *Knowledge-Based Systems*

Received date : 21 November 2022
Revised date : 23 March 2023
Accepted date : 24 March 2023

Please cite this article as: G. Wang, J. Yu, M. Nguyen et al., Motif-based graph attentional neural network for web service recommendation, *Knowledge-Based Systems* (2023), doi: https://doi.org/10.1016/j.knosys.2023.110512.

This is a PDF file of an article that has undergone enhancements after acceptance, such as the addition of a cover page and metadata, and formatting for readability, but it is not yet the definitive version of record. This version will undergo additional copyediting, typesetting and review before it is published in its final form, but we are providing this version to give early visibility of the article. Please note that, during the production process, errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

© 2023 Published by Elsevier B.V.

Journal Pre-proof

Credit Author Statement

Guiling Wang: Methodology, Writing, Validation, Visualization. **Jian Yu**: Conceptualization, Methodology. **Mo Nguyen:** Writing- Original draft preparation, Software. **Yuqi Zhang:** Writing- Original draft preparation, Software. **Sira Yongchareon**: Software, Validation.: **Yanbo Han**: Project administration, Funding acquisition.

Journal Pre-proof

Journal Pre-proof

Revised Manuscript (Clean Version)

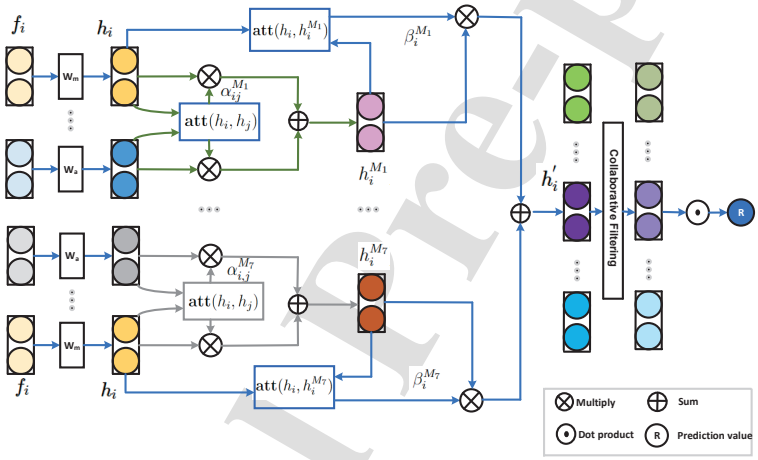
Click here to view linked References

Graphical Abstract

Motif-based Graph Attentional Neural Network for Web Service Recommendation

Guiling Wang, Jian Yu, Mo Nguyen, Yuqi Zhang, Sira Yongchareon, Yanbo Han

Corresponding Author: Jian Yu



Journal Pre-proof

Journal Pre-proof

Highlights

Motif-based Graph Attentional Neural Network for Web Service Recommendation

Guiling Wang, Jian Yu, Mo Nguyen, Yuqi Zhang, Sira Yongchareon, Yanbo Han

- We report a Motif-based Graph Attention Network for service recommendation (MGSR).
- The model aggregates high-order information of motif-based neighbours into the embedding progress.
- The model identifies all the seven up-to-four-node motifs and generates a motif adjacency matrix for each motif.
- The model simultaneously learns the weights of different nodes from different motifs, and shares an attention network to calculate attention scores between nodes within a motif-induced graph as well as between nodes across different motif-induced graphs.

Journal Pre-proof

Journal Pre-proof

Motif-based Graph Attentional Neural Network for Web Service Recommendation

Guiling Wang[a], Jian Yu[b,*], Mo Nguyen[b], Yuqi Zhang[b], Sira Yongchareon[b], Yanbo Han[a]

[a]*Beijing Key Laboratory on Integration and Analysis of Large-Scale Stream Data, School of Information Science and Technology, North China University of Technology, Beijing, China*
[b]*Department of Computer Science and Software Engineering, Auckland University of Technology, Auckland, New Zealand*

Abstract

Deep Neural Networks (DNN) based collaborative filtering has been successful in recommending services by effectively generalizing graph-structured data. However, most existing approaches focus on first-order interactions. Although recent approaches have utilized high-order connectivity, they still limit themselves to simple interactions and ignore the pattern of structural sub-graphs/motifs. In this study, we first explore the commonly used motifs in the Mashup-API interaction bipartite graph and propose a dedicated algorithm to generate the motif adjacency matrix. We then propose a Motif-based Graph Attention Network for service recommendation (MGSR) that utilizes a motif-based attention mechanism to capture the high-order information of various motifs, and a Collaborative Filtering model to generate the recommendation prediction. We have conducted extensive experiments on ProgrammableWeb dataset and our results demonstrate the superior performance of our proposed framework over some state-of-the-art approaches.

Keywords: service recommendation, collaborative filtering, motif-based graph neural networks, graph high-order connectivity, motif-based attention

*Corresponding author
Email addresses: wangguiling@ncut.edu.cn (Guiling Wang), jian.yu@aut.ac.nz (Jian Yu), thithuymo.nguyen@aut.ac.nz (Mo Nguyen), yuqi.zhang@autuni.ac.nz (Yuqi Zhang), sira.yongchareon@aut.ac.nz (Sira Yongchareon), hanyanbo@ncut.edu.cn (Yanbo Han)

Journal Pre-proof

Journal Pre-proof

PACS: 89.20.Ff

1. Introduction

With the rapid development of technologies such as cloud computing, edge computing, and Internet of Things, the number of web-accessible computing resources encapsulated as Web services and APIs (APIs in short) is continuously increasing [1]. To facilitate efficient exploration and selection of services, recommender systems, particularly Collaborative Filtering (CF) based methods and models, have been successfully applied to recommend suitable services to developers [2, 3, 4].

In recent years, Deep Neural Network (DNN) based models have outperformed traditional CF based method [5, 6, 2, 3, 4, 7, 8, 9]. However, most approaches only deal with direct invocations between mashups and APIs, disregarding high-order structures. The mashup-API invocation pairs are treated as separate data instances, creating so-called *information isolated island* and thus ignoring the inherent structure among invocations [10]. To address this limitation, recent DNN-based CF models such as [11, 12, 13] have leveraged high-order relations to tackle the cold-start problem of service recommendation for new mashups. Nevertheless, such approaches have not given attention to sub-graphs (or motifs) in the bipartite network.

By applying graph embedding to a graph that depicts the interactions between mashups and APIs, we can acquire service embeddings and deploy them to calculate the similarity between pairs of mashups and APIs. Subsequently, their invocation probability can be determined via this similarity measure. Graph embedding-based recommendation approaches exhibit better performance compared with traditional recommender systems. While traditional recommender systems learn the model parameters by analyzing the graph topological features such as users' co-interactions with frequently used items [14] or global topological diffusion [15, 16], graph embedding-based recommendation learns the embedding vectors and captures graph topological features by embedding techniques [17]. To do the analysis of graph topological features, some works utilize subgraphs [18], motifs [19, 20], and neighborhood to extract the features embedding and perform recommendations. Recent works on graph embedding for recommendation [21, 22, 23] have demonstrated the success of such approaches. Although several recent works consider high-order relation or structures for service recommen-

Journal Pre-proof

Journal Pre-proof

dation [24, 25, 26], motif-based feature embedding representation learning method has yet not been investigated in service recommendation.

In this paper, we first explore the common motifs in the Mashup-API bipartite graph and identify all the seven up-to-four-node motifs in an *oriented bipartite graph* (defined in Section 3). Furthermore, we propose a method to generate a motif adjacency matrix for each motif. Based on the generated motif adjacency matrices, we propose a motif-based graph attention mechanism that aggregates high-order information in the motifs. We then propose a Motif-based Graph Attention Network for service recommendation (MGSR).

This paper has the following main contributions:

1. We identify and define the various connectivity structures as motifs in oriented bipartite graphs and propose a dedicated algorithm to generate the corresponding motif adjacency matrices.
2. We propose a motif-based graph attention mechanism to attach the proposed motifs of connectivity structures.
3. We propose a Motif-based Graph Attention Network for Service Recommendation (MGSR) that aggregates high-order information of motif-based neighbours into the embedding progress for mashup-API recommendation. To the best of our knowledge, it is the first time that a motif-based graph attention network model is used in service recommendation.
4. We have conducted extensive empirical studies on the ProgrammableWeb dataset [1] and the results demonstrate the superior performance of MGSR over some state-of-the-art frameworks.

The remainder of this paper is organized as follows. Section 2 presents the related work; Section 3 describes definitions of motifs, the motif-based graph convolution self-attention method, and presents the MGSR model; Section 4 demonstrates the experiments running with comparison results and further analysis; and Section 5 concludes the paper.

[1] We downloaded the database from https://dev.maxmind.com/ in November 2020.

Journal Pre-proof

Journal Pre-proof

2. Related work

2.1. Graph representation learning

One of the commonly used data structures is graph, which is applied widely in many fields relating to computer science such as social media, biological protein-protein networks, molecular graph architecture, and their recommender systems. Graphs grasp the relations or edges among nodes. In [27], graphs are defined as the backbone of the countless systems that contain the accessible relational information of all interactions. Furthermore, graph takes a pivotal role in modern machine learning. There are many existing applications in different fields that leverage graph data to learn the embedding features for forecasting and recommendation purposes. These applications include predicting individuals' roles in collaborative networks, offering recommendations in social media, classifying protein roles in biological relationship graphs, and many more. In machine learning, how to embed the graph structure into a model is a hot research field. Conventional machine learning models use the graph statistical summary such as clustering coefficients [28], kernel functions [29] to extract the graph-based information, and embed the engineered features to calculate the neighbor's architecture [30]. However, such engineered features lack flexibility, specifically they might not adapt to the training and design processes, leading to a time-consuming and costly development process.

Therefore, a number of approaches that learn the representation through encoding the graph structural information have appeared [31, 32]. Such approaches use a mapping that embeds nodes, subgraphs, or the whole graph into a lower dimensional vector space. The objective is to optimize the mapping and reflect the original graph-based structure. After that, they use the embeddings as the input for downstream machine learning tasks. The main difference between the previous approaches and the representation learning approach lies in how graph structural information is learned. While previous approaches consider it as a reprocessing process that uses hand-engineered statistics to draw out the structural information, the representation method treats this problem as a machine learning task that applies a data-driven framework to encode the graph-based information.

2.2. Network motifs and high-order Graph Neural Networks

Network motifs with high-order connectivity are presented as fundamental building blocks of networks that exhibit complex structural patterns [33].

Journal Pre-proof

Journal Pre-proof

[34] explores motifs in biological networks and demonstrates that specific patterns of motifs correlate with the perturbation's robustness. On the other hand, [35] considers motifs in temporal networks and states that each type of motif presents different organization structures from different domains.

Existing works have studied the effectiveness of high-order connectivity with different graph-based machine learning models [36, 37, 38, 39, 40, 41]. DeepGL [42] learns the inductive relational functions using motifs. [38] studies the high-order network embeddings and approves the superior performance of several motif-based matrix formulas for generating these embeddings.

[39] proposes a hierarchical motif convolution for graph classification by identifying the task of sub-graphs. In addition, it designs a graph convolution framework for heterogeneous networks by leveraging connectivity based on motifs. [37] also demonstrates that GCN-based models and the one-dimension Weisfeiler-Lehman Isomorphism heuristic have similar deficiencies as mentioned above. Therefore, they propose a high-order framework for graph classification.

2.3. GNN and Motif-based Network in recommender systems

There are some existing works that utilize Graph Neural Network model to address the recommendation problem [43, 44, 45]. Similar to GraphSAGE [46] learning feature representations through sampling and aggregating strategies, PinSAGE [45] uses a random walk mechanism to learn node representations from chosen neighbor nodes and then integrate such high-order representations with GCN framework for Web-scale recommender systems. [44] leverages the relationships of users and items to learn their representations. Instead of such random walk-based methods, some works [47, 48, 49] have used motifs to capture the graph structural information. Particularly, a spectral motif convolution approach [47] is built for convolution filters. Motif-CNN [48] identifies several types of motifs to create the receptive fields for the target node and then performs motif-based spatial convolution operations to elicit the latent interaction features. Another work on graph node classification [49] presents a motif-level self-attention model to learn the weight of different motifs using differentiation. Luo [50] firstly present a Motif-based Neural Network applying for the Reciprocal Recommendation on Online Dating application. This work defines motifs and utilizes a random walk algorithm to sample neighbour users to learn the embedding

Journal Pre-proof

Journal Pre-proof

features. The prediction is based on fully connected neural network layers of a concatenation of their embedding feature vectors.

Our approach distinguishes itself from previous approaches [47, 48, 49, 50] in several significant points. In contrast to all the existing approaches, we identify all the motifs up to four nodes in an oriented bipartite graph. We also propose an algorithm to generate the motif adjacency matrices and a new motif-based graph attentional neural network designed for web service recommendation. Our proposed attention mechanism allows for the simultaneous learning of unique node weights across multiple motifs. Additionally, our approach differs from the existing works in that it specifically focuses on oriented bipartite networks.

3. The MGSR model for web service recommendation

In this section, we first present the preliminaries and problem statement. Then, we define the oriented bipartite graph, Mashup-API invocation graph, motif adjacency matrix, and motif-based neighbor. Next, we propose algorithms for generating motif adjacency matrices. We then define the motif-based graph convolution layers with self-attention to learn the mashup and API embeddings. Finally, we attach the motif-based graph attention network to a collaborative filtering-based recommendation model.

3.1. Definitions and problem statement

In this subsection, the relevant definitions and the problem statement are given. We then introduce seven motifs for service recommendation and provide a proof sketch showing that these seven motifs comprise a complete set of motifs with four nodes or less in a mashup-API invocation graph, which is a type of oriented bipartite graph. For brevity, the frequently used notations in this paper are summarized in Table 1.

Definition 1 (Bipartite graph[51])**.** A bipartite graph $G = (U, V, E)$ is a graph where all nodes in G are partitioned into two disjoint classes U and V, i.e. $U \cap V = \emptyset$, such that every edge has its terminal node in different classes from its initial node: nodes in the same partition class must not be adjacent.

Definition 2 (Directed graph[51])**.** A directed graph is an ordered pair $G = (V, E)$ where V is a set of nodes and E is a set of edges. Each edge is an ordered node pair of an initial node and a terminal node. An edge is said to be directed from the initial node to the terminal node.

Journal Pre-proof

Journal Pre-proof

Table 1: Notation

Symbol	Description
$G = (U, V, E)$	Mashup-API invocation graph (MAG)
$\mathcal{M} = \{M_t\}$	$\mathcal{T}$ motifs, $t = 1, ..., \mathcal{T}$
$\mathcal{A} = \{A_t\}$	$\mathcal{T}$ motif adjacency matrices, $t = 1, ..., \mathcal{T}$
$\mathcal{N}_i^{M_t}$	motif-based neighbors of node i for M_t
f_m, f_a	initial input features of mashup m and API a
h	projected node feature
$e_{ij}^{M_t}$	importance of node i's motif-based neighbor node j
$\alpha_{ij}^{M_t}$	weight coefficient of node i's motif-based neighbor node j
$h_i^{M_t}$	motif representation of h_i in motif-induced graph
$e_i^{M_t}$	importance of node i and its motif representation
$\beta_i^{M_t}$	weight coefficient of node i and its motif representation
$\mathbf{a}$	weight vector for motif neighbor and motif representation
h'	final embedding output features

Definition 3 (Oriented graph[51])**.** An oriented graph $G = (V, E)$ is a directed graph such that none of its pairs of nodes is linked by more than one edge and none of its edges has the same initial node and terminal node.

Next, we define oriented bipartite graphs and mashup-API invocation graphs used in this paper.

Definition 4 (Oriented bipartite graphs)**.** An oriented bipartite graph $G = (U, V, E)$ is a bipartite graph such that all edges' initial nodes are in U and all edges' terminal nodes are in V, and none of its edges has the same initial node and terminal node.

Definition 5 (Mashup-API invocation graph (MAG))**.** A Mashup-API Invocation Graph (MAG) is an oriented bipartite graph $G = (U, V, E)$, where U denotes the set of mashups, V denotes the set of APIs, and $E = U \times V$ with $e_{ij} \in E$ is an edge between U and V. If an API is invoked by a mashup then the edge e_{ij} exists and is assigned a value $r_{ij} \geq 0$, otherwise $r_{ij} = 0$. In MAG, each edge represents a reference from a mashup to an API, so it is an oriented bipartite graph.

Journal Pre-proof

Journal Pre-proof

Based on the definition of MAG, we formulate our web service recommendation problem as follows:

Problem Statement: Let a MAG $G = (U, V, E)$ be a set of mashups (U), a set of APIs (V), and their historical invocation relationships (E), suppose we have a target mashup u_t and a set of APIs V_{u_t} which u_t has invoked, the goal is to recommend the top k APIs from $V \setminus V_{u_t}$ that should be most likely to be invoked by u_t.

As introduced in Section 2, motifs are fundamental building blocks of networks with high-order connectivity. For the rest of this paper, we use "motif" as shorthand for oriented bipartite graph motif. To exploit the structural information in motifs, motif adjacency matrices are proposed as in [52]:

Definition 6. A motif-induced graph's adjacency matrix A, also known as the motif adjacency matrix, is defined by its entries (i, j) which represent the number of instances that nodes i and j co-exist within the motif. According to Proposition 1, in the MAG, we define a set of $\mathcal{T}$ motifs $\mathcal{M} = \{M_1, ..., M_t, ..., M_\mathcal{T}\}$ (where $\mathcal{T} = 7$), and then we build a set of $\mathcal{T}$ different motif adjacency matrices $\mathcal{A} = \{A_1, ..., A_t, ..., A_\mathcal{T}\}$ where A_t is a motif adjacency matrix corresponding to M_t. The entry of A_t is defined as $(A_t)_{m,a}$, the number of motif instances of type M_t that contains both mashup m and API a.

Definition 7. Given a node i and a motif M_t, the motif-based neighbors $\mathcal{N}_i^{M_t}$ of node i are defined as the set of nodes which connect with node i in motif M_t.

To balance the importance of local structure and higher-order neighborhood connection, we limit our considerations of motifs to those consisting of no more than four nodes. In a MAG, this results in the following proposition regarding the set of motifs:

Proposition 1. *Provided an oriented bipartite mashup-API invocation graph MAG $G = (U, V, E)$, there are seven motifs in total, each containing up to four nodes, and the entire set of motifs is $\{M_1, M_2, M_3, M_4, M_5, M_6, M_7\}$. The seven motifs are shown in Figure 1, where the circles indicate the mashup node and the squares indicate the API node.*

Journal Pre-proof

Journal Pre-proof

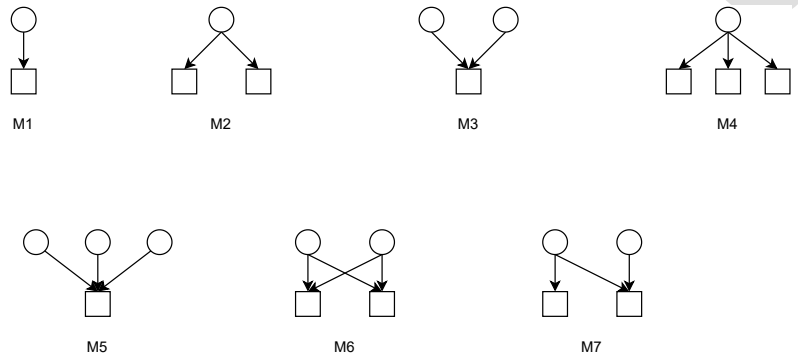


Figure 1: Motifs of two to four nodes in oriented bipartite MAG

Sketch of Proof. We define n_u, n_v as the number of mashups from U, the number of APIs from V and calculate the maximum number of possible edges as $e_{max} = n_u \times n_v$ in which these mashups/APIs build an entire bipartite graph. We calculate the minimum number of edges for a connected bipartite graph as $e_{min} = n_u + n_v - 1$. To build a motif with these mashups/APIs, we can select any n_e edges from the e_{max} edges given $n_e \geq e_{min}$.

Specifically, we enumerate all types of motifs as follows:

Motif type M_1**:** $n_u = 1, n_v = 1$, $e_{max} = e_{min} = 1$, so there are $\binom{1}{1} = 1$ possible motifs.

Motif type M_2**:** $n_u = 1, n_v = 2$, $e_{max} = e_{min} = 2$, so there are $\binom{2}{2} = 1$ possible motifs.

Motif type M_3**:** $n_u = 2, n_v = 1$, $e_{max} = e_{min} = 2$, so there are $\binom{2}{2} = 1$ possible motifs.

Motif type M_4**:** $n_u = 1, n_v = 3$, $e_{max} = e_{min} = 3$, so there are $\binom{3}{3} = 1$ possible motifs.

Motif type M_5**:** $n_u = 3, n_v = 1$, $e_{max} = e_{min} = 3$, so there are $\binom{3}{3} = 1$ possible motifs.

Motif type M_6**:** $n_u = 2, n_v = 2$, $e_{max} = 4, e_{min} = 3, n_e = 4$, so there are $\binom{4}{4} = 1$ possible motifs in case $n_e = 4$.

Motif type M_7**:** $n_u = 2, n_v = 2$, $e_{max} = 4, e_{min} = 3, n_e = 3$, there are $\binom{4}{3} = 4$ possible motifs in case $n_e = 3$. From Figure 1 we can observe that the four possible motifs are isomorphic, hence we only have one motif, denoted

Journal Pre-proof

as M_7, for $n_e = 3$. □

3.2. Motif adjacency matrices for MAG

Based on Definition 6 and 7, we generate motif adjacency matrices $\mathcal{A}$ by finding all motif instances for a motif M_t. For M_1, the motif adjacency matrix is simply the adjacency matrix of the graph. For the other motifs, the process of generating motif adjacency matrices is about finding the motif instances in the graph. M_2 to M_5 are all tree-structured motifs, i.e. they are composed of a single node of one type and multiple nodes of the other. Finding motif instances for them is similar. Both M_2 and M_4 comprise a single mashup node and multiple API nodes, so we need to choose multiple API nodes from each mashup node's neighbor set to form an instance together with the mashup node. Similarly, M_3 and M_5 comprise multiple mashup nodes and a single API node, so we need to choose multiple mashup nodes from the API node's neighbor set.

Finding motif instances for M_6 and M_7 is more complex. For M_6, we choose two API nodes from V and find the intersection of their neighbors, i.e. a set of mashup nodes. Any two mashup nodes in this set and the two chosen API nodes form a motif instance of M_6. For M_7, we choose two API nodes from V and then find the intersection and symmetric difference between the two API nodes' neighbor sets. A motif instance of M_7 is formed by two API nodes, one node from the intersection of mashup nodes, and one node from the symmetric difference of mashup nodes.

For motif M_1, the motif adjacency matrix is the same as the adjacency matrix of the graph so the algorithm is omitted here. For M_2 to M_7, their motif adjacency matrix generation algorithms are very similar so we only show the algorithm for M_2. The intuition behind the algorithms for M_2 to M_5 is that splitting all edges into groups by the type of nodes which is the single node in the motif (e.g. mashup node in M_2), then updating the motif adjacency matrix for each combination of the elements in the group (for M_2, the combination consists of one mashup node and two API nodes).

Algorithm 1 describes the motif adjacency matrix generation process for motif M_2 in which all edges E, the number of nodes N in the graph, and the graph's adjacency matrix A are provided as input. The algorithm begins with splitting all edges into groups. In each group Q_i, all edges source from the same mashup node i. We then generate a vector where the elements are the node degrees minus one, i.e. $deg(i) - 1$. After that, we assign zero to the counts for API nodes in d because only degrees for mashup nodes are used to

Journal Pre-proof

Journal Pre-proof

Algorithm 1: Motif M_2 adjacency matrix generation algorithm

Input: Edges E; number of nodes N; number of mashup nodes N_U; adjacency matrix A; neighbourhood function $\mathcal{N}$

Output: Motif adjacency matrix M

$d \leftarrow \{d_1, ..., d_N\}$,where $d_i = deg(i) - 1$ for $i \in U$, $d_j = 0$ for $j \in V$;
$D \leftarrow N \times N$ diagonal matrix built from d, where $D_{ii} = d_i$;
$M \leftarrow D \cdot A$;
for $i \in U$ **do**
  if $|\mathcal{N}(i)| > 1$ **then**
    for $(j,k) \in \binom{|\mathcal{N}(i)|}{2}$ *pairs of APIs* **do**
      Add one to entry (j,k) in M;
    end
  end
end
$M \leftarrow M + M^T$;

update the motif adjacency matrix. Next, we generate an upper triangular matrix by multiplying the adjacency matrix A with a diagonal matrix built from d. In each step of the outer loop, we first check whether the number of edges in group Q_i is greater than one since there have to be at least two API nodes plus one mashup node to form an instance of M_2. If so, it enters the inner loop which adds one to the counts between every pair of API nodes because any two API nodes can only coexist in one motif instance of M_2. At the end of this algorithm, we add matrix M's transpose matrix to itself to form a symmetric matrix.

3.3. Motif-based graph convolution layers with self-attention

Inspired by GAT [53], we design the motif-based graph attentional layer for the MGSR model. The attention mechanism learns the weights of different nodes from various motifs simultaneously. Moreover, the attention network utilized to calculate the attention scores between nodes in a motif-induced graph and between nodes in different motif-induced graphs is identical, as shown in Figure 2.

Firstly, we use the features of a set U of mashups, $F_m = \{f_m | \, m \in U\}$, and a set V of APIs, $F_a = \{f_a | \, a \in V\}$, as the input layer. Mashups and APIs have different feature spaces, $f_m \in \mathcal{R}^{D_m}$, $f_a \in \mathcal{R}^{D_a}$. From the input

Journal Pre-proof

features' vectors, the graph attentional layer projects them into the same feature space for each mashup/API. We denote the set of projected features' vectors as $H_m = \{h_m | \, m \in U\}$ and $H_a = \{h_a | \, a \in V\}$, where $h_m \in \mathcal{R}^{D'}$, $h_a \in \mathcal{R}^{D'}$. D_m, D_a, D' are the dimensions of input features of mashups and APIs and projected features. To this end, in the initial step, we update the mashups and APIs embedding features by a linear transformation weighted by shared parameter matrix $\mathbf{W_m} \in \mathcal{R}^{D_m \times D'}$ and $\mathbf{W_a} \in \mathcal{R}^{D_a \times D'}$ applied for all mashups and APIs respectively.

$$h_i = \begin{cases} \mathbf{W_m} \cdot f_i, & i \in U \\ \mathbf{W_a} \cdot f_i, & i \in V \end{cases} \tag{1}$$

where f_i and h_i are the original and projected feature of node i, and $h_i \in \mathcal{R}^{D'}$.

An attention network $\mathbf{att} : \mathcal{R}^{D'} \times \mathcal{R}^{D'} \to \mathcal{R}$ is shared on different nodes in the same motif-induced graph. **att** is performed to compute attention coefficients between a node i and i's motif-based neighbour:

$$e_{ij}^{M_t} = \mathbf{att}(h_i, h_j) \tag{2}$$

The reason behind sharing **att** is two-fold: on the one hand, the number of parameters can be reduced; on the other hand, those basic features that are not related to node position both in the graph and in the motif-induced graph can be learned in this way. We only compute the attention weight $e_{ij}^{M_t}$ for nodes $j \in \mathcal{N}_i^{M_t}$, where $\mathcal{N}_i^{M_t}$ is node i's neighbour nodes in the motif-induced graph for M_t. The attention score for the importance of node j to node i should be normalized across all choices of j using the softmax function:

$$\alpha_{ij}^{M_t} = \text{softmax}_j(e_{ij}^{M_t}) = \frac{\exp(e_{ij}^{M_t})}{\sum_{k \in \mathcal{N}_i^{M_t}} \exp(e_{ik}^{M_t})} \tag{3}$$

We adopt a single-layer feedforward neural network parameterized by a weight vector $\mathbf{a} \in \mathcal{R}^{2D'}$ with the LeakyReLU activation function for the attention network **att**. Different from GAT, $\mathbf{a}$ is shared among different motif-induced graphs. Then the attention score of $\alpha_{ij}^{M_t}$ corresponding to M_t's motif-induced graph is expressed as:

$$\alpha_{ij}^{M_t} = \text{softmax}_j(e_{ij}^{M_t}) = \frac{\exp(\text{LeakyReLU}(\mathbf{a^T}[h_i \parallel h_j]))}{\sum_{k \in \mathcal{N}_i^{M_t}} \exp(\text{LeakyReLU}(\mathbf{a^T}[h_i \parallel h_k]))} \tag{4}$$

Journal Pre-proof

where $\cdot^T$ and $\|$ are the transposition and concatenation operation respectively.

With $\alpha_{ij}^{M_t}$ calculated, the hidden embedding representation of h_i in motif-induced graph for M_t is:

$$h_i^{M_t} = \sum_{j \in \mathcal{N}_i^{M_t}} \alpha_{ij}^{M_t} h_i \tag{5}$$

To represent the importance between nodes in different motifs, we compute the attention coefficients between a node i and $h_i^{M_t}$, which is i's hidden embedding representation in motif-induced graph. When computing the attention score of the above, the attention network **att** used is the same as the attention network used to compute the attention scores between different nodes in the same motif-induced graph:

$$e_i^{M_t} = \mathbf{att}(h_i, h_i^{M_t}) \tag{6}$$

The attention score for the importance of i's hidden embedding representation in M_t's motif-induced graph is normalized across all choices of M_t using the softmax function:

$$\beta_i^{M_t} = \text{softmax}_{M_j}(e_i^{M_t}) = \frac{\exp(e_i^{M_t})}{\sum_{j=1}^{\mathcal{T}} \exp(e_i^{M_j})} \tag{7}$$

