Leveraging Graph Embedding for Opinion Leader Detection in Dynamic Social Networks

Yunming Hui(✉), Mel Chekol, and Shihan Wang(✉)

Utrecht University, Utrecht, The Netherlands
y.m.hui@outlook.com, {m.w.chekol,s.wang2}@uu.nl

Abstract. Detecting opinion leaders from dynamic social networks is an important and complex problem. The few methods in this field are poor in generalisation and cannot fully consider various dynamic features. In this paper, we propose a novel and generic method based on dynamic graph embedding and clustering. Inspired by the existing knowledge about dynamic opinion leader detection, the proposed method can exploit both the topological and temporal information of dynamic social networks comprehensively. It is also generalisable, as shown experimentally on three different dynamic social network datasets. The experimental results show that the proposed method runs faster than competitors.

Keywords: Social Network Analysis · Opinion Leader Detection · Dynamic Graph Embedding · Dynamic Networks

1 Introduction

In social networks, opinion leaders are individuals whose opinions significantly influence others [6]. They can be experts in a given subject or users who have a significant number of followers on social media. As online social platforms such as Twitter and Instagram continue to grow and the flow of information becomes more rapid, the role that opinion leaders can play also continues to grow. In recent years, there has been increasing attention on how to automatically detect opinion leaders from social networks [3,33].

In reality, social networks are dynamic. One key aspect of this dynamic is reflected in the change of *topology*. For example, individuals may join or leave the network and connections between individuals keep changing. Such topological dynamic is critical to the detection of opinion leaders, as changes in network structure may result in individuals previously identified as opinion leaders losing their influence, or the emergence of new opinion leaders. The dynamics of social networks can also be reflected in the change of *semantic properties* of individuals with respect to their influence. In social networks, an individual's influence not only depends on the strength of their social connections, but also is affected by semantic features such as expertise, reputation and expressiveness [17]. The change in individual semantic properties is also important for opinion leaders because opinion leaders inherently have requirements on identity.

S. Nowaczyk et al. (Eds.): ECAI 2023 Workshops, CCIS 1948, pp. 5–22, 2024.
https://doi.org/10.1007/978-3-031-50485-3_1

The mentioned dynamic factors are crucial for opinion leader detection in social networks [9]. However, the vast majority of existing research tends to overlook such dynamic nature. A few studies that consider them and dynamically identify opinion leaders also suffer from generalization [8,14,27,34]. They rely on certain semantic information that is only available on specific social networks. In fact, only the dynamics of topological features are present in most social networks used by state-of-the-art systems. Thus, a generic solution that can consider both the dynamics of semantic and topological features, while relying on the dynamics of topological features, is still lacking in dynamic opinion leader detection.

Recently, graph embedding has been widely used in social network research to transform nodes into low-dimensional vector representations [5]. In particular, the dynamic graph embedding technique is able to retain information about the temporal changes of a graph in the vector representations. Such representation can preserve temporal, topological and semantics information of a graph [4,5,39]. Even if the node representations are generated only based on topological information, they can still capture some aspects of semantics through the relationships and contextual information present in the graph structure [38]. Additionally, it also allows us to infer or capture implicit relationships that are available in a given graph. Inspired by this, we propose a novel method based on dynamic graph embedding to detect opinion leaders in dynamic social networks.

Specifically, we first perform graph embedding of dynamic social networks and then cluster individual nodes based on the vector representations. Opinion leaders, as a special class of individuals, share similar topological and semantic characteristics [1,3,7]. Since the node representations generated by our chosen dynamic graph embedding method (TGNs [31]) can represent these two characteristics of nodes well, the opinion leaders will be clustered into the same cluster. Then the problem is converted into selecting the cluster containing all opinion leaders which is far more easier. Based on the significant and effective metrics for detecting opinion leaders (i.e. centrality metrics), we also design a method for selecting the cluster. In order to verify the generalisability and feasibility of our method, we evaluated the detection results on three real datasets using the influence spreading mode.

Our method can be generalised to different dynamic social networks (that none of the existing methods is capable of). It can consider temporal topological information alone but also provides ways to incorporate temporal semantic information. Moreover, our method is competitive with state-of-the-art on dynamic social network datasets with dynamic topology, while achieving superior runtime performance. Furthermore, it is a model-agnostic framework that allows the comprehensive exploitation of temporal information of both topological and semantic features in social networks.

2 Problem Definition

We present three commonly used models in social networks and based on them define the problem of dynamic opinion leader detection.

2.1 Social Network Models

Social networks can be predominantly characterized by three models: the static graph model, the snapshot graph model, and the continuous graph model.

Static Graph Model. When a social network is static, it is represented by a directed static graph $G = (V, E)$, where V is the node set and E is the edge set. Each node in V represents an individual which can be attributed to represent the properties of the individual, such as age, organizational status, etc. Each edge in E represents an interaction between individuals, such as following, sending a message, etc. It can also be attributed to indicate the properties of the interaction. The direction of an edge represents the initiator of an interaction. As an example shown in Fig. 1(a), nodes 1–7 represent users 1–7, respectively. Edge $(1, 4)$ represents an interaction initiated by user 1 to user 4, and the other edges have similar meanings.

Snapshot Graph Model. In the discrete case, a dynamic social network existing between t_s and t_e is represented with a series of directed graphs, denoted by $G = \{G_1, \cdots, G_i, \cdots, G_n\}$. The time interval $[t_s, t_e]$ is divided into n sub-intervals and the length of each sub-interval is $l = (t_e - t_s)/n$. For each snapshot $G_i = (V_i, E_i)$, V_i is the node set at time $t_s + (i-1) * l$ and E_i is the edge set including all edges within time interval $[t_s + (i-1) * l, t_s + i * l]$. As the example shown in Fig. 1(b), the time interval [0, 40] is divided into 4 intervals and 4 snapshot graphs are generated correspondingly. In the first snapshot, all the nodes exist in the social network at time 10 and the timestamps of all the edges are within the time interval $[0, 10]$.

Continuous Graph Model. In the continuous case, a dynamic social network existing between time t_s and t_e is represented by a directed graph with edges and nodes annotated with timestamps, denoted by $G = (V, E, T)$. V and E are the collections of nodes and edges over time $[t_s, t_e]$. Edges and nodes have the same format as in the static graph model and therefore can also be attributed. $T : V, E \rightarrow t \in [t_s, t_e]$ is a function that maps each node and edge to timestamps between time t_s and t_e. Taking $node3$ in Fig. 1(c) as an example, $T(v_3) = [1, 31]$. It indicates that node 3 exists in the social network from timestamp 1 to timestamp 31, and $T(e(v_1, v_7)) = 32$ means the interaction between two nodes happens at timestamp 32.

In summary, the three models contain increasing amounts of evolving information of social networks. The static graph model contains no evolving information while the continuous graph model contains the full evolving details. Consequently, the complexity of these three models increases in order and the problems defined on them become more difficult to address [13].

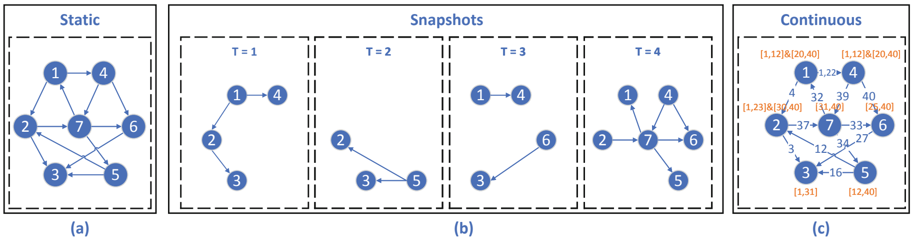

Fig. 1. An example social network represented with three different graph models. There are 7 users in this (dynamic) social network. Within a total of 40 time units, users may join/leave the network or interact with each other.

2.2 Dynamic Opinion Leader Detection

Definition 1 (Problem definition). *Given a social network represented with a continuous model $G = (V, E, T)$. The target of the dynamic opinion leader detection problem is to find a finite set of individuals. These individuals should have two main characteristics. One is the possession of a specific identity and authority and the reliance on them to influence other individuals. Another is that their influence, as evaluated based on the dynamic diffusion model, is also higher than that of other ordinary individuals.*

To maximise the use of temporal information of dynamic social networks, we define the dynamic opinion leader detection problem on the continuous graph model. In fact, temporal information plays an essential role in analysing and identifying opinion leaders. Take the network in Fig. 1 as an example. In-degree and out-degree of a node is the most naive way to measure whether a user is an opinion leader or not. The higher the out-degree of a user is, the more likely the user is an opinion leader. Thus, node 7 is most likely to be the opinion leader as it has both the highest in-degree and out-degree. However, considering the information diffusion procedure, it is clear that opinion leaders should accept something new before the majority of users to spread influence better. Therefore, node 7 should not be considered as an opinion leader because it participates in this dynamic social network at the last time. Instead, node 1 has more chance to be an opinion leader as node 1 may influence 4 nodes (nodes 2, 3, 4, 6) in this network, while node 7 can influence at most 2 nodes (nodes 1, 5). Therefore, it is clear that the temporal factor is not negligible and crucial to the opinion leader detection problem.

3 Related Work

In this section, we review the related work in opinion leader detection with a focus on dynamic opinion leader detection.

3.1 Opinion Leader Detection

Static Opinion Leader Detection. Opinion leaders have a high impact on the topology of social networks, which makes centrality metrics an important criterion for measuring opinion leaders in static opinion leader detection. Frequently used centrality includes degree centrality, betweenness centrality and closeness centrality [30,40]. Since these centrality metrics are often measured from a narrower aspect of centrality, they are not comprehensive. Therefore, a variety of methods have been developed, such as PageRank [28] and LeaderRank [20]. To counter the limitations of a single centrality metric, researchers also used various metrics in combination [12].

With the development of machine learning, many studies also use machine learning methods (mainly clustering or graph neural networks (GNNs) techniques) for opinion leader detection [2,15]. Opinion leaders, as a group of people with similar characteristics, can be well clustered into the same cluster by proper use of clustering techniques. Several methods followed this idea and applied the clustering algorithms to identify opinion leaders [2,11]. Later on, Yang et al. [41] used the DeepWalk graph embedding method to generate embedding vectors of users, and then combined them with network topology information to propose a local centrality index of network nodes to identify high-impact nodes. Luo et al. [21] used SNE (Social Network Embedding) model to obtain embedding vectors for each user. These embedding vectors were used to calculate the structure and text similarity between network nodes to improve efficiency.

3.2 Dynamic Opinion Leader Detection

Centrality metrics as an effective and easy-to-calculate metric have been applied to dynamic opinion leader detection as well. However, GNNs have not been used for related research yet. Song et al. [34] proposed a method for dynamic social networks consisting of users and comments. The researchers first rank the influence of comments by sentiment analysis. They also degenerate the dynamic social network into a static social network to calculate the degree of centrality and proximity prestige of each user. Then, opinion leaders are detected by considering three generated features. Only limited temporal information is considered in this study. It is reflected in the calculation of the influence of the comments: the longer the interval between sending the comments the less influence they have on each other. Huang et al. [14] followed this study and proposed a similar method. They improved the work by representing the relationship between the time factor and strength of impact between comments. A similar framework of generating snapshot networks to identify dynamic opinion leaders is also followed by Oueslati et al. [27]. The researchers first evaluated the influence of the post by semantic analysis of the post. Then, opinion leader detection is done based on the influence of the posts sent by each user. Only the impact of the post is counted in each snapshot graph and considered as a temporal feature in the detection. No temporal information has been used in the calculation of every user's influence. Chen et al. [8] proposed a distinct methodology. They do dynamic opinion leader detection on a post-and-follow dynamic social network. They first propose a method to transfer the dynamic network into a weighted static social network. The weights of edges are calculated based on the similarity of the two nodes and the time difference between the appearance of the two nodes (the only utilised temporal information). Then, they do community detection based on this weighted static social network and future detect the opinion leader with different attributes of nodes.

In summary, the limited existing methods of dynamic opinion leader detection do not make sufficient use of temporal information. Moreover, they are designed based on unique datasets containing information that is not always included in other datasets. Therefore, the generalisation of these methods is poor. Our method focuses on solving these problems.

To further clarify our focus, we also briefly mention and distinguish one related but different problem, namely dynamic information maximization. It identifies an initial set of nodes of predefined size k with the biggest influence spread in a dynamic social network. Please refer to [18] for a detailed review. This problem is different from opinion leader detection as it does not place any constraints on the identity of the node. When opinion leaders are difficult to be labelled in large-scale social networks, the influence of detected opinion leaders can be used to quantify the performance of opinion leader detection methods [3]. Aggarwal et al. [1] first proposed a method to evaluate the influence of a given set of nodes in a dynamic social network. This hill-climbing algorithm is then used to get the final seed set starting from a simple selection of individuals with the highest influence. Zhuang et al. [43] first construct a subgraph by probing a set of nodes in the underlying graph such that the influence diffusion can be best observed. Then, the seed set that can maximize the influence on the underlying graph is found on the subgraph. Unlike these two methods that select all individuals at once at the beginning stage, Michalski et al. [24] proposed a method that models dynamic social network using snapshots graph model and activates seeds based on sequential seeding in temporal networks. Notably, none of the existing dynamic information maximization methods applied dynamic graph embedding to capture various temporal features of dynamic networks.

4 Methodology

4.1 Overall Design and Framework

The overall framework of our method is presented in Fig. 2 and the specific flow of the method is shown in Appendix 7.2. Opinion leaders, as a special category of people, share the same characteristics, such as higher influence, accepting ideas and starting to spread them before most individuals do [16,37]. Based on this feature, we propose to categorise different individuals based on their (temporal) properties and then select the opinion leaders from all the categories. Such a choice is much simpler in computation compared to choosing opinion leaders from all the nodes. This is because the number of clusters (e.g. less than 10) is

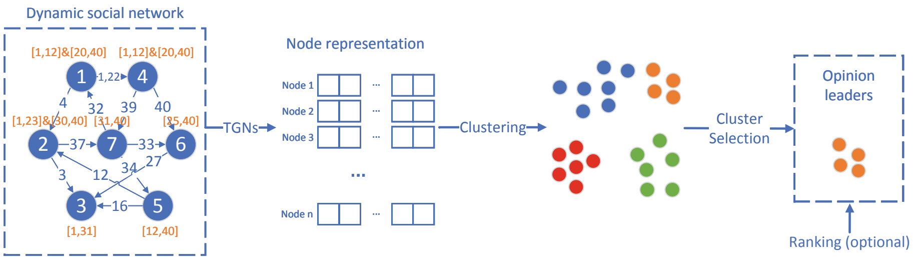

Fig. 2. Framework of the proposed method

usually smaller than that of opinion leaders (that is a small percentage of a large number of individuals and can be over a thousand).

So, we first transfer the dynamic graph into node representations (low dimensional vectors) with the dynamic graph embedding method. Then, the nodes are clustered based on node representations. This approach seeks to group nodes together based on the similarity of their embeddings, which encapsulate their structural roles within the network and how these roles evolve over time (especially influential factors). The principle here is that nodes within the same cluster should have similar features, as reflected in their embeddings. The resulting clusters can reveal patterns in the nodes' behaviors and roles that persist or evolve over time, and help us better understand the underlying structure and dynamics of the network [4,5,39]. In this manner, at a design level, our method makes better use of temporal information compared to existing methods.

Following these steps, the dynamic opinion leader detection problem is converted to select the cluster that contains all opinion leaders. Previous research has indicated that the centrality metrics can well reflect the probability of an individual being an opinion leader [3]. We therefore also design the algorithm for selecting the cluster containing all opinion leaders based on various centrality metrics. In real social scenarios, the number of nodes in this cluster may be higher than the required number of opinion leaders (e.g. we only care about the top ones). If needed, the opinion leaders in this cluster can also be ranked to select the most influential ones. In the next subsections, the method will be described in detail.

4.2 Dynamic Graph Embedding

In this framework, the node representations should be able to well represent the dynamic of topological features as well as the semantic features. Within various dynamic graph embedding methods introduced in Appendix 7.1, we employed the TGNs method which is good at retaining dynamics of both topological and semantic features and has been shown to perform significantly better than other existing dynamic graph embedding methods [31].

The TGNs method uses the encoder-decoder model. It has two core modules, a memory module and an embedding module. The memory module is used to represent the node's topological evolving history in a compressed format which allows TGNs to memorize long-term dependencies for each node in the graph. In the memory module, each node is represented with a low-dimensional vector which is updated when changes related to the node happen using Recurrent Neural Network (RNN). The main goal of the embedding module is to better aggregate the information of neighbouring nodes through Graph Neural Network (GNN) based on the information generated by the memory module. When the semantic features of nodes are available, the embedding module can incorporate the semantic information of nodes into the final node representations. Please refer to the original paper for details about the TGNs method [31]. In this way, the generated node representation shall encapsulate various features of nodes within the network both topologically and semantically, as well as how these features evolve over time.

4.3 Clustering and Cluster Selection

Next, we used the classical k-means [19] algorithm to cluster the nodes using their node representations. We followed the previous studies in social network analysis and used centrality metrics to determine the opinion leader clusters [3]. To apply three commonly used centrality metrics in our dynamic setting, we convert the dynamic social network represented with a continuous model into a snapshot model in this step. Then, we can calculate the three centrality metrics in each snapshot separately and add them proportionally. In addition, in dynamic social networks, opinion leaders should accept and disseminate ideas to others before most people do [37]. This means that the higher the centrality metric in the earlier snapshots the more likely they are to become opinion leaders. In other words, the high centrality metric in later snapshots, is not as useful as that in earlier snapshots. Node 7 in Fig. 1 is a good example. We therefore designed a decay function to reflect this pattern which will be explained in the following paragraphs.

Formally, we use the following equation to express the probability of an individual node being an opinion leader in a dynamic social network:

$$LScore(v) = \sum_{t=1}^{n} decay(\frac{t-1}{n}) \cdot (m_1 BC(v,t) + m_2 CC(v,t) + m_3 DC(v,t)), \quad (1)$$

where n is the number of snapshots and v is a node in the node set V. $BC(v,t)$, $CC(v,t)$ and $DC(v,t)$ is the betweenness centrality, closeness centrality and degree centrality of node v in t-th snapshot respectively. m_1, m_2 and m_3 are adjustable parameters to control the weighting of these three centrality metrics in the LScore. The decay function can be selected from the following five functions: $f_1(x) = 1 - x$, $f_2(x) = (x-1)^2$, $f_3(x) = 1 - x^2$, $f_4(x) = \cos(x)$, and $f_5(x) = (x+1)^2$.

Table 1. Statistics of three datasets. In the latter two datasets, the start and end times are anonymous indexes of time steps.

Dataset	Node Number	Edge Number	Start Time	End Time	Interval Length
UC-IRV	1,889	59,835	2004-04-15	2004-10-26	7 days
Bitcoinalpha	3,783	24,186	0	164246400	7,776,000 (90 days)
Bitcoinotc	5,881	35,592	0	164442412	7,776,000 (90 days)

These five decay functions represent different decay modes. The f_1 is a linear decay, which means that the contribution of centrality to importance decreases linearly. f_2 and f_3 are non-linear in decay, meaning that the effect of centrality on importance decays at different rates over time. The decaying effect of f_4 and f_5 is relatively small.

We further calculate the average probability of all nodes in a cluster and name it ALScore which is calculated as follows:

$$ALScore(c) = \frac{\sum_{i=1}^{|c|} LScore(v_i)}{|c|} \tag{2}$$

where $|c|$ is the number of nodes in cluster c and v_i is a node in cluster c. The cluster with the highest ALScore is considered as the cluster containing the opinion leaders. Sometimes the number of opinion leaders needed will be larger than the number of the top cluster containing opinion leaders. In such cases, more than one cluster can be considered, then individual nodes can be ranked to satisfy the requirement.

5 Experiments and Results

The code used for experiments can be found at here.

5.1 Experimental Setting

Dataset. Three different datasets used to carry out experiments are UC-IRV [29], Bitcoinalpha[1] and Bitcoinotc[2]. All of them can be structured as continuous graph models (defined in Sect. 2.1). The statistics of 3 datasets are shown in Table 1. The detailed information of datasets is explained in Appendix 7.3.

Parameter Setting. We used the source code[3] of the graph embedding method TGNs provided by the authors. The main parameters of TGNs used in experiments are as follows: the dimension of node embedding is 172, the learning rate

[1] http://snap.stanford.edu/data/soc-sign-bitcoin-alpha.html.
[2] https://snap.stanford.edu/data/soc-sign-bitcoin-otc.html.
[3] https://github.com/twitter-research/tgn.

is 0.001, the batch size is 200, the number of neighbours to sample is 20, the embedding model us graph attention network whose number of layer is 1 and number of heads used in attention layer is 2. The number of clusters is set to be 8 for UC-IRV and 7 for Bitcoinalpha and Bitcoinotc. The decay function we used is $decay_2$. m_1, m_2 and m_3 are all set to 1. The process of parameter optimization is shown in Appendix 7.4.

Evaluation Metric. Since it is difficult to label opinion leaders for large social networks, evaluating the influence of opinion leaders is a common method for evaluating the dynamic graph embedding method. The estimation of influence of nodes depends on the **diffusion model** which describes the rules of propagation of an individual's influence in a social network. Simulations based on the diffusion model can determine the number of individuals that an individual can influence.

The dynamic diffusion model we selected is Susceptible-Infectious (SI) model which is also used by Osawa et al. [26]. In the SI model, nodes can have two states S and I. A node in state S means it does not have information and a node in state I means it contains information. This model uses the snapshot form of a dynamic graph. If a node is in state I at time t, it tries to activate its neighbours at time $t{+}1$ with probability. In experiments, the affected probability of each node is set to 0.5. The ability of opinion leaders to spread influence is evaluated by the total number of nodes influenced by opinion leaders throughout the whole period. A larger number of influenced nodes means a higher impact on this network. In our experiments, due to the randomness in the diffusion process, we repeat the process 100 times and average the computation.

Baselines. We selected two kinds of baselines, static and dynamic opinion leader detection methods. The static opinion leader detection method we selected is **LeaderRank** [20]. It has a good performance among the static methods but ignores the dynamics of topological features. We therefore consider it as a representative of the static methods for comparison. Due to the unique data requirements and non-public datasets, all the existing dynamic opinion leader detection methods are difficult to be compared. Although the dynamic influence maximisation problem has some differences from the dynamic opinion leader detection problem, the requirement for the selected nodes to have high influence is the same. Therefore, two dynamic influence maximisation methods are selected to compare the influence of detected opinion leaders with ours. They are selected to represent two different mechanisms. **INDDSN** [1]: A method quickly estimates the influence of an individual in a dynamic social network and selects all individuals at once at the beginning stage. The forward version with better performance is used. **SSA** [24]: A state-of-art method selects individuals in a decentralised manner, which results in a higher spread of influence.

5.2 Clustering Result Analysis

The clustering results of three datasets are shown in Table 2. From the table, it can be seen that the size of clusters containing all opinion leaders are relatively

Table 2. Clustering results of three datasets. The numbers indicate the number of nodes in the cluster. The clusters containing opinion leaders are bolded.

Cluster	1	2	3	4	5	6	7	8
UC-IRV	292	**300**	162	288	251	214	166	226
Bitcoinalpha	671	621	**347**	333	560	728	523	–
Bitcoinotc	928	600	989	1,196	781	**243**	1,144	–

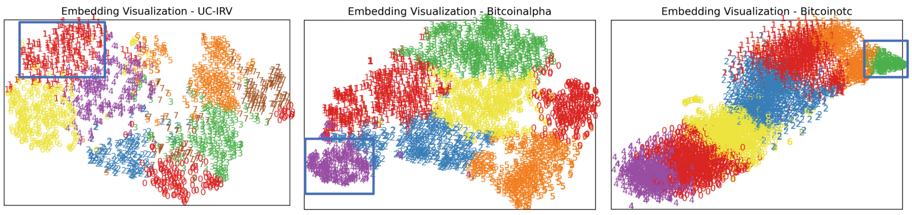

Fig. 3. Visualization of node representations of three datasets. The clusters containing opinion leaders are marked with the blue rectangular box.

small. This matches the characteristics of the opinion leader group that only a small set of individuals tend to lead the public in social networks [35].

To further validate our generated clusters, we also employed t-SNE [23] to visualise the node representation (See Fig. 3). A remarkable phenomenon from the visualization results is that all clusters containing opinion leaders are in the corners, i.e. the furthest away from all other clusters.

This phenomenon hints at the idea that opinion leaders possess distinct qualities or exert a significant level of influence that sets them apart from the general population within the dataset. Their placement in the corners of the visualization signifies that they are not only separate from other clusters, but their influence may also radiate outwards, impacting the entire network in a manner that distinguishes them from regular nodes. This phenomenon also confirms the feasibility of the method, i.e., the characteristics of the cluster containing the opinion leaders can be contained in the node embeddings (from dynamic graph embedding).

5.3 Performance Evaluation

We compared the influence of detected opinion leaders by the proposed method as well as the running time with three baselines.

Influence. As shown in Fig. 4, INDDSN has the best performance among the three baseline methods. The proposed method has similar or better performance on Bitcoinalpha and Bitcoinotc. In UC-IRV the performance of INDDSN is a little bit better than the proposed method. It is worth noting that the network of UC-IRV is dense and smaller than another two datasets which makes the

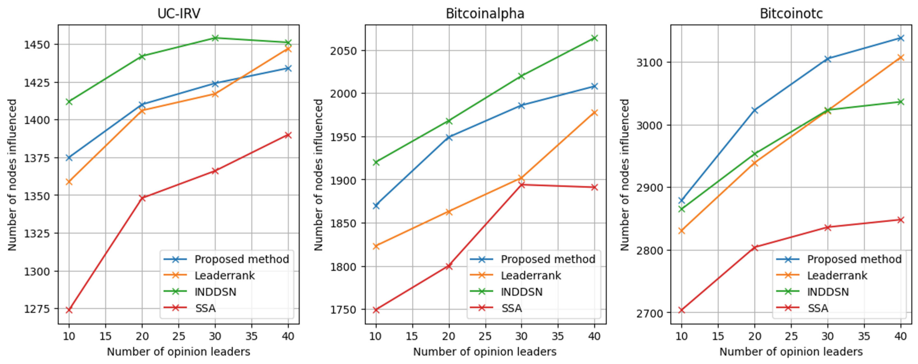

Fig. 4. Number of nodes influenced with different numbers of opinion leaders

difference less significant. Besides, as the number of opinion leaders increases, INDDSN's performance begins to deteriorate, and sometimes the number of people it can influence even decreases. This indicates that performance is difficult to guarantee when the number of opinion leaders is in high demand, and sometimes it is even lower than the static algorithm.

Running Time. The runtime performance of all methods with different numbers of opinion leaders is shown in Fig. 5. The running time of LeaderRank and the proposed method is similar for different numbers of opinion leaders as they produce the rank of each node at one time. The running time of INDDSN and SSA grows as the number of opinion leaders increases. The running time of INDDSN is the highest and is two to eight times higher than the running time of our proposed method.

Discussion. While the influential performance of the proposed method is similar to the state-of-art baseline INDDSN, our method still has superiority given the long running time of INDDSN. Even worse INDDSN uses a hill climbing algorithm. As the number of opinion leaders and the number of individuals in the social network increases, the time taken to search the neighbourhood solution space grows. When to stop searching is critical to the performance of the results, but how to set it up is tricky requiring a large number of experiments for each social network. Moreover, experimental results demonstrate that the advantages of our proposed method increase as the network size increases. This is particularly valuable because scalability is increasingly important for developing new solutions in social network analysis as there will be a high requirement for a larger size of the network in practical problems.

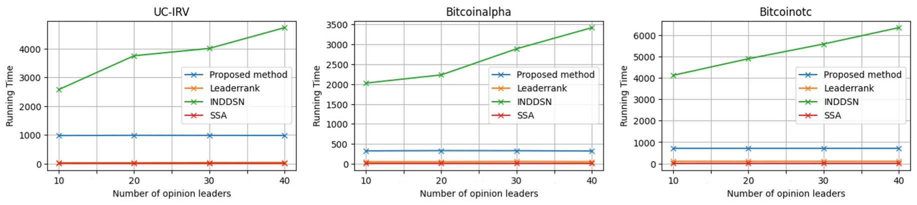

Fig. 5. Running time of different methods with different numbers of opinion leaders

6 Conclusion and Future Work

In this paper, we illustrate the importance of the dynamic nature of social networks for opinion leader detection. We proposed a novel method to detect opinion leaders from a continuously dynamic social network represented by the continuous model. We leveraged dynamic graph embedding and clustering algorithms to identify a group of opinion leaders who share similar temporal features. The results of generated clusters show that our method captures temporal and topological features of nodes. Compared to existing methods, our proposed method is less restrictive on the type of social network dataset. The method is evaluated on the influence of detected opinion leaders and running time, showing similar performance to other methods but higher operational efficiency. For future work, we want to explore new dynamic graph embedding methods by integrating influence-related knowledge into the node representations, which may bring better performance.

7 Appendix

7.1 Related Work on Dynamic Graph Embedding

Dynamic graph embedding methods can map the dynamic graph to a set of low-dimensional dense vectors. Each of these vectors represents a node. This representation improves the feasibility and efficiency of various network analysis tasks [4,39]. The purpose of dynamic graph embedding methods is that these low-dimensional vectors can not only maintain the topological information, but also the graph evolving information.

Following the static graph embedding methods [4], most early dynamic graph embedding methods update the graph representations quickly when social network changes [32,42]. The low-dimensional vector representation obtained by these methods does not represent the actual evolving information of graphs. Instead, only topological information about the graph after the change is captured. