Community-Based Influence Maximization Using Network Embedding in Dynamic Heterogeneous Social Networks

Qin, Xi; Zhong, Cheng; Lin, Hai Xiang

DOI
10.1145/3594544

Publication date
2023

Document Version
Final published version

Published in
ACM Transactions on Knowledge Discovery from Data

Community-Based Influence Maximization Using Network Embedding in Dynamic Heterogeneous Social Networks

XI QIN, School of Computer Science and Engineering, South China University of Technology and School of Computer, Electronics and Information, and the Key Laboratory of Parallel and Distributed Computing Technology in Guangxi Universities, Guangxi University
CHENG ZHONG, School of Computer, Electronics and Information, and the Key Laboratory of Parallel and Distributed Computing Technology in Guangxi Universities, Guangxi University
HAI XIANG LIN, Delft Inst Appl Math, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology

Influence maximization (IM) is a very important issue in social network diffusion analysis. The topology of real social network is large-scale, dynamic, and heterogeneous. The heterogeneity, and continuous expansion and evolution of social network pose a challenge to find influential users. Existing IM algorithms usually assume that social networks are static or dynamic but homogeneous to simplify the complexity of the IM problem. We propose a community-based influence maximization algorithm using network embedding in dynamic heterogeneous social networks. We use DyHATR algorithm to obtain the propagation feature vectors of network nodes, and execute k-means cluster algorithm to transform the original network into a coarse granularity network (CGN). On CGN, we propose a community-based three-hop independent cascade model and construct the objective function of IM problem. We design a greedy heuristics algorithm to solve the IM problem with $(1 - \frac{1}{e})$−approximation guarantee and use community structure to quickly identify seed users and estimate their influence value. Experimental results on real social networks demonstrated that compared with existing IM algorithms, our proposed algorithm had better comprehensive performance with respect to the influence value, more less execution time and memory consumption, and better scalability.

CCS Concepts: • **Computing methodologies** → **Supervised learning**; • **Theory of computation** → *Design and analysis of algorithms*; • **Applied computing** → Sociology;

Additional Key Words and Phrases: Network embedding, community diffusion, feature learning, feature representation

This work was supported by the Special Project of Science and Technology Development Research of Guangxi under grant no. ZL19107008.
Authors' addresses: X. Qin, School of Computer Science and Engineering, South China University of Technology, No. 382, Outer East Road, Panyu University City, Guangzhou, Guangdong 510641 China, and School of Computer, Electronics and Information, and the Key Laboratory of Parallel and Distributed Computing Technology in Guangxi Universities, Guangxi University, No. 100 Daxue Road East, Nanning, Guangxi 530004 China; email: qinxi@gxu.edu.cn; C. Zhong (corresponding author), School of Computer, Electronics and Information, and the Key Laboratory of Parallel and Distributed Computing Technology in Guangxi Universities, Guangxi University, No. 100 Daxue Road East, Nanning, Guangxi 530004 China; email: chzhong@gxu.edu.cn; H. X. Lin, Delft Inst Appl Math, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Mekelweg 4, Delft, South Halland Netherlands 2628 CD; email: h.x.lin@tudelft.nl.

1556-4681/2023/06-ART119 $15.00
https://doi.org/10.1145/3594544

ACM Reference format:
Xi Qin, Cheng Zhong, and Hai Xiang Lin. 2023. Community-Based Influence Maximization Using Network Embedding in Dynamic Heterogeneous Social Networks. *ACM Trans. Knowl. Discov. Data.* 17, 8, Article 119 (June 2023), 21 pages.
https://doi.org/10.1145/3594544

1 INTRODUCTION

Social networking refers to the use of internet-based social media sites to maintain connections with people. Virus marketing is to use existing social networks to promote products on social media platforms, and it uses public enthusiasm and interpersonal networks to spread marketing information like viruses. The word-of-mouth effect has been confirmed to be effective in rapidly propagating information at a low cost. Hence, how to find the top-k most influential users in the social networks, called seeds, and estimate the influence of these seeds, is an important research topic in academic and industrial community which has been described as the **influence maximization** (**IM**) problem [6, 25]. The IM analysis methods usually assess the diffusion among nodes based on the network topology and determine the influence of seeds by computing the propagation probability of edges throughout the network. Many IM algorithms focus on how to adapt to large-scale network and improve their running efficiency [12, 19, 27]. The topology of social networks are constantly changing, such as nodes increase, edges increase, and edge relationship change. To solve this problem, researchers have started to incorporate the dynamism of network structure into the design of IM algorithms [1, 3, 11, 17, 18, 22, 24, 31]. With the development of social network, there are multiple node types and edge types in the network. For example, in the early rules of social networks, the spread of information was primarily based on the friendship connections between users, and the topology relationship was limited to Node-Node. However, nowadays, users have the ability to comment or respond to posts from strangers. It means that the way of information spreads has evolved into Node-Item-Node. Therefore, researchers have begun to focus on exploring IM algorithms in heterogeneous networks [15, 28]. As current social networks have the characteristics for large-scale, dynamic, and heterogeneous, to solve IM algorithms has more challenging.

To solve the above problem, we present an IM algorithm using **dynamic heterogeneous network embedding** (**DHNE**) and community diffusion. This IM algorithm can be applied on dynamic heterogeneous networks with over tens of millions of edges. By utilizing the DyHATR algorithm [33], we are able to learn the heterogeneous and dynamic propagation characteristics of nodes from the original network, and generate a propagation feature vector for each node in the network. Then, we apply the k-means cluster algorithm to obtain the community structure and establish a **coarse-grained network** (**CGN**). On CGN, we propose a community-based three-hop **independent cascade** (**IC**) model, and design a greedy heuristic IM algorithm. We use community structure to quickly identify seed users and estimate their influence value. The contributions of this article are as follows.

(1) We propose a DHNE-based community diffusion model. The model uses network embedding, which represents dynamic heterogeneous propagation features using low-dimensionally dense feature vectors, to enhance the accuracy of community diffusion. Additionally, the proposed model uses the community structure to reduce its spatial complexity.
(2) We propose a **DHNE-based community influence maximization** (**DHNE-CIM**) algorithm with the $(1 - \frac{1}{e})-$ approximation guarantee to improve the efficiency of seed search and influence estimation.
(3) Experimental results on three large-scale dynamic heterogeneous networks demonstrated that our proposed algorithm had better comprehensive performance with respect to

influence value, more less execution time and memory requirement, and better scalability than existing IM algorithms.

The remainder of the article is organized as follows. In Section 2, we summarize related work. In Section 3, we give the problem statement. In Section 4, we introduce the proposed DHNE-based community diffusion model in detail. In Section 5, we describe the DHNE-CIM algorithm. Section 6 reports the experimental results. Section 7 concludes the artcile.

2 RELATED WORK

Kempe et al. [13] formally defined the IM problem. They proved that although IM problem is NP-hard, if the influence propagation function satisfies non-negativity, monotonicity, and submodularity, the greedy framework can be used to solve the IM problem with the $(1-\frac{1}{e}-\varepsilon)$-approximate ratio, where e is the base of the natural logarithm and ε is the sampling error. Kempe et al. also proposed two classic IM propagation models, namely the **linear threshold** (**LT**) model and the **independent cascade** (**IC**) model. For classical IM problem, researchers proposed some effective IM algorithms, such as CELF [16], CELF++ [10], TIM [30], and IMM [29]. With development of social networks, the continuously increasing number of users has resulted in very large-scale of social networks. Some researchers utilize community structures to optimize diffusion models and IM algorithms to compute IM problem for large-scale social network. Belak et al. [2] designed a cross-community influence strategy to achieve coarse-grained spread for social networks. To significantly accelerate IM calculation at the group level, Eftekhar et al. [7] proposed a coarse-grained diffusion model. Ji et al. [12] proposed a community diffusion model using network embedding to solve the IM problem of large-scale networks with millions of nodes. Based on supervised learning and reinforcement learning, Manchanda et al. [19] proposed a two-stage optimization framework called GCOMB to solve the IM problem with constraints on a super large network with billions of nodes. The above algorithms mainly focus on addressing how to improve the execution efficiency of IM problem, but they are only applicable to static social networks.

Dynamic evolution is a significant feature of real social networks. Therefore, researchers have studied how to perform IM analysis on the dynamic social networks. Aggarwal et al. [1] defined the initial graph G^0 and evolution graph G^t in time interval $[t, t+h]$ to represent the dynamic social network, and calculated the influence value and backtracked the seed set. To mine multiple seed sets at different times, Chen et al. [3] designed an **upper bound interchange** (**UBI**) greedy algorithm with 1/2-approximation guarantee. To analyze the influence of evolving networks, Ohsaka et al. [22] proposed a reachability tree-based technique and a sketching method with a real-time fully-dynamic index data structure to solve dynamic IM problem. To accelerate calculation, Meng et al. [20] proposed an efficient incremental algorithm to solve the dynamic IM problems in dynamic IC model, and Nesrine et al. [11] designed an incremental IM algorithm for dynamic social networks. With further study of **dynamic network influence maximization** (**DIM**), more algorithms for solving complex DIM problems have been proposed. Min et al. [21] constructed a topic-based time-sensitive dynamic propagation model, and designed a topic-based time-aware greedy algorithm and a topic-based time-aware heuristic algorithm to find seed set. Qin et al. [24] proposed a dynamic IM algorithm based on community-topic features. Yerasani et al. [35] further considered the budget constraint on the DIM problem and proposed a memetic algorithm to identify the most influential users at different time intervals. The above works mainly focused on solving the problem of DIM, and some ones began to focus on the problem of complex DIM.

In recent years, researchers have found that social networks are composed of multiple kinds of entities coexisting, and users frequently engage in several types of interactions. Therefore, the diversity of node and edge types is considered in designing IM algorithms. Kermani et al. [14]

proposed a novel competitive influence model to incorporate users heterogeneity, message content, and network structure. To solve the problem of multiple behavior edges among users, Feng et al. [8] designed an Inf2vec algorithm, which learns the node vectors from user behavior edges to represent social influence information and searches for seeds through vector relationships. To address the existence of heterogeneous nodes in social networks, such as user nodes and message nodes, Deng et al. [5] proposed a measuring influence model to capture the influence of heterogeneous social networks. Facing the situation that the network contains multiple types of nodes and multiple types of edges, Wang et al. [32] developed an improved algorithm to find the most influential users in the heterogeneous networks. To solve the IM problem on multiple heterogeneous networks, Kuhnle et al. [15] designed a multiplex IM algorithm with a heterogeneous diffusion model. Existing **heterogeneous influence maximization** (**HIM**) algorithms were mainly focused on static network. When the heterogeneous network structure changes over time, the existing HIM algorithms must recalculate all data.

To sum up, the existing IM methods are mostly focused on large-scale static homogeneous networks, static heterogeneous networks, and dynamic homogeneous networks. In this article, we focus on addressing IM problem in large-scale, dynamic, and heterogeneous social networks.

3 PROBLEM STATEMENT

In real social network, there are multiple types of nodes and edges, and the relationships among nodes in the network change over time. Therefore, real social networks are dynamic and heterogeneous. The dynamic heterogeneous social network is defined as follows.

Definition 1 (Dynamic Heterogeneous Social Network (DHSN)). Reference [36] can be defined as a set of sequential time snapshots within T time steps, denoted as $G = \{G^1, G^2, \ldots, G^T\}$. Each snapshot is a heterogeneous social network, denoted as $G^t = (V^t, E^t, \phi, \varphi)$, where V^t and E^t denote the set of nodes and set of edges at tth time step, respectively, $\phi : V^t \rightarrow \Gamma_V$ is the node type mapping function, $v \in V^t$ corresponds to a specific type in Γ_V, $\varphi : E^t \rightarrow \Gamma_E$ is edge type mapping function, $e \in E^t$ corresponds to a specific type in Γ_E, $|\Gamma_V| + |\Gamma_E| > 2, t = 1, 2, \ldots, T$.

The classical IM problem aims to identify a set of k most influential users in social network, called seed set, and to estimate the influence value of seed set. However, it is too complicated to directly use the edge diffusion information in DHSN to solve the dynamic heterogeneous influence maximization problem, so we use a mapping function $f(\cdot)$ to compress the high-dimensional propagation characteristics of nodes in the original space into a set of low dimensional node vectors $Y^t = \{y_v^t = f(v) | v \in V^t, y_v^t \in \mathbb{R}^d\}$ [23, 36], where $Y^t \in \mathbb{R}^{|V^t| \times d}$ are dynamic heterogeneous latent feature representation of $V^t, t = 1, 2, \ldots, T$. The mapping process of nodes is network embedding, and the **network embedding-based** (**NE-based**) DHIM is defined as follows.

Definition 2 (NE-based Dynamic Heterogeneous Influence Maximization (NE-based DHIM)). Given a DHSN $G = \{G^1, G^2, \ldots, G^T\}$, $G^t = (V^t, E^t, \phi, \varphi)$, a set of latent feature representation of nodes $Y^t = \{y_v^t = f(v) | v \in V^t, y_v^t \in \mathbb{R}^d\}$, a user seed set $S^t \subseteq V^t$, a positive integer k, and an information diffusion model $\Omega(Y^t)$, which capture the stochastic process of S^t spreading information on G^t, NE-based DHIM aims to find a k-size seed set $S^{t*} \subseteq V^t$, which can maximize the number of affected nodes under the diffusion model Ω with parameter Y^t, and the DHIM problem can be formulated to solve S^{t*} as follows.

$$S^{t*} = \arg\max \sigma_{G,\Omega(Y^t)}(S^t), S^t \subseteq V^t, Y^t \in \mathbb{R}^{|V^t| \times d}, |S^t| = k, t = 1, 2, \ldots, T, \tag{1}$$

where $\sigma_{G,\Omega(Y^t)}(S^t)$ represents the influence value of S^t. We omit the subscript of $\sigma_{G,\Omega(Y^t)}(S^t)$ and simplified it as $\sigma(S^t)$ when the context is clear.

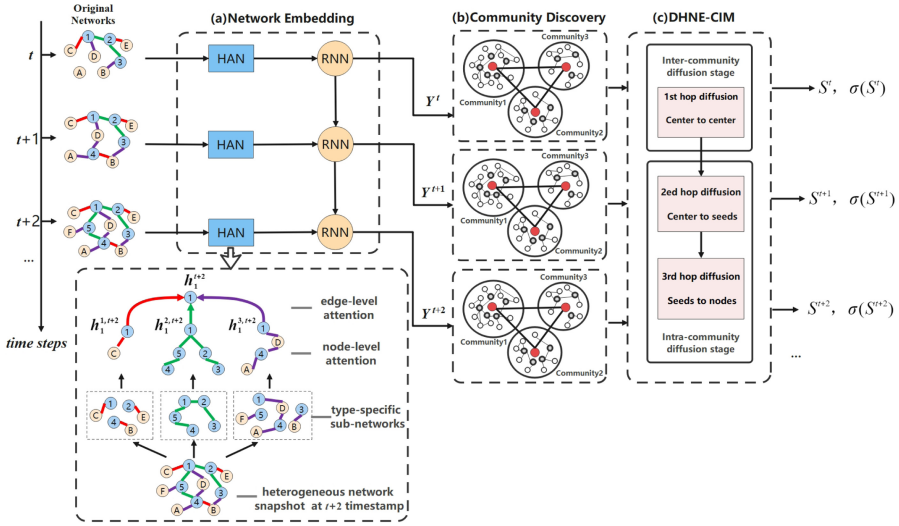

Fig. 1. Framework of DHNE-based community diffusion model.

4 PROPOSED MODEL FRAMEWORK

To solve the NE-based DHIM problem, we propose a DHNE-based community diffusion model. The framework of our model is shown in Figure 1, where HAN is hierarchical attention network and RNN is recurrent neural network.

The model first uses DHNE algorithm DyHATR [33] to learn the latent diffusion feature representation of network nodes (Figure 1(a)). Then, the model cluster nodes in the latent feature space to find communities, and use the cluster center vector as the embedding representation of community (Figure 1(b)). Finally, the DHNE-CIM algorithm is used to track and identify the seed set and calculate the influence value of the seed set at each time step (Figure 1(c)). Table 1 lists the important symbols used in the proposed model.

4.1 Dynamic Heterogeneous Social Network Embedding

Network embedding is a way to represent high-dimensional sparse network with low-dimensional dense vector space, and it not only captures the heterogeneous topology of the network, but also captures the evolution of the network topology. We use the algorithm DyHATR [33] to learn embedding representation for DHSN.

Given a DHSN $G = \{G^1, G^2, \ldots, G^T\}$, $G^t = (V^t, E^t, \phi, \varphi)$, DyHATR first captures network heterogeneous information at each time step by HAN model. The model HAN first divides G^t into several type-specific sub-networks according to edge type, uses node-level attention model with multi-head mechanisms to learn weight coefficient of node pair(v, u) in the rth type at tth snapshot $\alpha^{rt}_{v,u}$, and unifies the multi-type node relationship as user node embedding $\hat{h}^{rt}_v$ [33]:

$$\begin{aligned} \alpha^{rt}_{v,u} &= \frac{\exp(\tau(a_r^\top \cdot [W^r \cdot x_v || W^r \cdot x_u]))}{\sum_{k \in N^{rt}_v} \exp(\tau(a_r^\top \cdot [W^r \cdot x_v || W^r \cdot x_k]))}, \\ \hat{h}^{rt}_v &= \tau\left(\sum_{u \in N^{rt}_v} \alpha^{rt}_{v,u} \cdot W^r \cdot x_u\right), \end{aligned} \tag{2}$$

Table 1. The Important Symbols and Their Meanings in Proposed Model

Symbol	Description
G	dynamic heterogeneous social network
T	total number of time steps
t	the tth time step, and $t = 1, 2, \ldots, T$
G^t	the snapshot of G at time step t
V^t	the node set of the network snapshot at time step t
E^t	the edge set of the network snapshot at time step t
ϕ	the node type mapping function
φ	the edge type mapping function
Γ_V	the collection of node type
Γ_E	the collection of edge type
S^t	a user set of time step t
S^{t*}	the seed set of time step t
$\sigma_{G,\Omega(Y^t)}(S^t)$,$\sigma(S^t)$	influence value of S^t
Y^t	latent feature representation set of nodes at time step t
Ω	information diffusion model
$f(\cdot)$	the mapping function from the original network space to the network latent feature space
$p(y_v^t, y_u^t)$	the diffusion proximity between nodes v and $u \in V^t$
c_i^t	a community in G^t
$z_{c_i}^t$	the center vector of community c_i^t
M	the number of communities
k	the size of seed set
m	the number of candidate communities
l	the number of seeds in each candidate community
H^t	the coarse granularity network of G^t
c_{int}^t	the diffusion entrance in first-hop diffusion
$N(c_{int}^t)$	the neighbor communities of c_{int}^t
CC^t	the candidate community set in G^t
$q_{c_i}^{'t}$	the influence probability of first-hop diffusion
$q_{c_i,s}^{''t}$	the influence probability of second-hop diffusion
$q_{s,v}^{'''t}$	the influence probability of third-hop diffusion

where $\tau(\cdot)$ is the activation function, $||$ indicates the concatenation operation, r represents the edge type, t indicates the time step, x_v represents the initial feature vector of node v, W^r is conversion matrix of the rth edge type, N_v^{rt} is the set of sampled neighbors of node v for the rth edge type in tth snapshot, $a_r^\top$ indicates the transposition of parameterized weight vector of attention function in the rth edge type, $\hat{h}_v^{rt}$ is embedding of node v for the rth type edge in tth snapshot. Because the multi-head node-level attention model has θ attention-heads, it is necessary to aggregate the learning results of all attention-heads by formula Equation (5), where $\hat{h}_v^\theta$ is a simplified symbol of $\hat{h}_v^{rt}$, and the aggregating result is h_v^{rt} [33]:

$$h_v^{rt} = concat(\hat{h}_v^1, \hat{h}_v^2, \ldots, \hat{h}_v^\theta), \tag{3}$$

where $h_v^{rt} \in \mathbb{R}^L$, and $L \ll |V^t|$ is the dimension of each node embedding learned by node-level attention model.

Further, the model HAN uses edge-level attention model to aggregate node embeddings under all edge types to generate embedding representation of node v in the tth snapshot h_v^t [33]:

$$\begin{aligned} \beta_v^{rt} &= \frac{\exp(q^\top \cdot \tau(W \cdot h_v^{rt} + b))}{\sum_{k=1}^{|\Gamma_E|} \exp(q^\top \cdot \tau(W \cdot h_v^{kt} + b))}, \\ h_v^t &= \sum_{r=1}^{|\Gamma_E|} \beta_v^{rt} \cdot h_v^{rt}, \end{aligned} \tag{4}$$

where β_v^{rt} is normalized weight coefficient of node v for the rth edge type at the tth snapshot, W is learning weight matrix, b is bias vector, $q^\top$ indicates transposition of the edge-level attention parameter vector. After learning, the model HAN outputs node embedding set at each time snapshot, $\{h_1^t, h_2^t, \ldots, h_{|V^t|}^t\}, h_v^t \in \mathbb{R}^F, F \ll |V^t|, t = 1, 2, \ldots, T$, where F is the dimension of each node embedding learned by edge-level attention model.

Finally, DyHATR algorithm uses the RNN model to connect multiple snapshots in consecutive time, and learns evolutionary mode of network from snapshot sequence. The **gated-recurrent-unit** (**GRU**) [4] in RNN is configured as follows [33]:

$$\begin{aligned} u^t &= \tau(W_u \cdot [h_v^t || y_v^{t-1}] + b_u), \\ r^t &= \tau(W_r \cdot [h_v^t || y_v^{t-1}] + b_r), \\ \tilde{y_v^t} &= \tanh(W_y \cdot [h_v^t || (r^t \odot y_v^{t-1})] + b_y), \\ y_v^t &= (1 - u^t) \odot y_v^{t-1} + u^t \odot \tilde{y_v^t}, \end{aligned} \tag{5}$$

where $u^t \in \mathbb{R}^d$ is update gate vector, $r^t \in \mathbb{R}^d$ is reset gate vector, $W_u, W_r, W_y \in \mathbb{R}^{d \times 2F}$ and $b_u, b_r, b_y \in \mathbb{R}^d$ represent training parameter matrixes and bias vectors, respectively. The outputs of the RNN model are the final embedding representation of DHSN, denoted as $Y^t = \{y_v^t | v \in V^t, y_v^t \in \mathbb{R}^d\}, t = 1, 2, \ldots, T$, where d is the dimension of the final embedding vector.

4.2 Diffusion Model and Influence Estimation

We combine DHNE with the classical IC model to design a community diffusion model. There are three definitions related to the diffusion model.

Definition 3 (Diffusion Proximity). Represents the similarity of propagation features between two nodes. Given a DHSN $G = \{G^1, G^2, \ldots, G^T\}$, $G^t = (V^t, E^t, \phi, \varphi)$ and its latent feature representation $Y^t = \{y_v^t | v \in V^t, y_v^t \in \mathbb{R}^d\}, t = 1, 2, \ldots, T$, the diffusion proximity between nodes v and $u \in V^t, p(y_v^t, y_u^t)$ can be defined as follows:

$$p(y_v^t, y_u^t) = \frac{1}{2}\left(1 - \frac{y_v^t \cdot y_u^t}{\|y_v^t\| \times \|y_u^t\|}\right). \tag{6}$$

Diffusion proximity is normalized cosine distance. The larger the diffusion proximity, the greater the influence probability between two nodes.

Definition 4 (Community). is a set of nodes with similar propagation features to each other. The community in G^t can be denoted as $c_i^t, 0 \le i \le M$, i is the community number and M is the number of communities in the network snapshot. If the nodes in the network have community tags, we take the average vector of community nodes as the community center. If the nodes in the network do not have community tags, we identify the communities by clustering algorithm, and take the cluster center as the community center. The center vector of community c_i^t is denoted as $z_{c_i}^t, 0 \le i \le M, t = 1, \ldots, T$.

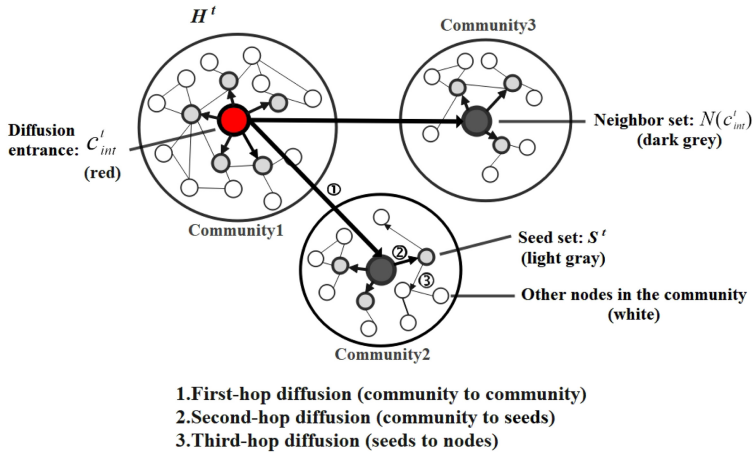

Fig. 2. Skeleton of DHNE-based community diffusion model.

Definition 5 (Coarse Granularity Network (CGN)). is a network with community as the unit. CGN is denoted as H^t, where $H^t = \{c_i^t | 0 \le i \le M\}, t = 1, 2, \ldots, T$.

Our diffusion model is a community-based three-hop IC model on CGN. The first hop is inter-community diffusion, and the second and the third hops are intra-community diffusion. A skeleton of our proposed DHNE-based community diffusion model is shown in Figure 2. The model finds m candidate communities through the first-hop diffusion, finds l seeds in each candidate community via the second-hop diffusion, and calculates the influence value of $k = m \times l$ seeds through the third-hop diffusion.

For the network snapshot G^t, we set the largest community as the diffusion entrance in first-hop diffusion, and denote it as c_{int}^t. The first-hop diffusion is the spread from c_{int}^t to its neighbor communities $N(c_{int}^t)$. The calculation of $N(c_{int}^t)$ is as follows:

$$N(c_{int}^t) = \left\{ c_i^t | max(p(z_{c_{int}}^t, z_{c_i}^t), m-1), \forall c_i^t \in H^t \backslash c_{int}^t \right\}, \tag{7}$$

where $p(z_{c_{int}}^t, z_{c_i}^t)$ means the diffusion proximity from the entrance community center to each community center. We calculate the diffusion proximity to search the $m-1$ communities that have the closest diffusion relationship with c_{int}^t, and add them into $N(c_{int}^t)$. $c_{int}^t \cup N(c_{int}^t)$ constitutes candidate community set, denoted as CC^t, where $|CC^t| = m, m \le |H^t|$. The influence probability of first-hop diffusion, $q_{c_i}^{\prime t}$, can be calculated as follows:

$$q_{c_i}^{\prime t} = p(z_{int}^t, z_{c_i}^t), \forall c_i^t \in H^t \backslash c_{int}^t. \tag{8}$$

The second-hop diffusion is the spread from the candidate community centers to their adjacency nodes. After the second-hop diffusion, the seed set S^t can be obtained

$$\begin{aligned} S_{c_i}^t &= \{v | max(p(z_{c_i}^t, y_v^t), l), \forall v \in c_i^t, \forall c_i^t \in CC^t\}, \\ S^t &= \cup_{c_i^t \in CC^t} S_{c_i}^t. \end{aligned} \tag{9}$$

We look for l nodes with the greatest diffusion proximity to $z_{c_i}^t$ and add them into the candidate seed set $S_{c_i}^t$. The influence probability of second-hop diffusion, $q_{c_i,s}^{\prime\prime t}$, can be calculated as follows:

$$q_{c_i,s}^{\prime\prime t} = p(z_{c_i}^t, y_s^t), \forall c_i^t \in CC^t, \forall s \in S_{c_i}^t. \tag{10}$$

The third-hop diffusion is the spread from the seeds in candidate community to other inactive nodes within the same community. The influence probability of each node inside the candidate communities, $q'''^t_{s,v}$, are calculated as follows:

$$q'''^t_{s,v} = \sum_{v \in c_i^t \setminus S_{c_i}^t} p(y_s^t, y_v^t), \forall s \in S_{c_i}^t. \tag{11}$$

Then, the influence value $\sigma(S^t)$ under community-based dynamic heterogeneous diffusion model is calculated as follows:

$$\sigma(S^t) = \sum_{c_i^t \in CC^t} \sum_{s \in S_{c_i}^t} \sum_{v \in c_i^t \setminus S_{c_i}^t} q'^t_{c_i} q''^t_{c_i,s} q'''^t_{s,v}. \tag{12}$$

Our model finds seeds by coarse-grained propagation at community level, and the influence value of seed set is directly estimated by formulas Equation (12). The model does not use the time-consuming single-node diffusion and Monte Carlo simulation, so that it can effectively reduce the time and spatial complexity of IM calculation. Next, we prove the approximation guarantee of solving IM problem using our model.

Kempe et al. [13] proved that if the influence propagation function satisfies non-negativity, monotonicity, and submodularity, then the greedy method can be used to solve IM problem with the $(1-\frac{1}{e})$−approximation guarantee. Because $\sigma(\cdot) : \mathbb{R}^{|V|^t \times d} \rightarrow \mathbb{R}$ is non-negative and monotonic, we only need to prove the submodularity of $\sigma(\cdot)$, and we can use the greedy-based algorithm to approximately solve the NE-based DHIM problem with $(1-\frac{1}{e})$−approximation guarantee.

Submodularity: $\sigma(\cdot)$ is submodular if and only if $\sigma(S^t \cup \{x\}) - \sigma(S^t) \geq \sigma(R^t \cup \{x\}) - \sigma(R^t)$ holds for two seed sets S^t and R^t, $S^t \subseteq R^t, x \in c$.

Proof.

$$\sigma(S^t \cup \{x\}) - \sigma(S^t) = q'^t_c q''^t_{c,x} \sum_{v \in c \setminus (S^t \cup \{x\})} q'''^t_{x,v},$$

$$\sigma(R^t \cup \{x\}) - \sigma(R^t) = q'^t_c q''^t_{c,x} \sum_{v \in c \setminus (R^t \cup \{x\})} q'''^t_{x,v}.$$

Because $S^t \subseteq R^t$, $c \setminus (R^t \cup \{x\}) \subseteq c \setminus (S^t \cup \{x\})$. Hence, $q'^t_c q''^t_{c,x} \sum_{v \in c \setminus (S^t \cup \{x\})} q'''^t_{x,v} \geq q'^t_c q''^t_{c,x} \sum_{v \in c \setminus (R^t \cup \{x\})} q'''^t_{x,v}$ That is to say, $\sigma(\cdot)$ is submodular. □

5 ALGORITHM

In this section, we give a specific description of the DHNE-CIM algorithm mentioned in the framework of DHNE-based community diffusion model. The inputs of DHNE-CIM algorithm include DHSN G, number T of time steps, number M of communities, number m of candidate communities, number l of seeds in each candidate community, dimension d of embedding vector, and number r of types in heterogeneous network. The final output of DHNE-CIM algorithm consists of k-size seed set S^t and their corresponding influence values $\sigma(S^t)$ for T time steps, where $k = m \times l$ and $t = 1, \ldots, T$. At each time step, DHNE-CIM algorithm first executes DyHATR algorithm [33] to calculate the embedding representation Y^t for input network G^t and generates the CGN H^t by running k-means clustering algorithm on Y^t. Secondly, DHNE-CIM algorithm chooses the initial diffusion community c_{int}^t from H^t. Then, DHNE-CIM algorithm determines m candidate communities and searches l seeds in each candidate community. Finally, the influence value of seed set $\sigma(S^t)$ is calculated.

During the embedding stage, the algorithm DyHATR [33] is executed to learn the embedding representation of DHSN. At each time step, DyHATR uses the HAN [34] model to aggregate the

ALGORITHM 1: DHNE-CIM

Input: G, T, M, m, l, d, r
Output: $S^1, S^2, \ldots, S^T$ and $\sigma(S^1), \sigma(S^2), \ldots, \sigma(S^T)$
// 1.Embedding stage: calculate the embedding representation of DHSN
$(Y^1, Y^2, \ldots, Y^T) \leftarrow DyHATR(G, T, d, r)$;
// Community discovery and generating CGN
for $t = 1; t \leq T; t++$ **do**
  // Cluster Y^t into M classes by k-means algorithm, H^t is CGN and Z^t is community center matrix
  $\{H^t, Z^t\} \leftarrow k - means(Y^t, M)$;
end
// 2.IM calculation stage
for $t = 1; t \leq T; t++$ **do**
  // 1st-hop diffusion
  $f \leftarrow \emptyset, S^t \leftarrow \emptyset, \sigma(S^t) \leftarrow 0.0$;
  for $i = 0; i \leq |H^t| - 1; i++$ **do**
    // Initialize the propagation entrance community
    if $isMax(|c_i^t|)$ **then**
      $int \leftarrow i$;
      $c_{int}^t \leftarrow H^t[int]$;
    end
  end
  $z_{c_{int}}^t \leftarrow Z^t[int]$; // Find the entrance community from H^t by community index
  Find out $N(c_{int}^t)$ according to formula Equation (7);
  $CC^t \leftarrow c_{int}^t \cup N(c_{int}^t)$;
  // 2ed-hop diffusion
  for *each community* c_i^t *in* CC^t **do**
    Calculate $q'^t_{c_i}$ according to formula Equation (8);
    $S^t \leftarrow$ Calculate $S_{c_i}^t$ according to formula Equation (9);
    for *each community* s *in* $S_{c_i}^t$ **do**
      Calculate $q''^t_{c_i,s}$ according to formula Equation (10);
      // 3rd-hop diffusion
      for *each community* v *in* $c_i^t \backslash S_{c_i}^t$ **do**
        $inf \leftarrow inf + q'^t_{c_i} q''^t_{c_i,s} q'''^t_{s,v}$;
      end
    end
    $\sigma(S^t) \leftarrow \sigma(S^t) + inf$;
  end
end
return $S^1, S^2, \ldots, S^T$ *and* $\sigma(S^1), \sigma(S^2), \ldots, \sigma(S^T)$;

features from several type-specific sub-networks, and uses the RNN [26] model to learn the evolving features. The required time and space of running DyHATR are $O(n^2)$, where n is the number of nodes in the network. Next, DHNE-CIM executes algorithm k-means algorithm to cluster the embedding network Y^t, and its required time and space are $O(n)$. The IM calculation stage required time and space are $O(k \times c) \approx O(n)$, respectively, where k is the number of seeds, and c is the number of nodes in each community.

Table 2. Required Time, Space, and Approximation Ratio of the Five Algorithms

Algorithm	Time Complexity in Embedding Stage	Space Complexity in Embedding Stage	Time Complexity of IM algorithm	Space Complexity of IM algorithm	Approximation Ratio
DHNE-CIM	$O(n^2)$	$O(n^2)$	$O(n)$	$O(n)$	$1-1/e$
GroupIM [12]	$O(m'+n+nlogn)$	$O(m'+n+nlogn)$	$O(m'log(m')+n^2+m\times c)$	$O(m'log(m')+n^2+m\times c)$	$1-1/e$
Inf2vec [8]	$O(n)$	$O(n)$	$O(n^2)$	$O(n^2)$	$1-1/e$
DIM [22]	-	-	$O(\frac{(m'+n)\log n)}{\varepsilon^3})$	$O(\frac{(m'+n)\log n)}{\varepsilon^3})$	$1-1/e-\varepsilon$
KSN [15]	-	-	$O(\frac{(m'+n)\log n)}{\varepsilon^2})$	$O(\frac{(m'+n)\log n)}{\varepsilon^2})$	$\frac{(1-\varepsilon)(1-1/e-\varepsilon)}{(o+1)r}$

"-" denotes that algorithms DIM and KSN do not need embedding processing.

According to Reference [9], if $\sigma(\cdot)$ is non-negative, monotonic, submodular, and $\sigma(\emptyset)=0$, for $\hat{S}^t$ obtained by the greedy strategy-based IM algorithm, $\sigma(\hat{S}^t) \geq (1-(1-\frac{1}{k})^k)\times\sigma(S^{t*})$ holds. Because $1-\frac{1}{e} < (1-(1-\frac{1}{k})^k), k>0$, and $\lim_{k\to\infty}(1-(1-\frac{1}{k})^k) = 1-\frac{1}{e}$. So, the approximation ratio of algorithm DHNE-CIM is $(1-\frac{1}{e})$.

Table 2 shows the time-space complexity and approximation ratio of our proposed algorithm DHNE-CIM with other four existing algorithms GroupIM [12], Inf2vec [8], DIM [22], and KSN [15], where n is the number of nodes in network, m' is the number of edges in the network, m and c are the number of candidate communities and the number of nodes in each candidate community in algorithms DHNE-CIM and GroupIM, respectively, o is the number of nodes overlapping in multiple networks and r is the number of edge types in algorithm KSN, respectively. Algorithms DIM and KSN do not need embedding processing, so we use "-" to fill in the "Time Complexity in Embedding Stage" and "Complexity in Embedding Stage" columns in Table 2.

The three algorithms DHNE-CIM, Inf2vec, and GroupIM are NE-based IM algorithms, so they need to execute embedding processing. DHNE-CIM learns dynamic heterogeneous information, Inf2vec learns static heterogeneous information, and GroupIM learns static homogeneous information. So, during the embedding stage, the time-space complexity of algorithms DHNE-CIM is higher than that of algorithms Inf2vec and GroupIM. For the propagation stage, because $O(n) < O(\frac{(m'+n)\log n}{\varepsilon^2}) < O(\frac{(m'+n)\log n}{\varepsilon^3}) < O(n^2)$,the time-space complexity of DHNE-CIM is the lowest among the five algorithms, Furthermore, $\frac{(1-\varepsilon)}{(o+1)r}(1-\frac{1}{e}-\varepsilon) < (1-\frac{1}{e}-\varepsilon) < (1-\frac{1}{e})$, where $0<\varepsilon<1$, $r>2$. Therefore, the approximation ratio of algorithm DHNE-CIM is the same as that of algorithms Inf2vec and GroupIM, and higher than that of algorithms DIM and KSN.

6 EXPERIMENT

We conducted experiments on the high-performance parallel cluster system of Guangxi University.[1] The computing node used in the experiment is configured as a 40-core CPU, each of which contains two Intel Xeon Gold 6,230, 192 GB memory, and two Tesla 4 GPU cards. The operating system is CentOS 7.4. The three algorithms DHNE-CIM, GroupIM, and Inf2vec were implemented in Python language, the two algorithms DIM and KSN were implemented in C++ language.

6.1 Datasets

To compare the performance of algorithm DHNE-CIM with other four existing algorithms, we chose three open-source dynamic heterogeneous social network datasets. The first two datasets are SX-Supper[2] and Higgs-Twitter[3] provided by Stanford University. The third dataset is from

[1] hpc.gxu.edu.cn.
[2] https://snap.stanford.edu/data/sx-superuser.html.
[3] https://snap.stanford.edu/data/higgs-twitter.html.

Table 3. Statistics of Network Datasets Used

Dataset	*Node*	*Edge*	*Node Type*	*Edge Type*	*T*
SX	194,085	1,443,339	1	3	10
Higgs	456,625	14,855,842	1	3	10
Tencent	2,320,895	50,655,143	1	3	10

Table 4. Ability of Algorithms That Adapt to Large-Scale DHSN

Algorithms	Large-scale	Dynamic	Heterogeneous
DHNE-CIM	yes	yes	yes
GroupIM	yes	no	no
Inf2vec	no	no	yes
DIM	no	yes	no
KSN	no	no	yes

social network Tencent-Weibo[4] presented in KDD2012. Table 3 shows the key characteristics of the three datasets, where *Node* and *Edge* denote the number of nodes and the number of edges in the dataset respectively, *Node Type* is the number of node types contained in the dataset, *Edge Type* indicates the number of edge types in the dataset, and T represents the number of time steps.

6.2 Evaluation Indicators

We use four indicators, namely influence value, execution time, memory capacity used, and scalability, to evaluate the performance of the IM algorithms in the large-scale dynamic heterogeneous network environment.

(1) Influence value: It is the approximate solution of influence function $\sigma(\cdot)$, which is an indicator to evaluate the quality of seeds found by IM algorithm. In an IM algorithm, k influential nodes are selected as seeds for a given propagation model, and influence value of seeds corresponds to the total expectation of network nodes successfully influenced by seeds.
(2) Execution time: It represents the time to execute an IM algorithm, which searches for seeds and estimates influence value.
(3) Memory capacity used: It denotes the maximum amount of memory consumed during the execution of an IM algorithm
(4) Scalability: It is used to evaluate whether the IM algorithm has good expansion capability with growth of social network. It refers to the required time that the IM algorithm recompute the latest results along with dynamic growth of social network.

6.3 Algorithms in the Experiment

We selected four existing IM algorithms for experimental comparison. Among them, GroupIM [12] is an IM algorithm using network embedding and community diffusion, Inf2vec [8] is a heterogeneous IM algorithm using network embedding, DIM [22] is a classical dynamic IM algorithm using the sketching method, and KSN [15] is a multi-network heterogeneous IM algorithm based on the classic IMM algorithm [29] and knapsack algorithm. Table 4 lists the key features of the five IM algorithms.

[4]https://www.kaggle.com/c/kddcup2012-track1/data.

6.4 Experiment Setting

The task of dynamic IM algorithm is to track influence value, execution time, memory capacity used, and scalability at each time step. For each dataset, we create a sequence of T snapshots, $\{G^1, G^2, \ldots, G^T\}$, where $G^{t+1} = G^t + \Delta G, |\Delta G| = |G|/T, t = 1, 2, \ldots T-1$.

Due to the inability of some algorithms in the experiment to handle large datasets, or lack of dynamic adaptability or heterogeneous adaptability, we preprocessed the datasets according to different situations. For the algorithms that cannot handle big datasets, we sampled the original datasets to allow these algorithms to run on the sample sets. For the static IM algorithms, we removed the temporal duplicate edges of the dataset. To implement homogeneous IM algorithms on a heterogeneous network, we first partitioned the network into several homogeneous sub-networks based on the types of edges. Next, we applied the homogeneous IM algorithm on each sub-network and recorded the corresponding execution time, memory consumption, and influence value after eliminating duplication. Finally, we aggregated these results to obtain the overall performance of the algorithm on the entire heterogeneous network.

According to Reference [22], for algorithm DIM, parameter β was set to 32, w was set to $\beta(n+m)\log n$. According to Reference [8], for algorithm Inf2vec, influence context length L was set to 50, 10 influence context paths were generated by each node, learning rate γ was set to 0.005, *window_size* was set to 4, *skip_size* was set to 4, maximum number of iterations I was set to 10, influence embedding vector dimension d was set to 50, and subnet weight α was set to 0.25. According to Reference [12], algorithm GroupIM used Skip-gram model to obtain the network embedding, in which walk length L was set to 80, number of walks per node was set to 40, learning rate γ was set to 0.005, *window_size* was set to 10, *skip_size* was set to 4, and influence embedding vector dimension d was set to 128, and constant coefficient δ was set to 0.5. For algorithm DHNE-CIM, influence embedding vector dimension d was set to 32.

6.5 Ablation Study

Our algorithm DHNE-CIM uses algorithm DyHATR in the network embedding stage. Algorithm DyHATR has two important components, HAN and RNN. To study the contribution of each component to the modeling capability, we conducted ablation experiments on SX-Super, Higgs-Twitter, and Tencent-Weibo datasets, respectively. We obtained three network embeddings by HAN learning, RNN learning, and the complete model learning. These three network embeddings are used to execute the DHNE-CIM algorithm to achieve the influence values for searching 10 seeds. We labeled the experimental results as HAN-CIM, RNN-CIM, and DHNE-CIM, respectively. Figure 3 reports the ablation experimental results. We can see from Figure 3 that RNN-CIM performed better than HAN-CIM on SX Supper and Tencent Weibo datasets, indicating that RNN component plays a greater role, while HAN component plays a smaller role in DHNE-CIM algorithm on these two networks. On dataset Higgs-Twitter, the HAN component plays a slightly greater role than the RNN component. The influence values of DHNE-CIM algorithm using two components together on the three networks are significantly more than the influence values of DHNE-CIM algorithm using single component. It can be seen that the superposition of two components is necessary for the DHNE-CIM algorithm.

6.6 Results and Discussion

The SX-Supper dataset has strong dynamics and weak heterogeneity. Firstly, we evaluated the influence value, execution time, and memory capacity used of running the five algorithms DHNE-CIM, GroupIM, Inf2vec, DIM, and KSN on the SX-Supper dataset. Table 5 shows their experimental results for searching 10, 50, 100, 150, and 200 seeds on the final snapshot of SX-Supper dataset, with the best performing results highlighted in bold.

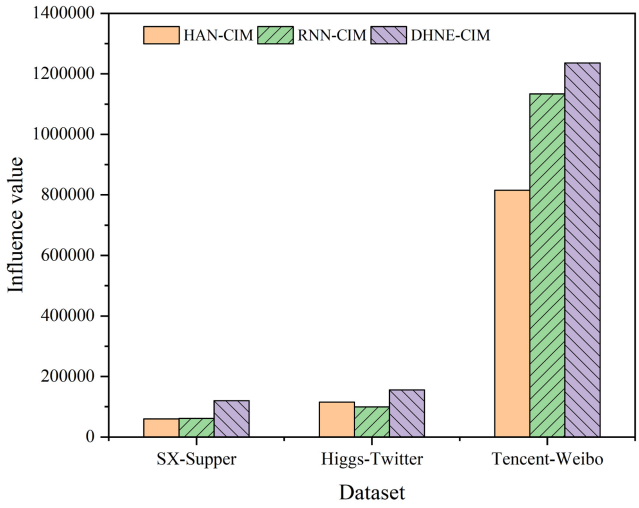

Fig. 3. Ablation study for DHNE-CIM running on three datasets.

From Table 5, we can see that the performance of running the five algorithms on the dataset SX-Supper for a varying number of seeds. The influence value of our proposed algorithm DHNE-CIM was significantly higher than that of algorithms GroupIM and Inf2vec which also use network embedding. This is because GroupIM and Inf2vec cannot learn the dynamic features of topology. Similarly, the influence value of algorithm DHNE-CIM was also significantly higher than that of static heterogeneous algorithm KSN. The influence of algorithm DHNE-CIM was higher than that of algorithm DIM. DIM is an IM algorithm that relies on a dynamic homogeneous model and lacks heterogeneous topology features.

In terms of running speed, the experimental results show that the algorithm DHNE-CIM was the fastest among five algorithms. DHNE-CIM, GroupIM, and Inf2vec are all based on the network embedding model, in which the propagation features are converted into the form of node feature vectors. Therefore, the execution time of these three algorithms did not increase significantly with increase of the number of searching seeds. However, the two algorithms DIM and KSN are based on the traditional edge probability diffusion model, their computing time are increased significantly as increase of the number of searching seeds. Moreover, both algorithms DHNE-CIM and GroupIM ran faster than algorithm Inf2vec. The reason is that algorithm DHNE-CIM and GroupIM are based on community propagation, but algorithm Inf2vec is based on individual propagation. In addition, we also find that the running speed of DHNE-CIM was faster than that of GroupIM. This is because GroupIM uses an edge-based layer clustering algorithm to discover communities and its execution time is related to the number of edges, while DHNE-CIM uses a node-based clustering algorithm to discover communities and its execution time is related to the number of nodes. The number of edges in social network is usually greater than the number of nodes, so algorithm DHNE-CIM ran faster than algorithm GroupIM.

For memory consumption, the DHNE-CIM used the least memory capacity among the five algorithms, and the algorithms GroupIM and Inf2vec used the second and the third smallest memory capacity. This indicates that the memory consumption of the NE-based IM algorithms was less than that of those IM algorithms based on the traditional edge probability diffusion model, such as DIM and KSN. This is because the memory capacity consumed by the NE-based algorithms was only related to the number of embedding vectors. Algorithm DHNE-CIM calculated IM at community

Table 5. Performance of Running Five Algorithms on the SX-Supper Dataset

Number of seeds	Algorithms	Influence velue	Execution time (s)	Memory capacity used(MB)
10	DHNE-CIM	**120,202.05**	**88.21**	**23.48**
	GroupIM	7,909.78	225.18	128.50
	Inf2vec	127.15	11,430.85	218.57
	DIM	17,277.92	40,214.00	476.30
	KSN	22,588.90	25,339.83	11,520.00
50	DHNE-CIM	**120,202.17**	**89.58**	**23.48**
	GroupIM	8,409.78	225.20	128.50
	Inf2vec	2,000.29	11,430.85	218.57
	DIM	19,531.39	40,556.00	476.30
	KSN	22,927.40	141,789.60	37,826.80
100	DHNE-CIM	**120,202.28**	**124.05**	**23.48**
	GroupIM	9,409.78	226.03	128.50
	Inf2vec	8,754.73	11,430.85	218.57
	DIM	21,017.79	45,046.00	476.30
	KSN	23,139.50	501,666.04	64,929.80
150	DHNE-CIM	**120,202.32**	**154.45**	**23.48**
	GroupIM	10,909.78	226.09	128.50
	Inf2vec	206,05.34	11,430.85	218.57
	DIM	22,105.68	45,069.00	476.30
	KSN	23,289.50	1,063,517.78	118,462.00
200	DHNE-CIM	**120,202.95**	**162.65**	**23.48**
	GroupIM	12,909.78	226.10	128.50
	Inf2vec	37,665.24	11,430.85	218.57
	DIM	22,979.33	45,091.00	476.30
	KSN	23,549.00	1,907,512.68	136,978.00

The best performing results highlighted in bold.

level, and its memory usage was lower than that of algorithm Inf2vec using individual nodes embedding. Although both algorithms DHNE-CIM and GroupIM calculated IM in community level, the memory consumption of clustering algorithm used in GroupIM was more than that of clustering algorithm used in DHNE-CIM. In addition, the memory consumption of running dynamic IM algorithms was less than that of running static IM algorithms. This is because the memory usage of the dynamic IM algorithms is proportional to the amount of network variation. When the network structure updated, the dynamic algorithms only calculated the part of changed, while the static algorithms needed to recalculate the entire network, resulting in more memory consumption. KSN was a static IM algorithm based on traditional edge probability diffusion model. Compared to algorithm KSN, our algorithm DHNE-CIM represented the network heterogeneous information as feature vectors of nodes, and estimated the influence value by the community structure and vector calculation, so its memory usage did not be increased as increase of the number of searching seeds.

Unlike the SX-Supper dataset, the dataset Tencent-Weibo has only heterogeneous duplicated edges but do not have temporal duplicated edges. Table 6 shows the experimental results of running the five IM algorithms on the final snapshot of dataset Tencent-Weibo,with the best performing results highlighted in bold.

Table 6. Performance of Running Five Algorithms on the Dataset Tencent-Weibo

Number of seeds	Algorithms	Influence velue	Execution time (s)	Memory capacity used(MB)
10	DHNE-CIM	**1,236,072.43**	**196.68**	**146.82**
	GroupIM	11,982.92	1,119.07	470.92
	Inf2vec	109.38	210,070.06	932.79
	DIM	2,768.34	1,017,432.00	3,141.41
	KSN	1,819.70	3,306.21	15,620.70
50	DHNE-CIM	**1,450,359.22**	**307.82**	**146.82**
	GroupIM	12,482.92	1,120.07	470.92
	Inf2vec	2,714.74	210,070.06	932.79
	DIM	3,294.17	1,017,561.00	3,141.41
	KSN	2,374.25	22,251.50	15,342.70
100	DHNE-CIM	**1,476,892.54**	**352.95**	**146.82**
	GroupIM	13,482.92	1,119.56	470.92
	Inf2vec	10,279.11	210,070.06	932.79
	DIM	3,748.80	1,017,710.00	3,141.41
	KSN	3,046.57	65,345.16	22,451.60
150	DHNE-CIM	**1,489,682.61**	**377.92**	**146.82**
	GroupIM	14,982.92	1,121.03	470.92
	Inf2vec	22,128.03	210,070.06	932.79
	DIM	4,121.89	1,017,861.00	3,141.41
	KSN	8,482.50	121,567.62	25,955.30
200	DHNE-CIM	**1,495,349.74**	**530.10**	**146.82**
	GroupIM	16,982.92	1,119.45	470.92
	Inf2vec	38,222.19	210,070.06	932.79
	DIM	4,445.94	1,018,044.00	3,141.41
	KSN	15,567.00	197,215.67	29,626.70

The best performing results highlighted in bold.

From Table 6, we can see that for the dataset Tencent-Weibo with strong heterogeneity and weak dynamics, our algorithm DHNE-CIM still had the highest influence value among five algorithms for a varying number of seeds, but the performance of other four algorithms had fluctuated. NE-based IM algorithms GroupIM and Inf2vec, and static heterogeneous IM algorithm KSN had better performance than dynamic homogeneous algorithm DIM. This indicates that our algorithm DHNE-CIM was more robust than other four algorithms.

For the execution time and memory capacity used, we can see from Table 6 that our algorithm DHNE-CIM required the least amount among the five algorithms.

Unlike the previous two datasets SX-Supper and Tencent-Weibo, dataset Higgs-Twitter has a large number of dynamic duplicated edges and heterogeneous duplicated edges. Table 7 shows the experimental results of running the five IM algorithms on the final snapshot of the dataset Higgs-Twitter. Higgs-Twitter,with the best performing results highlighted in bold.

As can be seen in Table 7, for the dataset Higgs-Twitter, our algorithm DHNE-CIM had the best ability to capture network propagation feature, and obtained the largest influence value among the five algorithms. This is because algorithm DHNE-CIM used DHNE to capture the dynamic and heterogeneous characteristics of nodes, and obtain more propagation features than other four algorithms. Algorithm GroupIM could not capture dynamic information and heterogeneous

Table 7. Performance of Running Five Algorithms on the Dataset Higgs-Twitter

Number of seeds	Algorithms	Influence velue	Execution time (s)	Memory capacity used(MB)
10	DHNE-CIM	**155,944.24**	**90.34**	**30.46**
	GroupIM	12,384.96	938.95	120.32
	Inf2vec	117.98	7,699.25	175.72
	DIM	17,248.09	27,366.00	232.16
	KSN	1,520.61	8,376.2	6,074.05
50	DHNE-CIM	**155,944.26**	**116.77**	**30.46**
	GroupIM	12,884.96	939.22	120.32
	Inf2vec	2,227.95	7,699.25	175.72
	DIM	23,657.68	27,420.00	232.16
	KSN	2,099.04	28,312.72	13,603.00
100	DHNE-CIM	**155,944.27**	**133.28**	**30.46**
	GroupIM	13,884.96	940.65	120.32
	Inf2vec	9,005.57	7,699.25	175.72
	DIM	27,773.61	27,507.00	232.16
	KSN	2,499.61	78,239.33	18,744.20
150	DHNE-CIM	**155,944.27**	**129.83**	**30.46**
	GroupIM	15,384.96	941.31	120.32
	Inf2vec	19,177.68	7,699.25	175.72
	DIM	30,354.35	27,637.00	232.16
	KSN	2,831.18	168,699.09	23,321.30
200	DHNE-CIM	**155,944.27**	**151.33**	**30.46**
	GroupIM	17,384.96	943.04	120.32
	Inf2vec	33,688.51	7,699.25	175.72
	DIM	32,232.59	27,790.00	232.16
	KSN	3,109.73	229,815.69	28,167.10

The best performing results highlighted in bold.

information in the network, and it only used community diffusion to keep influence value. Although heterogeneous features are embedded into node feature vectors, the algorithm Inf2vec cannot obtain temporal features in the network. Algorithm DIM obtained the propagation features from dynamic edges, so it had advantage when running on the dataset Higgs-Twitter. However, due to the loss of heterogeneous characteristics, the influence value of DIM was lower than that of the algorithm DHNE-CIM. The influence value of algorithm KSN is the worst because it can only obtain network heterogeneous features but cannot capture network temporal features. Therefore, KSN has lost many diffusion features on the dataset Higgs-Twitter.

We can also see that for the dataset Higgs-Twitter, algorithm DHNE-CIM required the shortest execution time among the five algorithms. Although algorithm DHNE-CIM extracted more propagation feature types from the network than the other four algorithms, its execution speed was the fastest. This is because DHNE-CIM completed dynamic heterogeneous features extraction from the network in embedding processing stage, and used community propagation and vector-calculation-based influence estimation method in IM calculation stage. Therefore, the efficiency of algorithm DHNE-CIM was greatly improved.

In addition, for the dataset Higgs-Twitter, the memory consumption of algorithm DHNE-CIM was less than that of the other four algorithms.

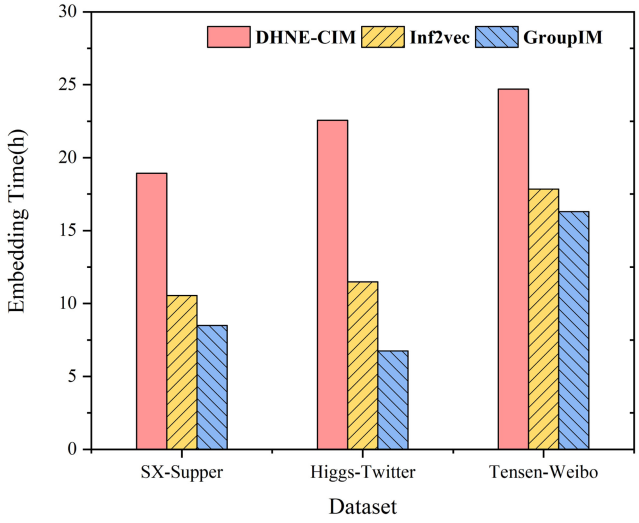

Fig. 4. Required embedding time of running algorithms DHNE-CIM, Inf2vec and GroupIM on three datasets.

In summary, algorithm DHNE-CIM achieved better performance and robustness on three datasets than other four algorithms on the whole. Algorithm DHNE-CIM can be used for IM computing of large-scale DHNS with low computing cost in practical applications.

Algorithms DHNE-CIM, Inf2vec, and GroupIM all need network embedding. Figure 4 shows their embedding time. From Figure 4, we can see that the required embedding time of algorithm DHNE-CIM was longest, Inf2vec ranked the second, and algorithm GroupIM required the shortest embedding time. The reason is that the algorithm DHNE-CIM needed to learn dynamic evolution features and heterogeneous features from network snapshot sequence G^1 to G^T, while algorithm Inf2vec just learned the heterogeneous features of G^T, and algorithm GroupIM only learned the static homogeneous relationship between nodes without the differences in edge types.

DHNE-CIM and GroupIM are both NE-based community diffusion IM algorithms. In reference [12], algorithm GroupIM uses the Skip-gram model to learn network embeddings. To further compare the performance of two similar IM algorithms, we use the network embedding obtained from the DyHATR algorithm as input for GroupIM. Then, we compare the influence value produced by DHNE-CIM and DyHATR-GroupIM. The comparison results on three datasets are shown in Figure 5. The influence value of algorithm DHNE-CIM is significantly higher than that of algorithm GroupIM. It can be seen that algorithm DHNE-CIM performs better than algorithm GroupIM. The main reason is that algorithm DHNE-CIM used the diffusion proximity from seeds to each node in the community to accurately calculate the influence probability within the community, while algorithm GroupIM approximately treated the influence probability of all nodes in the community as the same.

Finally, we evaluated the scalability of five algorithms on three datasets, respectively. We run five algorithms on 10 consecutive snapshots of each dataset to search for 100 seeds and record required time on snapshot G^1, G^5 and G^{10}. The purpose was to observe which IM algorithm had better adaptability when the network scale increased from G^1 to G^{10}, where $G^{t+1} = G^t + \Delta G, |\Delta G| = |G|/10, t = 1, 2, \ldots, 9$. The experimental results are shown in Figure 6.

As can be seen from Figures 6(a)–6(c), the maximal increase in execution time of algorithm DHNE-CIM was within 200 seconds, while the maximal increase in execution time of algorithms DIM, KSN, Inf2vec, and GroupIM were about 7.9 hours, 44 hours, 2.1 hours, and 940 seconds

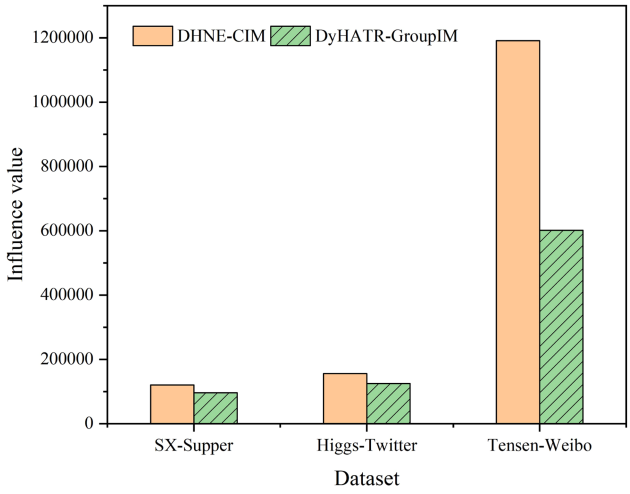

Fig. 5. Influence values of running the algorithm DHNE-CIM and GroupIM.

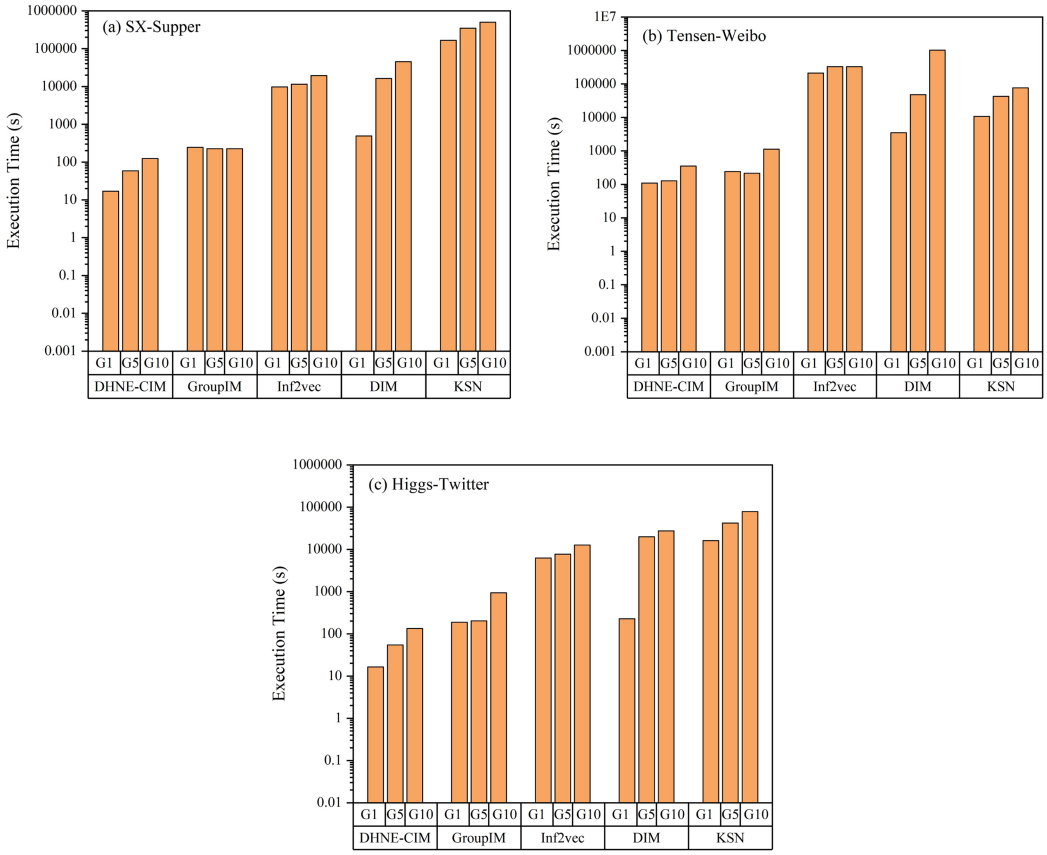

Fig. 6. Scalability for five IM algorithms running on three datasets.

respectively. Both algorithms Inf2vec and GroupIM establish the node embedding matrix based on the maximal scale of G^{10} to adapt to growth from G^1 to G^{10}. Therefore, even if the network size was very small, their running time are also very long. It illustrates that the scalability of algorithm DHNE-CIM is the best among the five algorithms.

7 CONCLUSION

In this article, we proposed a DHNE-based community diffusion model and influence maximization algorithm. Its main idea is to obtain the propagation feature vector matrix of nodes by performing network embedding algorithm to solve the feature representation problem in large-scale dynamic heterogeneous social networks, and utilize k-means algorithm to identify communities and search seeds and compute influence values in large-scale dynamic heterogeneous social networks by using community structure and vector calculations. The experiments show that our algorithm performs better on real social networks and has better robustness and scalability than existing algorithms.

In practical applications, users often navigate multiple platforms simultaneously in social networks. This means that social networks have complex multiplex structure. However, this structure poses a challenge for influence maximization, as the topological structures and information expressions differ between networks. In future research, there are several challenges to overcome, including identifying overlapping users, defining the diffusion model, and calculating the total influence value of seed set in multiplex dynamic heterogeneous networks.

ACKNOWLEDGMENTS

Thanks to Naoto Ohsaka for assistance with providing source code of algorithm DIM. Thanks for the editor and anonymous reviewers, constructive comments to help us to improve the manuscript.

REFERENCES

[1] C. Aggarwal, S. Lin, and P. Yu. 2012. On influential node discovery in dynamic social networks. In *Proceedings of the 12th SIAM International Conference on Data Mining* (2012), 636–647.

[2] V. Belak, S. Lam, and C. Hayes. 2012. Towards maximising cross-community information diffusion. In *Proceedings of the 2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining* (2012), 171–178.

[3] X. D. Chen, G. J. Song, X. R. He, and K. Xie. 2015. On influential nodes tracking in dynamic social networks. In *Proceedings of the 15th SIAM International Conference on Data Mining* (2015), 613–621.

[4] K. Cho, B. Van Merrienboer, C. Gulcehre, D. Ba Hdanau, F. Bougares, H. Schwenk, and Y. Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)* (2014), 1724–1734.

[5] X. Deng, F. Long, B. Li, D. Cao, and Y. Pan. 2020. An influence model based on heterogeneous online social network for influence maximization. *IEEE Transactions on Network Science and Engineering* 7, 2 (2020), 737–749.

[6] P. Domingos and M. Richardson. 2001. Mining the network value of customers. In *Proceedings of the 7th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2001), 57–66.

[7] M. Eftekhar, Y. Ganjali, and N. Koudas. 2013. Information cascade at group scale. In *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2013), 401–409.

[8] Shanshan Feng, Gao Cong, Arijit Khan, Xiucheng Li, Yong Liu, Yeow Meng Chee, and Ieee. 2018. Inf2vec: Latent representation model for social influence embedding. In *Proceedings of the IEEE 34th International Conference on Data Engineering* (2018), 941–952. https://doi.org/10.1109/icde.2018.00089

[9] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. 1978. An analysis of approximations for maximizing submodular set functions - 1. *Mathematical Programming* 14 (1978), 265–294.

[10] A. Goyal, W. Lu, and L. Lakshmanan. 2011. CELF++: Optimizing the greedy algorithm for influence maximization in social networks. In *Proceedings of the 20th International Conference Companion on World Wide Web* (2011), 47–48.

[11] N. Hafiene, W. Karoui, and L. B. Romdhane. 2020. An incremental approach to update influential nodes in dynamic social networks. *Procedia Computer Science* 176 (2020), 781–790.

[12] Yaoxuan Ji, Li Pan, and Peng Wu. 2019. Influence maximization on large-scale networks with a group-based method via network embedding. In *Proceedings of the 2019 IEEE Fourth International Conference on Data Science in Cyberspace (DSC)* (2019), 176–182.

[13] D. Kempe, J. Kleinberg, and E. Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2003), 137–146.
[14] Mama Kermani, Sff Ardestani, A. Aliahmadi, and F. Barzinpour. 2017. A novel game theoretic approach for modeling competitive information diffusion in social networks with heterogeneous nodes. *Physica A: Statistical Mechanics and its Applications* 466 (2017), 570–582.
[15] A. Kuhnle, M. A. Alim, X. Li, H. Zhang, and M. T. Thai. 2018. Multiplex influence maximization in online social networks with heterogeneous diffusion models. *IEEE Transactions on Computational Social Systems* 5 (2018), 1–12.
[16] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. Vanbriesen, and N. Glance. 2007. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2007), 420–429.
[17] W. Li, K. Zhong, J. Wang, and D. Chen. 2020. A dynamic algorithm based on cohesive entropy for influence maximization in social networks. *Expert Systems with Applications* 169, 2 (2020), 1–12.
[18] W. H. Li, Q. Bai, and M. J. Zhang. 2019. SIMiner: A stigmergy-based model for mining influential nodes in dynamic social networks. *IEEE Transactions on Big Data* 5, 2 (2019), 223–237.
[19] S. Manchanda, A. Mittal, A. Dhawan, S. Medya, S. Ranu, and A. Singh. 2020. GCOMB: Learning budget-constrained combinatorial algorithms over billion-sized graphs. In *Proceedings of the 2020 Annual Conference on Neural Information Processing Systems* (2020). DOI : https://doi.org/10.48550/arXiv.1903.03332
[20] Y. H. Meng, Y. H. Yi, F. Xiong, and C. X. Pei. 2019. T x one Hop approach for dynamic influence maximization problem. *Physica a-Statistical Mechanics and Its Applications* 515 (2019), 575–586.
[21] H. Y. Min, J. X. Cao, T. F. Yuan, and B. Liu. 2020. Topic based time-sensitive influence maximization in online social networks. *World Wide Web-Internet and Web Information Systems* 23, 3 (2020), 1831–1859.
[22] N. Ohsaka, T. Akiba, Y. Yoshida, and K. Kawarabayashi. 2016. Dynamic influence analysis in evolving networks. *Proceedings of the VLDB Endowment* 9, 12 (2016), 1077–1088.
[23] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge Discovery and Data Mining* (2014), 701–710.
[24] Xi Qin, Cheng Zhong, and Qingshan Yang. 2021. An influence maximization algorithm based on community-topic features for dynamic social networks. *IEEE Transactions on Network Science and Engineering* 9, 2 (2021), 608–621.
[25] M. Richardson and P. Domingos. 2002. Mining knowledge-sharing sites for viral marketing. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2002), 61–70.
[26] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. 1986. Learning representations by back propagating errors. *Nature* 323, 6088 (1986), 533–536.
[27] J. Shang, S. Zhou, X. Li, L. Liu, and H. Wu. 2016. CoFIM: A community-based framework for influence maximization on large-scale networks. *Knowledge-Based Systems* 117 (2016), 88–100.
[28] S. S. Singh, K. Singh, A. Kumar, and B. Biswas. 2019. MIM2: Multiple influence maximization across multiple social networks. *Physica A: Statistical Mechanics and its Applications* 526 (2019), 1–22.
[29] Y. Tang, Y. Shi, and X. Xiao. 2015. Influence maximization in near-linear time. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2015), 1539–1554.
[30] Y. Tang, X. Xiao, and Y. Shi. 2014. Influence maximization: Near-optimal time complexity meets practical efficiency. In *Proceedings of the ACM SIGMOD International Conference on Management of Data* (2014), 75–86.
[31] G. Tong, W. Wu, S. Tang, and D. Z. Du. 2017. Adaptive influence maximization in dynamic social networks. *IEEE/ACM Transactions on Networking* 25, 1 (2017), 112–125.
[32] Rui Wang, Yongkun Li, Shuai Lin, and Hong Xie. 2021. On modeling influence maximization in social activity networks under general settings. *ACM Transactions on Knowledge Discovery from Data* 15, 6 (2021), 1–28.
[33] H. Xue, L. Yang, W. Jiang, Y. Wei, and Y. Lin. 2020. Modeling dynamic heterogeneous network for link prediction using hierarchical attention with temporal RNN. *ECML PKDD 2020: Machine Learning and Knowledge Discovery in Databases* 12457 (2020), 282–298.
[34] Z. Yang, D. Yang, C. Dyer, X. He, and E. Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2016), 1480–1489.
[35] S. Yerasani, S. Tripathi, M. Sarma, and M. K. Tiwari. 2020. Exploring the effect of dynamic seed activation in social networks. *International Journal of Information Management* 51 (2020), 1–7.
[36] Y. Yin, L. X. Ji, J. P. Zhang, and Y. L. Pei. 2019. DHNE: Network representation learning method for dynamic heterogeneous networks. *IEEE Access* 7 (2019), 134782–134792.

Received 23 June 2022; revised 10 December 2022; accepted 5 April 2023