Again a single-layer feedforward neural network parameterized by a weight vector $\mathbf{a} \in \mathcal{R}^{2D'}$ with the LeakyReLU activation function for **att** is adopted:

$$\beta_i^{M_t} = \text{softmax}_{M_j}(e_i^{M_t}) = \frac{\exp(\text{LeakyReLU}(\mathbf{a}^T[h_i \parallel h_i^{M_t}])}{\sum_{j=1}^{\mathcal{T}} \exp(\text{LeakyReLU}(\mathbf{a}^T[h_i \parallel h_i^{M_j}]))} \tag{8}$$

Then the final output features for each node is obtained by applying a nonlinearity σ (in our experiment, the ELU activation function is adopted):

$$h_i' = \sigma(\sum_{j=1}^{\mathcal{T}} \beta_i^{M_j} h_i^{M_t}) \tag{9}$$

In order to stabilize the learning process, the multi-head attention mechanism is performed by executing K independent attention networks and transformation of Equation 9. Then the output features of the non-final layers are

Journal Pre-proof

Journal Pre-proof

concatenated and averaged for the final layer:

$$h'_i = \Big\Vert_{k=1}^{K} \sigma(\sum_{j=1}^{\mathcal{T}} \beta_i^{M_j(k)} h_i^{M_t}) \tag{10}$$

$$h'_i = \sigma(\frac{1}{K} \sum_{k=1}^{K} \sum_{j=1}^{\mathcal{T}} \beta_i^{M_j(k)} h_i^{M_t}) \tag{11}$$

The combination process of such graph self-attention layer is demonstrated in Figure 2. This process is used for both mashups and APIs. For instance, Figure 2 shows the mashup h_i and its motif-based neighbors. It is worth noting that h_i may have different neighbors in motif M_1 and motif M_7. In Figure 2, $\alpha_{i,j}^{M_1}$ and $\alpha_{i,j}^{M_7}$ are the node-level attention weights between h_i and its motif-based neighbors connected in motif M_1 and M_7 according to motif adjacency matrix A_1 and A_7. $h_i^{M_1}$ and $h_i^{M_7}$ are the motif-based hidden features for M_1 and M_7's motif-induced graph. $\beta_i^{M_1}$ and $\beta_i^{M_7}$ are the attention score of the weight for the hidden features of node i in M_1 and M_7's motif-induced graph respectively. The output h'_i is generated by calculating the weighted sum of $h_i^{M_t}$, where t is from 1 to $\mathcal{T}$.

As a result, the output sets of features H'_m and H'_a are obtained from the motif-based graph self-attention layers and contain the high-order graph relation, which can be used as auxiliary information for predictive model in the next subsection.

3.4. Motif-based graph attention collaborative filtering for service recommendation (MGSR)

In this section, we aim to project the invocation between the pair of mashup m and API a provided their latent representation h'_m and h'_a. The prediction value R will be obtained by the dot product of h'_m and h'_a learning through a stochastic gradient descent method.

We use the original invocation relationship between mashup and API as the actual score and the dot product of h'_m and h'_a as the estimated score. The quadratic loss function is used to continuously optimize h'_m and h'_a:

$$Loss = \sum_{m \in U, a \in V} \left(r_{ma} - h_m'^T h'_a\right)^2 + \lambda(\|h'_m\|^2 + \|h'_a\|^2) \tag{12}$$

Journal Pre-proof

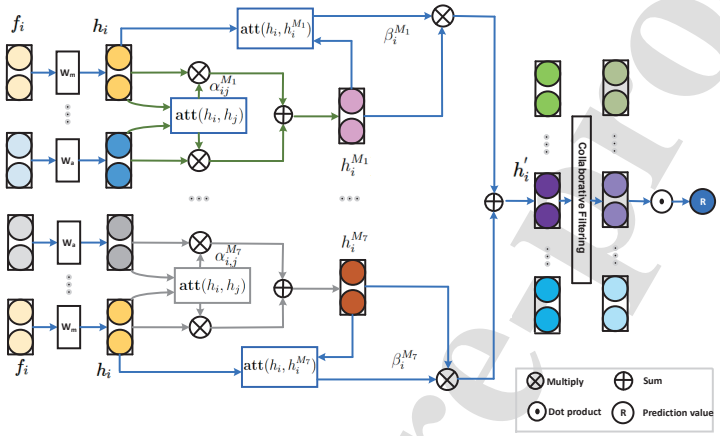


Figure 2: Motif-based graph attention mechanism

where r_{ma} is the original invocation relationship between mashup and API, λ is the L2 regularization coefficient.

Then h'_m and h'_a is normalized and dot-producted to calculate the prediction value between m and a:

$$R = h'_m \odot h'_a \tag{13}$$

The value of $R \in [0, 1]$ indicates the likelihood of invocation between the mashup and API.

4. Experimental results

We use the ProgrammableWeb dataset to evaluate our proposed model. Our objective is to answer the following research questions:

- RQ1: How much does MGSR outperform the state-of-the-art graph-based CF models?
- RQ2: How do different types of motifs influence the performance of the proposed model?

Journal Pre-proof

4.1. Datasets and baselines

Datasets. In Web API application, we use the dataset of ProgrammableWeb which can be presented as a bipartite network to evaluate the performance of the proposed model. The dataset consists of 17829 APIs and 6340 mashups, and their invocation data. Specifically, if mashup m_1 invokes API a_5, then their invocation link/edge has weight 1. We use the following steps for data reprocessing: (a) removing all blank APIs and mashups resulted in obtaining 5691 mashups and 1170 APIs; (b) using a portion of the mashup-API invocations as the train set and the validation set, and using the rest as the test set.

Baselines We compare our proposed MGSR with some state-of-the-art approaches:

- AMF [54]: an attentional Matrix factorization with document context and API co-invocation.
- NGCF [55]: a GCN-based collaborative filtering method that propagates embeddings on the MAG to exploit the high-order connectivity signal of mashups and users.
- HACF [56]: a High-order Data Augmentation Collaborative Filtering method for service recommendation.
- GAT-CF [53]: a graph attention network based collaborative filtering model.
- MISR [11]: a multiplex invocation-oriented service recommendation model which leverages three types of invocations between services and mashups and incorporates them into a DNN that can identify both their explicit and implicit relationships.

4.2. Settings

Evaluation metrics. We use hit ratio (HR) and Normalized Discounted Cumulative Gain (NDCG) to evaluate the performance of API recommendation. We select APIs that have not been utilized previously for each mashup in the training dataset. After that, we sort them in ascending order according to their estimated ratings and truncate the top-k ranked for the recommendation list. The HR is calculated from the top-k recommendation list by using the equation:

Journal Pre-proof

Journal Pre-proof

Table 2: Baseline comparison over different k on HR metric

k	2	3	4	5	6	7	8	9	10
AMF	0.4021	0.4152	0.4182	0.4262	0.4352	0.4521	0.4921	0.5201	0.5583
NGCF	0.4372	0.4552	0.5037	0.5537	0.6012	0.6252	0.6317	0.6677	0.6882
MISR	0.6494	0.6760	0.7023	0.7211	0.7286	0.7392	0.7392	0.7407	0.7441
HACF	**0.6645**	**0.7190**	0.7367	0.7483	0.7547	0.7603	0.7604	0.7606	0.7616
GAT-CF	0.5976	0.6951	0.7561	**0.7805**	**0.8171**	**0.8293**	0.8293	0.8293	0.8293
MGSR17	0.5976	0.6707	**0.7561**	**0.7805**	**0.8171**	**0.8293**	**0.8415**	**0.8415**	**0.8415**

Table 3: Baseline comparison over different k on NDCG metric

k	2	3	4	5	6	7	8	9	10
AMF	0.3321	0.3498	0.3694	0.3972	0.3999	0.4093	0.4109	0.4271	0.4306
NGCF	0.3521	0.3841	0.4393	0.4647	0.4678	0.4580	0.4603	0.4613	0.4647
MISR	0.5209	0.5050	0.5343	0.5301	0.5351	0.5433	0.5569	0.5568	0.5583
HACF	**0.5626**	0.5666	0.5627	0.5657	0.5711	0.5716	0.5718	0.5725	0.5726
GAT-CF	0.5255	**0.5718**	0.602	**0.6108**	**0.6239**	**0.6272**	0.6272	0.6197	0.6197
MGSR17	0.5255	0.5604	**0.6009**	0.6095	0.6229	0.6263	**0.6319**	**0.6319**	**0.6264**

$$HR@k = \frac{Numberofhits@k}{N_r} \tag{14}$$

Where N_r is the number of recommended APIs. We then measure the effectiveness of recommendation by using the Discounted Cumulative Gain metric and determining the ranking of relevant APIs in the prediction list. Specifically, the NDCG from rank 1 to k for the recommendation list is obtained by the following equation:

$$NDCG@k = \sum_{i=1}^{k} \frac{2^{r_i} - 1}{\log_2(i+1)} \tag{15}$$

Hyper-parameters. All datasets are split into a train set, validation set, and test set with ratios 60%, 20%, and 20% respectively. We use the train set to train the proposed MGSR model and evaluate the HR and NDCG on a test set. For the evaluation process, all the mashups in the test set have a list of invoked APIs. For each mashup, all APIs in the test set are scored with their estimated invocation value which is between 0 and 1. After that, we sort them in ascending order according to their estimated ratings and truncate the top-k for the recommendation list. We use $k = 10$ for our experiments.

We implement MGSR based on Pytorch using Graph-tools to develop the model. In order to ensure MGSR achieves the best performance, we use

Journal Pre-proof

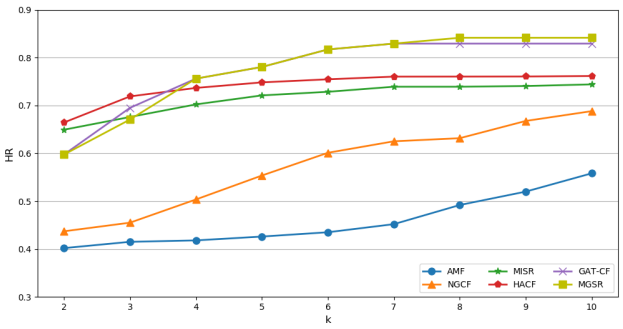


Figure 3: HR results of baselines and MGSR17.

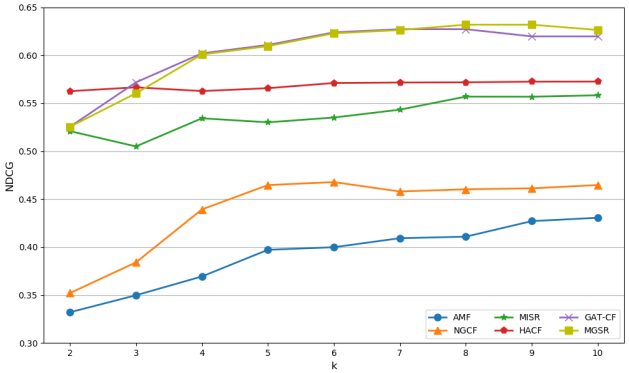


Figure 4: NDCG results of baselines and MGSR17.

grid search to tune the hyperparameters. As a result, the suitable values of hyperparameters obtained are learning rate, convergence epoch, hidden size, drop out which are set at 0.005, 1000, 8, and 0.6 respectively.

Journal Pre-proof

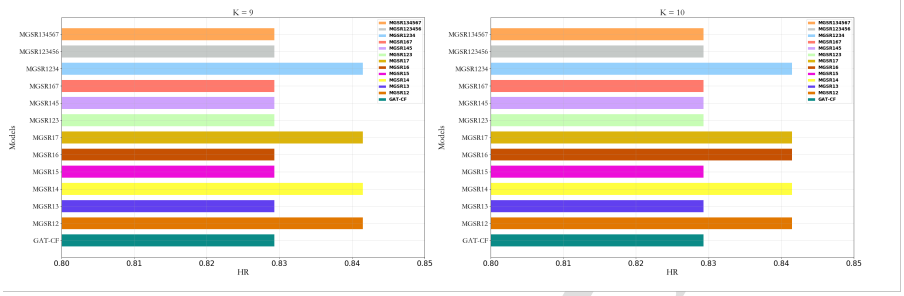


Figure 5: HR results of MGSR variants

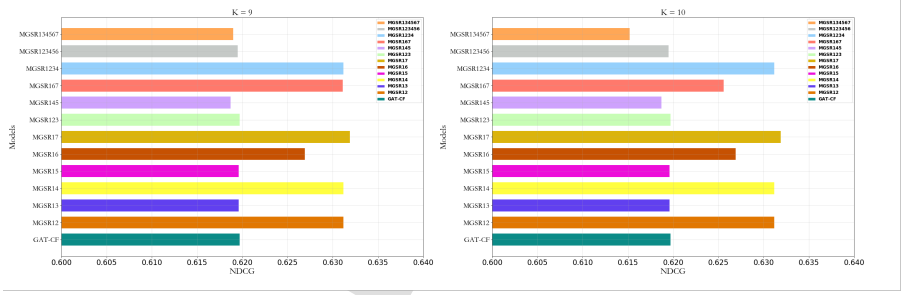


Figure 6: NDCG results of MGSR variants.

4.3. Comparison results

For all the baselines and MGSR variants, we run the models until the values of loss converge and obtain the best results of HR and NDCG. We calculate HR and NDCG at k from 2 to 10. We perform 10-fold cross-validation, use 9 folds for training and 1 for testing, and repeat the experiments 10 times to report the average HR and NDCG.

Table 2 and Table 3 show the performance details of the baselines and MGSR17, which obtains the best result among MGSR variants (see Subsection 4.4).

Overall, the results show that our proposed model MGSR17 obtains superior performance against existing works including AMF, NGCF, and MISR

Journal Pre-proof

Table 4: HR results of variants of MGSR

Variant	2	3	4	5	6	7	8	9	10
GAT-CF	0.5976	0.6951	0.7561	0.7805	0.8171	0.8293	0.8293	0.8293	0.8293
MGSR12	0.5976	0.6829	0.7561	0.7805	0.8171	0.8293	0.8293	0.8415	0.8415
MGSR13	0.5976	0.6829	0.7561	0.7805	0.8171	0.8293	0.8293	0.8293	0.8293
MGSR14	0.5976	0.6829	0.7561	0.7805	0.8171	0.8293	0.8293	0.8415	0.8415
MGSR15	0.5976	0.6829	0.7561	0.7805	0.8171	0.8293	0.8293	0.8293	0.8293
MGSR16	0.5976	0.6829	0.7561	0.7805	0.8171	0.8293	0.8293	0.8293	0.8415
MGSR17	0.5976	0.6707	0.7561	0.7805	0.8171	0.8293	0.8415	0.8415	0.8415
MGSR123	0.5976	0.6951	0.7561	0.7805	0.8171	0.8293	0.8293	0.8293	0.8293
MGSR145	0.5976	0.6707	0.7561	0.7805	0.8171	0.8293	0.8293	0.8293	0.8293
MGSR167	0.5976	0.6829	0.7561	0.7805	0.8171	0.8293	0.8293	0.8293	0.8293
MGSR1234	0.5976	0.6829	0.7561	0.7805	0.8171	0.8293	0.8293	0.8415	0.8415
MGSR123456	0.5976	0.6829	0.7561	0.7805	0.8171	0.8293	0.8293	0.8293	0.8293
MGSR134567	0.5976	0.6707	0.7561	0.7805	0.8171	0.8293	0.8293	0.8293	0.8293

Table 5: NDCG results of variants of MGSR

Variant	2	3	4	5	6	7	8	9	10
GAT-CF	0.5255	0.5718	0.602	0.6108	0.6239	0.6272	0.6272	0.6197	0.6197
MGSR12	0.5255	0.5665	0.6017	0.6103	0.6238	0.6271	0.6271	0.6312	0.6312
MGSR13	0.5255	0.5665	0.6017	0.6103	0.6238	0.6271	0.6271	0.6196	0.6196
MGSR14	0.5255	0.5665	0.6017	0.6103	0.6238	0.6271	0.6271	0.6312	0.6312
MGSR15	0.5255	0.5665	0.6017	0.6103	0.6238	0.6271	0.6271	0.6196	0.6196
MGSR16	0.5255	0.5665	0.6017	0.6103	0.6238	0.6271	0.6271	0.6269	0.6269
MGSR17	0.5255	0.5604	0.6009	0.6095	0.6229	0.6263	0.6319	0.6319	0.6319
MGSR123	0.5255	0.5718	0.602	0.6108	0.6239	0.6272	0.6272	0.6197	0.6197
MGSR145	0.5255	0.5604	0.6009	0.6095	0.6229	0.6263	0.6263	0.6187	0.6187
MGSR167	0.5255	0.5665	0.6017	0.6103	0.6238	0.6271	0.6271	0.6311	0.6256
MGSR1234	0.5255	0.5665	0.6017	0.6103	0.6238	0.6271	0.6271	0.6312	0.6312
MGSR123456	0.5255	0.5657	0.6012	0.6099	0.6234	0.6267	0.6267	0.6195	0.6195
MGSR134567	0.5255	0.5604	0.6009	0.6095	0.6229	0.6263	0.6263	0.619	0.6152

which do not consider the high-order connectivity of the MAG. For later approaches such as HACF and GAT-CF that attach mashup-API high-order relations, MGSR does not outperform in all metrics but achieves the best HR and NDCG in most of larger values of k, particularly with k from 5 to 10. Figure 3 and Figure 4 provide better visualization for such comparison. From the figures, we can also see that the group of models that use high-order connectivity performs much better than the ones that do not use it, which demonstrates the benefit of attaching high-order information to predictive models.

4.4. The influence of different types of motifs on the MGSR's performance

In this subsection, we study the performance of each type of motif. Table 4 and Table 5 show the details of HR and NDCG scores when applying

Journal Pre-proof

Journal Pre-proof

different types of motifs on the MGSR framework. Three interesting observations can be made below:

- First, from the results, we can conclude that different combinations of motifs could achieve different model performances.
- Second, the HR and NDCG scores are quite similar for small values of k, while they are more distinct when $k > 5$. Particularly, for $k = 2$, there is no difference among the models. All of the models obtain 0.5976 for HR and 0.5255 for NDCG. At $k = 3$, we can see a significant increase in the performance of MGSR variants compared with GAT-CF. As can be seen in Figure 5, the HR value of variants that contain motifs M1, M3, and M5 has smaller numbers. This fact is also true for NDCG in Figure 6.
- Lastly, we notice that MGSR17 outperforms the other combinations of motifs.

5. Conclusion and future work

Web services have gained momentum at a fast pace in the past decade with a great number of APIs published on the Internet. These APIs have become widely used in web and mobile applications. However, this booming growth brought difficulties to select proper APIs for building mashups. A great number of research works have been proposed for service recommendation while the intrinsic relation in a graph structure has not been fully considered.

Our work focuses on exploiting motifs in the mashup-API bipartite network and the influence of graph structures on the embedding features of services. The proposed MGSR model uses the motif-based graph attention neural network architecture and attaches it to the CF-based model. The attention network used for calculating the attention scores between the nodes in a motif-induced graph and between nodes in different motif-induced graphs is the same. The experimental results show that MGSR outperforms some state-of-the-art service recommendation models. In the Future, we will explore whether more complex types of motifs and additional information (e.g., spatial and temporal information) can bring better performance to our model.

Journal Pre-proof

Journal Pre-proof

Acknowledgements

This work is supported by Projects of International Cooperation and Exchanges NSFC (Grant No. 62061136006) and the Key Program of National Natural Science Foundation of China (Grant No. 61832004).

References

[1] O. Adeleye, J. Yu, G. Wang, S. Yongchareon, Constructing and evaluating evolving web-api networks-a complex network perspective, IEEE Transactions on Services Computing (2021).

[2] X. Chen, X. Liu, Z. Huang, H. Sun, Regionknn: A scalable hybrid collaborative filtering algorithm for personalized web service recommendation, in: 2010 IEEE international conference on web services, IEEE, 2010, pp. 9–16.

[3] Y. Hu, Q. Peng, X. Hu, R. Yang, Time aware and data sparsity tolerant web service recommendation based on improved collaborative filtering, IEEE Transactions on Services Computing 8 (5) (2014) 782–794.

[4] F. Xie, J. Wang, R. Xiong, N. Zhang, Y. Ma, K. He, An integrated service recommendation approach for service-based system development, Expert Systems With Applications 123 (2019) 178–194.

[5] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th international conference on World Wide Web, 2001, pp. 285–295.

[6] H. Zhang, F. Shen, W. Liu, X. He, H. Luan, T.-S. Chua, Discrete collaborative filtering, in: Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 325–334.

[7] T. Liang, L. Chen, J. Wu, H. Dong, A. Bouguettaya, Meta-path based service recommendation in heterogeneous information networks, in: International Conference on Service-Oriented Computing, Springer, 2016, pp. 371–386.

Journal Pre-proof

Journal Pre-proof

[8] F. Xie, L. Chen, D. Lin, Z. Zheng, X. Lin, Personalized service recommendation with mashup group preference in heterogeneous information network, IEEE Access 7 (2019) 16155–16167.

[9] X. Song, J. Li, T. Cai, S. Yang, T. Yang, C. Liu, A survey on deep learning based knowledge tracing, Knowledge-Based Systems 258 (2022) 110036.

[10] X. Wang, X. He, T.-S. Chua, Learning and reasoning on graph for recommendation, in: Proceedings of the 13th International Conference on Web Search and Data Mining, 2020, pp. 890–893.

[11] Y. Ma, X. Geng, J. Wang, A deep neural network with multiplex interactions for cold-start service recommendation, IEEE Transactions on Engineering Management 68 (1) (2020) 105–119.

[12] L. Yao, X. Wang, Q. Z. Sheng, B. Benatallah, C. Huang, Mashup recommendation by regularizing matrix factorization with api co-invocations, IEEE Transactions on Services Computing 14 (02) (2021) 502–515.

[13] G. Chen, L. Chen, Augmenting service recommender systems by incorporating contextual opinions from user reviews, User Modeling and User-Adapted Interaction 25 (08 2015).

[14] B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in: Proceedings of the 10th international conference on World Wide Web, 2001, pp. 285–295.

[15] Y.-C. Zhang, M. Medo, J. Ren, T. Zhou, T. Li, F. Yang, Recommendation model based on opinion diffusion, EPL (Europhysics Letters) 80 (6) (2007) 68003.

[16] Y.-C. Zhang, M. Blattner, Y.-K. Yu, Heat conduction process on community networks as a recommendation model, Physical review letters 99 (15) (2007) 154301.

[17] P. Goyal, E. Ferrara, Graph embedding techniques, applications, and performance: A survey, Knowledge-Based Systems 151 (2018) 78–94.

[18] J. Zhao, X. Wang, C. Shi, B. Hu, G. Song, Y. Ye, Heterogeneous graph structure learning for graph neural networks, in: 35th AAAI Conference on Artificial Intelligence (AAAI), 2021.

Journal Pre-proof

Journal Pre-proof

[19] S. Lim, J.-G. Lee, Motif-based embedding for graph clustering, Journal of Statistical Mechanics: Theory and Experiment 2016 (12) (2016) 123401.

[20] S. Zhang, Z. Hu, A. Subramonian, Y. Sun, Motif-driven contrastive learning of graph representations, arXiv preprint arXiv:2012.12533 (2020).

[21] Z. Sun, Q. Guo, J. Yang, H. Fang, G. Guo, J. Zhang, R. Burke, Research commentary on recommendations with side information: A survey and research directions, Electronic Commerce Research and Applications 37 (2019) 100879.

[22] Y. Gao, Y.-F. Li, Y. Lin, H. Gao, L. Khan, Deep learning on knowledge graph for recommender system: A survey, arXiv preprint arXiv:2004.00387 (2020).

[23] Q. Guo, F. Zhuang, C. Qin, H. Zhu, X. Xie, H. Xiong, Q. He, A survey on knowledge graph-based recommender systems, IEEE Transactions on Knowledge and Data Engineering (2020).

[24] C. Wei, Y. Fan, J. Zhang, H. Lin, A-hsg: Neural attentive service recommendation based on high-order social graph, in: 2020 IEEE International Conference on Web Services (ICWS), 2020, pp. 338–346.

[25] C. Wei, Y. Fan, J. Zhang, High-order social graph neural network for service recommendation, IEEE Transactions on Network and Service Management (2022) 1–1.

[26] M. Luo, P. Chen, T. Sun, Y. Xia, N. Jiang, X. Wang, W. Wei, A novel high-order cluster-gcn-based approach for service recommendation, in: C. Xu, Y. Xia, Y. Zhang, L.-J. Zhang (Eds.), Web Services – ICWS 2021, Springer International Publishing, Cham, 2022, pp. 32–45.

[27] R. Angles, C. Gutierrez, Survey of graph database models, ACM Computing Surveys (CSUR) 40 (1) (2008) 1–39.

[28] S. Bhagat, G. Cormode, S. Muthukrishnan, Node classification in social networks, in: Social network data analytics, Springer, 2011, pp. 115–148.

Journal Pre-proof

Journal Pre-proof

[29] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, K. M. Borgwardt, Graph kernels, Journal of Machine Learning Research 11 (2010) 1201–1242.

[30] D. Liben-Nowell, J. Kleinberg, The link-prediction problem for social networks, Journal of the American society for information science and technology 58 (7) (2007) 1019–1031.

[31] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, S. Y. Philip, A comprehensive survey on graph neural networks, IEEE transactions on neural networks and learning systems 32 (1) (2020) 4–24.

[32] U. Fang, J. Li, N. Akhtar, M. Li, Y. Jia, GoMIC: Multi-view image clustering via self-supervised contrastive heterogeneous graph co-learning, World Wide Web (Oct. 2022).

[33] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, U. Alon, Network motifs: simple building blocks of complex networks, Science 298 (5594) (2002) 824–827.

[34] R. J. Prill, P. A. Iglesias, A. Levchenko, Dynamic properties of network motifs contribute to biological network organization, PLoS biology 3 (11) (2005) e343.

[35] A. Paranjape, A. R. Benson, J. Leskovec, Motifs in temporal networks, in: Proceedings of the tenth ACM international conference on web search and data mining, 2017, pp. 601–610.

[36] N. Ahmed, R. A. Rossi, J. Lee, T. Willke, R. Zhou, X. Kong, H. Eldardiry, Role-based graph embeddings, IEEE Transactions on Knowledge and Data Engineering (2020).

[37] C. Morris, M. Ritzert, M. Fey, W. L. Hamilton, J. E. Lenssen, G. Rattan, M. Grohe, Weisfeiler and leman go neural: Higher-order graph neural networks, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 4602–4609.

[38] R. A. Rossi, N. K. Ahmed, E. Koh, Higher-order network representation learning, in: Companion Proceedings of the The Web Conference 2018, 2018, pp. 3–4.

Journal Pre-proof

Journal Pre-proof

[39] C. Yang, M. Liu, V. W. Zheng, J. Han, Node, motif and subgraph: Leveraging network functional blocks through structural convolution, in: 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), IEEE, 2018, pp. 47–52.

[40] F. Xia, S. Yu, C. Liu, J. Li, I. Lee, Chief: Clustering with higher-order motifs in big networks, IEEE Transactions on Network Science and Engineering 9 (3) (2022) 990–1005.

[41] M. Zhang, G. Wang, L. Ren, J. Li, K. Deng, B. Zhang, Metonr: A meta explanation triplet oriented news recommendation model, Knowledge-Based Systems 238 (2022) 107922.

[42] R. A. Rossi, R. Zhou, N. K. Ahmed, Deep inductive network representation learning, in: Companion Proceedings of the The Web Conference 2018, 2018, pp. 953–960.

[43] S. Wu, F. Sun, W. Zhang, X. Xie, B. Cui, Graph neural networks in recommender systems: A survey, ACM Comput. Surv. (may 2022).

[44] M. Gao, L. Chen, X. He, A. Zhou, Bine: Bipartite network embedding, in: The 41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '18, Association for Computing Machinery, New York, NY, USA, 2018, p. 715–724.

[45] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, 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, 2018, pp. 974–983.

[46] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, Advances in neural information processing systems 30 (2017).

[47] F. Monti, K. Otness, M. M. Bronstein, Motifnet: a motif-based graph convolutional network for directed graphs, in: 2018 IEEE Data Science Workshop (DSW), IEEE, 2018, pp. 225–228.

[48] A. Sankar, X. Zhang, K. C.-C. Chang, Motif-based convolutional neural network on graphs, arXiv preprint arXiv:1711.05697 (2017).

Journal Pre-proof

Journal Pre-proof

[49] J. B. Lee, R. A. Rossi, X. Kong, S. Kim, E. Koh, A. Rao, Graph convolutional networks with motif-based attention, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM '19, Association for Computing Machinery, New York, NY, USA, 2019, p. 499–508.

[50] L. Luo, K. Liu, D. Peng, Y. Ying, X. Zhang, A motif-based graph neural network to reciprocal recommendation for online dating, in: International Conference on Neural Information Processing, Springer, 2020, pp. 102–114.

[51] R. Diestel, Graph Theory, fifth edition Edition, Vol. 173 of Graduate Texts in Mathematics, Springer, New York, 2017.

[52] A. R. Benson, D. F. Gleich, J. Leskovec, Higher-order organization of complex networks, Science 353 (6295) (2016) 163–166.

[53] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, Y. Bengio, Graph attention networks, arXiv preprint arXiv:1710.10903 (2017).

[54] M. Nguyen, J. Yu, T. Nguyen, Y. Han, Attentional matrix factorization with context and co-invocation for service recommendation, Expert Systems with Applications 186 (2021) 115698.

[55] X. Wang, X. He, M. Wang, F. Feng, T.-S. Chua, Neural graph collaborative filtering, in: Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval, 2019, pp. 165–174.

[56] M. Nguyen, J. Yu, T. Nguyen, S. Yongchareon, High-order autoencoder with data augmentation for collaborative filtering, Knowledge-Based Systems 240 (2022) 107773.

Journal Pre-proof

Journal Pre-proof

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Journal Pre-proof