Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

Learning Multiple Network Embeddings for Social Influence Prediction ⋆

Feng Wang [*,**,***], **Jinhua She** [***], **Yasuhiro Ohyama** [***], **Min Wu** [*,**,†]

[*] *School of Automation, China University of Geosciences, Wuhan 430074, China*
[**] *Hubei Key Laboratory of Advanced Control and Intelligent Automation for Complex Systems, Wuhan 430074, China (email: {wangfeng, wumin}@cug.edu.cn)*
[***] *School of Engineering, Tokyo University of Technology, Tokyo 192-0982, Japan (email: {she, ohyama}@stf.teu.ac.jp)*

Abstract: How to effectively predict social influence is an essential issue in social network analysis. Almost all reported methods for social influence prediction are mainly concerned with estimating influence probabilities for each linking edge. However, all of this past work cannot accurately predict influence probabilities for all edges due to the problem of data sparsity. Unlike conventional approaches, this work focuses on exploring a cross problem for multiple network embeddings and social influence prediction. This study developed a new end-to-end approach, Multi-Influor, that learns multiple influence vectors for each user in social networks, instead of estimating influence probabilities for each edge. The multiple network embeddings consider multi-dimensional influence factors that incorporate pairwise node interactions, network structures, and global similarity comparisons. Moreover, this study solves the problem of influence evaluation caused by sparse observations. Extensive comparisons based on large-scale datasets indicate that the Multi-Influor approach outperforms several state-of-the-art baselines, and the experimental results demonstrate that the Multi-Influor approach is more practical on real-world social networks.

Keywords: Multiple Network Embeddings, Representation Learning, Online Social Networks, Neural Network, Social Influence Prediction.

1. INTRODUCTION

The rapid development of online social networks (OSNs) lays a foundation for spreading users' ideas and messages. The related work of social network analysis has found that users' friends or neighbors greatly affect these users' behaviors or opinions from a perspective of psychology [Aslay, *et al.* (2018)]. Thus, this study defines social influence as a user's ability that users' actions (behaviors or opinions) affect their neighbors in OSNs. The extensive researches in social influence analysis pave the way for various applications, such as resource recommendation [Zhang, *et al.* (2018)], community detection [Al-Garadi, *et al.* (2018)], and viral marketing [Yang, *et al.* (2017)].

One essential problem in social influence analysis is to estimate influence probabilities based on action logs, and extensive works have been done on this problem [Pal, *et al.* (2014); Aslay, *et al.* (2018); Sun, *et al.* (2018)]. Almost all previously related work aims at calculating influence parameters for each edge in OSNs. However, since the problem of sparse observations for influence spreads, the previous works cannot accurately learn edge parameters to predict influence probabilities, especially for edges without sufficient diffusion data. Moreover, these conventional edge-based approaches cannot consider other influence factors (such as network structure and preference similarity) or detect hidden influence relationships between users (high-order influence spreads). Thus they perform poorly on the predictions of influence probabilities.

This work focuses on exploring a cross problem for multiple network embeddings and social influence prediction. Inspired by the research of network representation learning [Bai, *et al.* (2019); Grover, *et al.* (2018)], this study directly learns multiple latent vectors for each user to capture social influences, instead of estimating influence probabilities for each edge. Moreover, this work alleviates the challenge of sparse observation data via latent vectors. To our knowledge, this may be the first study of social influence prediction that jointly captures influence spreads and multiple network embeddings.

⋆ This work was supported in part by China Postdoctoral Science Foundation under Grant 2019M662740, the National Natural Science Foundation of China under Grant 61733016, the 111 Project of China under Grant B17040, and the Fundamental Research Funds for the Central Universities under Grant CUGCJ1812. The first author is an overseas researcher under Postdoctoral Fellowship of Japan Society for the Promotion of Science (JSPS), and his JSPS Fellowship ID is P19704.
† Corresponding author: Min Wu (email: wumin@cug.edu.cn)

To address the above challenges, this study developed a new end-to-end approach, **Multi**ple **Influ**ence-to-vect**or** (**Multi-Influor**), that learns multiple influence vectors for each user in social networks. The first key step to predicting social influence is how to generate multiple influence learning contexts. This work devised a new method of generating multiple influence contexts, which includes node-level, subnetwork-level, and global graph-level influence contexts. Then, the Multi-Influor method learns node-level and graph-level latent vectors based on the generated contexts (details in Section 3). Therefore, the multiple network embeddings incorporate multi-dimensional influence factors for pairwise node interactions, network structures, and global similarity comparisons.

In this study, we conducted extensive comparisons based on four large-scale datasets. The comparison results illustrate that the Multi-Influor approach outperforms several state-of-the-art baselines. Moreover, the experimental results demonstrate that the Multi-Influor approach is more practical on real-world social networks.

2. PROBLEM STATEMENT

2.1 Preliminaries

In this study, we model each social network as a graph $G = (V, E)$, where V denotes a user set and E denotes a set of linking edges between users. The linking edge (u, v) denotes that user u is a neighbor of user v. In addition, this study obtains diffusion episodes from action logs. The action logs contain sets of tuples in the form of (u, i, t_u^i) that denotes user u performed an action related to item i at time t_u^i. Each item i (such as information, idea and opinion) corresponds to one diffusion episode $D_i = \{(u, t_u^i), \cdots, (v, t_v^i)\}$, which denotes users in D_i adopt to perform action i in a chronological order.

Definition 2.1. (**User Social Action**). A binary action state for user u, $s_u^i \in \{0, 1\}$, is calculated based on action logs A, where $s_u^i = 0$ denotes user u has not performed the action i at time t_u^i, and $s_u^i = 1$ denotes user u has performed the action i before or at time t_u^i.

Definition 2.2. (**Social Influence Spread**). Based on a social network $G = (V, E)$ and diffusion episodes D_i, a social influence spreads from user u to user v if it satisfies: (1) user $u \in V$ and $v \in V$; (2) edge $(u, v) \in E$; (3) action time $t_u^i < t_v^i$.

2.2 Problem Definition

This study aims to transfer each user's representation into a low-dimensional latent vector for social influence prediction. In general, this study attempts to learn multiple representations for $|V|$ users based on a given social network and action logs, which are used to evaluate the influence probabilities for $|E|$ edges.

Multiple Network Embeddings Problem. Based on a social network $G = (V, E)$ and action logs, this work aims to learn multiple social influence representations for each user: node-level vector Vec_u and graph-level vectors $I_u, S_u, b_u, \hat{b_u}$.

3. THE DEVELOPED METHOD: MULTI-INFLUOR

In this section, we proceed to detail the newly developed Multiple Influence-to-vector (Multi-Influor) method, which is an end-to-end neural network-based method that aims at addressing the challenges mentioned in Section 1. The framework of Multi-Influor (Fig. 1) contains three parts. First, this work provides how to generate multiple influence contexts in stage 1. Second, based on the generated influence contexts, this paper presents a learning approach for multiple influence representations in stage 2. Finally, this study feeds the combined multiple vectors into a fully connected neural network to predict influence probabilities in stage 3.

3.1 Stage 1: Multiple Influence Contexts Generating

In this study, we introduce a method of generating multiple influence contexts, which includes node-level, subnetwork-level, and graph-level influence contexts. These influence contexts focus on multi-dimensional influence factors for pairwise node interactions, network structures, and global similarity comparisons.

Definition 3.1. (**Influence Context**). Based on a social network $G = (V, E)$ and action logs, this study defines an influence context for user u as a set of users who are probably affected by user u.

Node-level influence context Based on a social network and a given user u, the straightforward way to generate a node-level influence context for user u is to adopt the breadth-first search (BFS) method that starts from user u. Then, r-neighbours $N_u^r = \{v : d(u, v) \leq r\}$ can be acquired, where $d(u, v)$ is the shortest path between user u and v. This work applies a sub-network induced by N_u^r to establishing a network of r-ego user u. To facilitate the neural network learning in network embeddings, we suppose to sample a fixed-size sub-network. In addition, this is easy to solve the problem of different sampling sizes.

According to network structures and edge weights estimated by diffusion episodes D_i, this study performs random walks on the network of r-ego user u. A random walk iteratively traverses to neighbors with different probabilities and stops running until it collects a fixed-size of users (L_N). Then, this work regards the sampled network as a node-level influence context for user u, C_u^{ni}, that can embed pairwise nodes' interaction information into user u's latent influence vector.

Subnetwork-level influence context This work builds an influence spread network to incorporate subnetwork-level influence (high-order influence diffusion) information into latent influence vectors, which illustrates influence diffusions for each item in social networks (as shown on the left part of Fig. 1).

Definition 3.2. (**Influence Spread Network**). Based on a social network $G = (V, E)$ and action logs, we define an influence spread network as $G_i = (V_i, E_i)$ if it satisfies: (1) $V_i \subset V$ and $E_i \subset E$; (2) each edge (u, v) denotes a social influence spreads from user u to v.

The Multi-Influor method captures high-order diffusion information via random walk processes on influence spread

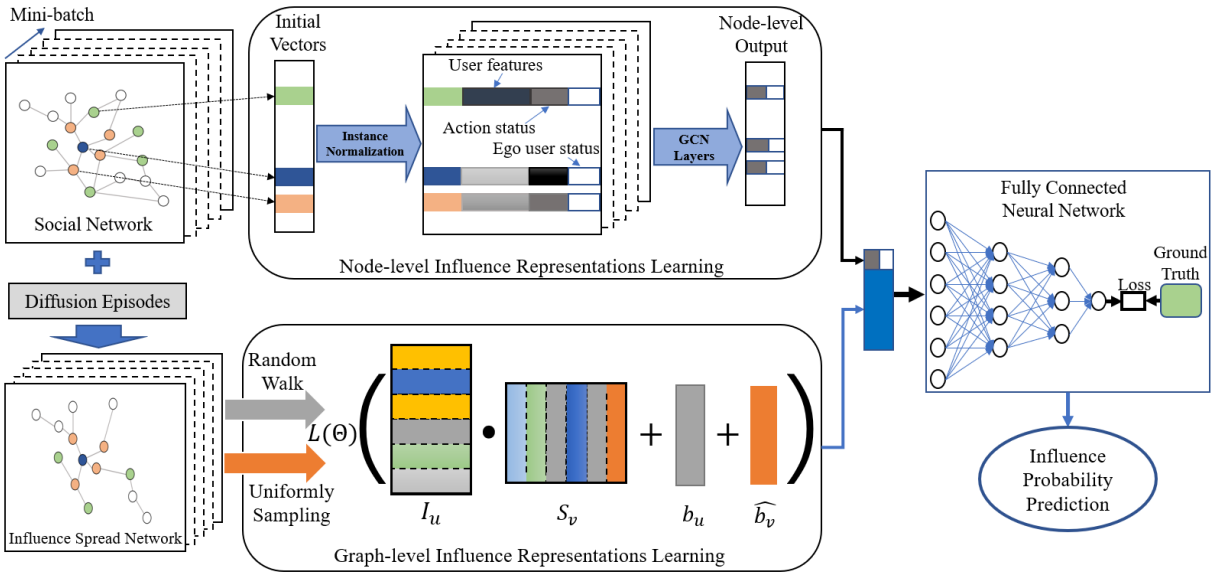


Fig. 1. An overview illustration of Multi-Influor. The top part denotes data flows for node-level influence representations learning. The bottom part denotes data flows for graph-level influence representations learning.

networks, which incorporates structure information into subnetwork-level embeddings. Moreover, this approach can solve the problem of influence estimating caused by sparse propagation data.

This study generates user u's subnetwork-level influence context, C_u^{si}, based on the influence spread network G_i. We apply random walks with restarts to simulating users' influence diffusions. The random walk process starts from user u and randomly selects one neighbor of user u to traverse. Next, based on the currently activated users, we randomly select one neighbor of these users to traverse in the next sampling step. In addition, the process is probable to back to user u at each step. This work uses a length threshold (L_S) to limit the size of C_u^{si} and stops sampling when the threshold L_S is reached.

Global graph-level influence context This study further incorporates users' preference similarities in latent influence vectors by using graph-level influence contexts. Given an influence spread network $G_i = (V_i, E_i)$, users perform the same actions indicates that these users share the same interest.

To capture the similarity of users' interests, the Multi-Influor method uniformly samples a set of users (L_G) by a given user u in the influence spread network G_i. Note that the samples of similar users denote the global graph-level influence context C_u^{gi}.

Multiple influence contexts generating algorithm Based on a social network G and a given user u, this work aims to generate multiple influence contexts $C_u^i = C_u^{ni} + C_u^{si} + C_u^{gi}$. To balance the contributions of these multi-level contexts, we suppose the component weights (α, β, γ) are all in the range of 0 to 1, and $\alpha + \beta + \gamma = 1$.

First, this work generates the node-level influence context ($L_N = \alpha \cdot L$) via random walks in the social network G. Then, the subnetwork-level influence context ($L_S = \beta \cdot L$) can be generated based on the influence spread network G_i. Finally, this study uniformly samples the users in V_i, and the length of the graph-level influence context $L_G = \gamma \cdot L$. These multiple influence contexts contain three types of sampled data. The time complexity of multiple influence contexts generating algorithm is $O(L)$, where L denotes the total length threshold for influence contexts ($L = L_N + L_S + L_G$).

3.2 Stage 2: Multiple Network Embeddings

The stage of multiple network embeddings provides an effective approach that simulates relationships between users and their multiple influence contexts. Inspired by the related work in the field of natural language processing, we apply a modified word2vec technique to learning multiple influence representations for each user in social networks [Grover, *et al.* (2018)].

Node-level influence representation learning This study develops a neural network-based approach that learns node-level influence vectors. Based on the node-level influence contexts, this work aims to estimate the probability of affecting user v conditional on his/her r-ego network G_v^r and action state of his/her r-neighbors S_v^r. We formulate the method of predicting the influence probability for user v after a given time interval Δt as

$$Pr(S_v^{t+\Delta t} | G_v^r, S_v^r).$$

To calculate node-level influence probabilities, this work formulates the node-level influence prediction as a binary graph classification problem, which can be solved by maximizing the objective function as

$$O(\Theta) = \prod_{(u,v) \in E} Pr(S_v^{t+\Delta t} | G_v^r, S_v^r). \quad (1)$$

Based on the node-level influence contexts, all users learn an embedding matrix $X \in \mathbb{R}^{d \times n}$, each column corresponding to the representation of a user in the network. Each user u_i maps to his/her d-dimensional vector $x_{u_i} \in \mathbb{R}^d$ using a pre-trained embedding layer. Then, this work adopts

an instance normalization technique [Huang, *et al.* (2017)] to calculate the instance normalization for each embedding dimension d as

$$y_{u_i} = \frac{x_{u_i} - \mu_i}{\sqrt{\sigma_i^2 - \epsilon}} \tag{2}$$

where μ_i is the mean of vectors, σ_i is the variance of vectors, and ϵ is a small number for numerical stability.

With a node-level embedding for each user, this work also devises to incorporate various user-specific features into the input layer of the graph convolutional network (GCN), such as content features and demographic features [Qiu, *et al.* (2018)]. Aside from instance normalization vectors, this work also considers a binary variable to indicate users' action states.

In the neural network learning, we stack multiple GCN layers to conduct nonlinear transformations [Narayan, *et al.* (2018)]. The input of each GCN layer is a vertex feature matrix H, and the output H' is calculated as

$$H' = GCN(H) = g(A(G)HW^T + b) \tag{3}$$

where the notation $A(G)$ denotes a $n \times n$ matrix that captures structural information for the network G, and W and b are model parameters.

Finally, this work supposes the output layer provides a two-dimension representation (Vec_u) for each user. We use the vector of user u to predict the ego user's state and the influence probability of user u. We compare the prediction results with ground truth data to optimize the objective function (Eq. (1)). Therefore, we learn a node-level influence vector for each user.

Graph-level influence representation learning To capture graph-level influence information, this work uses a skip-gram method to generate the graph-level context for a given user u [Krishna, *et al.* (2019)]. The key step in graph-level influence embedding is to evaluate the probability of observing the graph-level influence context, $Pr(C_u^{si+gi}|u)$, including the contexts of subnetwork-level and global graph-level influence ($C_u^{si+gi} = C_u^{si} + C_u^{gi}$). This work calculates the probability $Pr(C_u^{si+gi}|u)$ by using the independent probability $Pr(v|u)$ that influences propagate from user u to v, and the notation v denotes a user in the graph-level context C_u^{si+gi}.

$$Pr(C_u^{si+gi}|u) = \prod_{v \in C_u^{si+gi}} Pr(v|u) \tag{4}$$

This work generates a list of (u, C_u^{si+gi}) tuples for each diffusion episode, which is denoted as T_{D_i}. The whole influence context tuples, T, contains all the observed episodes D_i in action logs A. This work attempts to maximize the log-probability of $Pr(C_u^{si+gi}|u)$. Thus, this study defines the loss function of the graph-level influence representation learning as

$$L(\Theta) = -\sum_{T_{D_i} \in T} \sum_{(u, C_u^{si+gi}) \in T_{D_i}} \sum_{v \in C_u^{si+gi}} \log Pr(v|u). \tag{5}$$

Next, this work calculates the loss function by evaluating the probabilities that influences propagate from user u to v. The Multi-Influor method calculates the probability

Algorithm 1 Multi-Influor

Input: A social network $G = (V, E)$, diffusion episodes D_i, component weights α, β, γ, a learning rate η and the dimension of influence vector k.

Output: Multiple influence representations for each user $u \in V$.

1: Initialize vectors I_u and S_v with uniform distribution $[-1/k, 1/k]$, $Vec_u \leftarrow 0$, $b_u \leftarrow 0$, $\hat{b_v} \leftarrow 0$, $T \leftarrow \emptyset$.
2: *Multiple influence contexts generating*:
3: **for** $D_i \in A$ **do**
4: Generate a node-level influence context C_u^{ni};
5: Extract an influence spread network $G_i = (V_i, E_i)$.
6: **for** each user $u \in V_i$ **do**
7: Generate C_u^{si} and C_u^{gi};
8: Insert (u, C_u^i) into T_{D_i};
9: Insert T_{D_i} into T.
10: *Multiple influence representations learning*:
11: **for** $T_{D_i} \in T$ **do**
12: **for** $(u, C_u^i) \in T_{D_i}$ **do**
13: **for** $v \in C_u^i$ **do**
14: Node-level influence vector: Vec_u;
15: Graph-level influence vectors: I_u, S_v, b_u, $\hat{b_v}$;
16: Update all parameters until convergence.
17: **return** $Vec_u, I_u, S_u, b_u, \hat{b_u}$.

$Pr(v|u)$ via the inner product of users' latent vectors $I_u \cdot S_v$, where I_u denotes the influence ability vector for user u and S_v denotes the susceptibility vector for user v. Moreover, this work uses b_u and $\hat{b_v}$ to reflect the bias of influence vector for user u and the bias of susceptibility vector for user v, respectively. Therefore, this work formulates the probability that user u influences user v as a softmax function (Eq. 6).

$$Pr(v|u) = e^{(I_u \cdot S_v + b_u + \hat{b_v})} / sum(u) \tag{6}$$

where $sum(u) = \sum_{w \in V_i} e^{(I_u \cdot S_w + b_u + \hat{b_w})}$ is a normalization term.

To alleviate the issue of computation efficiency, the negative sampling can be used to compute the softmax function [Peng, *et al.* (2017)]. Then, this study uses the method of stochastic gradient descent to learn all parameters. During each step, this work updates parameters Θ via gradient calculations as

$$\Theta \leftarrow \Theta + \eta \frac{\partial}{\partial \Theta} (\log (Pr(v|u))) \tag{7}$$

where η denotes a learning rate and $\frac{\partial}{\partial \Theta}$ denotes the gradient of parameters Θ.

Algorithm 1 summarizes the Multi-Influor method, which contains two essential parts: multiple influence contexts generating (lines 2-9) and multiple influence representations learning (lines 10-16).

3.3 Stage 3: Influence Probability Predicting

After obtaining multiple influence representations for each user, this work uses a multi-layer fully connected neural network to gradually reduce the vector's dimension for each user. In the end, we predict the influence probability, $\hat{s_{uv}}$, that influences propagate from user u to v. This study

compares the prediction results with ground truth data, and optimizes the log-likelihood loss function as

$$L_\Theta = -\sum_{(u,v)\in G} \log(Pr(\hat{s_{uv}}|C_u^i)). \quad (8)$$

To sum up, the Multi-Influor method incorporates the fine-grained influence relationships that are captured by the node-level embedding and the coarse influence information that is captured by the graph-level embedding, which provides a thorough view of influence probability estimating to the prediction approach.

4. EVALUATION

4.1 Experimental Setup

This study conducted computational experiments based on four real-world social networks, which are widely used in the research of influence analysis. Table 1 lists the basic statistics of these datasets, including Sina Weibo[1], Epinions [Goyal, *et al.* (2011)], WikiVote [Zhou, *et al.* (2013)], and NetHEPT [Pal, *et al.* (2014)].

Table 1. Basic Statistics of Real-world OSNs

Dataset	Nodes	Edges	Avg. Degree	Contexts
Sina Weibo	63641	1391718	12.9	270K
Epinions	131828	841372	6.1	352K
WikiVote	7115	103689	10.5	11.9K
NetHEPT	27770	352807	12.2	34K

In this study, we conducted the comparisons of the Multi-Influor approach and several baselines (SVM [Al-Garadi, *et al.* (2018)], Emb-IC [Feng, *et al.* (2018)], Node2vec [Grover, *et al.* (2018)], Inf2vec [Feng, *et al.* (2018)]).

This study randomly selected 60%, 20%, 20% instances of sampling data for training, validation, and testing, respectively. The size of a mini-batch was set to be 512 across four datasets. The Multi-Influor method performed random walks with a restart probability of 0.75 in the steps of contexts sampling. Furthermore, this work set the dimension of initial vectors to be 64 in the node-level embedding, and to be 60 in the graph-level embedding. Each initial vector in the input layers was pre-trained via the autoencoder technique.

In the step of neural network learning, this work supposed the learning rate $\eta = 0.005$ and the influence vector's dimension $k = 62$. According to the empirical study on tuning set [Bai, *et al.* (2019)], the total length threshold for influence context L was set to 160, and the component weights were set as $\alpha = 0.4, \beta = 0.3$, and $\gamma = 0.3$.

4.2 Evaluation Metrics

Prediction performance comparison. The task of influence diffusion prediction focuses on predicting predict probabilities of whether users will be influenced by seed users. This study evaluated the predictive performance for different methods in terms of precision (Prec.), recall (Rec.), F1-measure, and area under curve (AUC).

Influence spread comparison. This work used the prediction results to detect seeds and solve the influence maximization problem. We compared the seeds' influence, which is demonstrated by the range of influence spread. We conducted these comparisons on different datasets and used the independent cascade (IC) model to simulate influence spread processes on OSNs [Aslay, *et al.* (2018)].

[1] http://open.weibo.com/

4.3 Prediction Performance Comparison

This subsection presents the comparisons of diffusion probability predictions based on different OSNs (Table 2). In general, the Multi-Influor approach achieves significantly better performance than the conventional methods (SVM and Emb-IC) in the datasets of Sina Weibo, Epinions, and NetHEPT. The methods of Multi-Influor and Inf2vec perform best, but the Multi-Influor method performs better than the Inf2vec method in diffusion probability predictions. These may because the Multi-Influor method considers multiple network embeddings that incorporate more influence factors.

Table 2. Diffusion Prediction Comparisons

Data	Model	Prec.	Rec.	F1	AUC
Sina Weibo	SVM	32.27	71.23	46.23	66.79
	Emb-IC	33.41	65.73	42.63	66.49
	Node2vec	30.17	63.28	41.91	64.21
	Inf2vec	30.25	64.12	43.29	65.19
	Multi-Influor	**49.33**	**78.23**	**53.31**	**73.67**
Epinions	SVM	53.21	61.44	56.62	56.12
	Emb-IC	43.66	65.12	52.37	62.24
	Node2vec	50.22	65.37	51.23	69.24
	Inf2vec	33.17	63.28	41.91	64.21
	Multi-Influor	**60.17**	**71.32**	**59.93**	**76.15**
WikiVote	SVM	22.17	31.32	40.91	46.17
	Emb-IC	23.98	35.44	42.31	**46.41**
	Node2vec	30.32	33.82	42.36	44.67
	Inf2vec	30.17	33.28	41.91	44.21
	Multi-Influor	**33.34**	**39.21**	**43.75**	43.23
NetHEPT	SVM	52.03	61.63	52.92	44.29
	Emb-IC	49.21	64.67	52.21	62.43
	Node2vec	50.87	63.05	51.91	65.22
	Inf2vec	56.17	63.28	51.94	64.21
	Multi-Influor	**59.36**	**69.24**	**58.73**	**69.23**

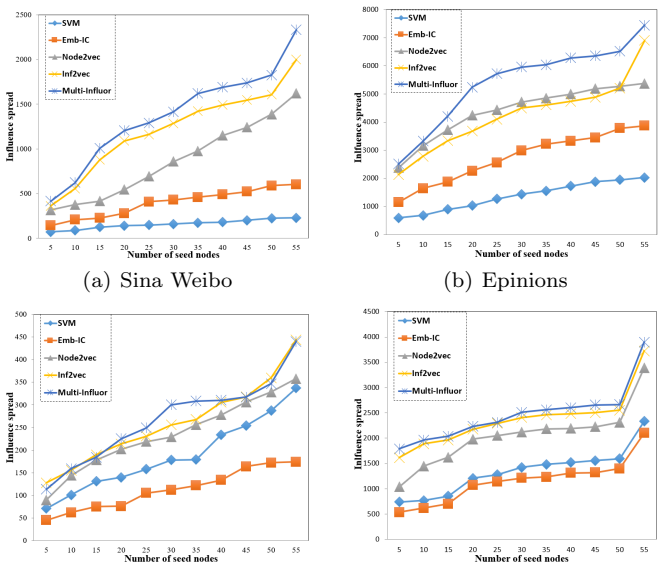
However, in the dataset of WikiVote, Node2vec and Emd-IC sometimes outperform than the Multi-Influor approach. Since the data size of WikiVote is small, the Multi-Influor approach lacks training data to improve the performance of neural network-based learning (GCN layers and fully connected neural network). Adding more sampling data in the training processes can improve the performance of Multi-Influor. These experimental results demonstrate that the Multi-Influor approach is highly data-driven.

4.4 Influence Spread Comparison

We uniformly used a greedy algorithm to detect seeds based on the prediction results [Sun, *et al.* (2018)]. Then, we compared the range of activated users influenced by seeds. After adding a new user to the seeds, we set the number of influence spread simulations to be 10,000. In addition, this study conducted comparisons under the conditions of different datasets and seed sizes.

In general, the comparisons of influence spread (Fig. 2) illustrate that the Multi-Influor approach always performs best, which means that it can lay a solid foundation

for solving the influence maximization problem. In the datasets of Sina Weibo, Epinions, and NetHEPT (Fig. 2 (a), (b), (d)), the Multi-Influor method performs much better than SVM and Emb-IC, and above 30% better than Node2vec. However, with much less training data in WikiVote, the approaches all perform worse than in other datasets. In WikiVote (Fig. 2 (c)), the Multi-Influor method only performs better than SVM and Emb-IC, and it achieves near performance to Node2vec and Inf2vec.



(a) Sina Weibo (b) Epinions

(c) WikiVote (d) NetHEPT

Fig. 2. Influence Spread Comparisons

The size of training data in neural networks could strongly affect the performance of Multi-Influor. However, these experimental results indicate that using more information propagation data improves the performance of the Multi-Influor approach. The real-world social networks provide extensive user data, which guarantees the performance of Multi-Influor. In other words, the Multi-Influor approach can be more practical on large-scale datasets.

5. CONCLUSION

This study focused on exploring a cross problem for multiple network embeddings and social influence prediction, and we developed a new end-to-end approach, namely Multi-Influor, that learns multiple network embeddings for each user in OSNs. The basic idea is to learn a neural-network-based function that considers multi-dimensional influence factors. The Multi-Influor approach takes a social network and action logs as input and outputs predictive influence probabilities. The experimental results illustrate that the Multi-Influor approach performs better than other baselines on influence predictions. Moreover, Multi-Influor is a highly data-driven method that can be more practical on real-world social networks.

REFERENCES

Al-Garadi, M. Ali, K. Varathan, S. Ravana, E. Ahmed, G. Mujtaba, M. Khan, S. Khan. Analysis of online social network connections for identification of influential users: Survey and open research issues. *ACM Computing Surveys*, Volume 51, Number 1: 16, 2018.

Aslay, C., L. Lakshmanan, W. Lu, X. Xiao. Influence maximization in online social networks. *ACM Inter. Conf. on Web Search and Data Mining*, pages. 775-776, 2018.

Bai, Y., H. Ding, S. Bian, T. Chen, Y. Sun, W. Wang. SimGNN: A neural network approach to fast graph similarity computation. *Twelfth ACM Inter. Conf. on Web Search and Data Mining*, pages. 384-392, 2019.

Feng, S., G. Cong, B. An, Y. Chee. Poi2vec: Geographical latent representation for predicting future visitors. *Thirty-First AAAI Conf. on Artificial Intelligence*, pages. 102-108, 2017.

Feng, S., G. Cong, A. Khan, X. Li, Y. Liu, Y. Chee. Inf2vec: Latent representation model for social influence embedding. *IEEE 34th Inter. Conf. on Data Engineering*, pages. 941-952, 2018.

Goyal, A., W. Lu, L. Lakshmanan. SIMPATH: An efficient algorithm for influence maximization under the linear threshold model. *IEEE Inter. Conf. on Data Mining*, pages. 211-220, 2011.

Grover, A., J. Leskovec. node2vec: Scalable feature learning for networks. *22nd ACM SIGKDD Inter. Conf. on Knowledge Discovery and Data Mining*, pages. 855-864, 2018.

Huang, X., S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. *IEEE Inter. Conf. on Computer Vision*, pages. 1501-1510, 2017.

Krishna, P., A. Sharada. Word embeddings-skip gram model. *Inter. Conf. on Intelligent Computing and Communication Technologies*, pages. 133-139, 2019.

Narayan, A., P. H. Roe. Learning graph dynamics using deep neural networks. *IFAC-PapersOnLine*, Volume 51, Number 2, pages. 433-438, 2018.

Pal, S. K., S. Kundu, C. A. Murthy. Centrality measures, upper bound, and influence maximization in large scale directed social networks. *Fundamenta Informaticae*, Volume 130, Number 3, pages. 317-342, 2014.

Peng, H., J. Li, Y. Song, Y. Liu. Incrementally learning the hierarchical softmax function for neural language models. *Thirty-First AAAI Conf. on Artificial Intelligence*, pages. 3267-3273, 2017.

Qiu, J., J. Tang, H. Ma, Y. Dong, K. Wang, J. Tang. DeepInf: Modeling influence locality in large social networks. *24th ACM SIGKDD Inter. Conf. on Knowledge Discovery and Data Mining*, pages. 2110-2119, 2018.

Sun, L., W. Huang, P. Yu, W. Chen. Multi-round influence maximization. *24th ACM SIGKDD Inter. Conf. on Knowledge Discovery and Data Mining*, pages. 2249-2258, 2018.

Wang, X., H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, P. Yu. Heterogeneous graph attention network. *The ACM World Wide Web Conference*, pages. 2022-2032, 2019.

Yang, L., A. Giua, Z. Li. Minimizing the influence propagation in social networks for linear threshold models. *IFAC-PapersOnLine*, Volume 50, Number 1, pages. 14465-14470, 2017.

Zhou, C., P. Zhang, J. Guo, X. Zhu, L. Guo. UBLF: An upper bound based approach to discover influential nodes in social networks. *IEEE Inter. Conf. on Data Mining*, pages. 907-916, 2013.

Zhang, Q., J. Wu, Q. Zhang, P. Zhang, G. Long, C. Zhang. Dual influence embedded social recommendation. *World Wide Web*, Volume 21, Number 4, pages. 849-874, 2018.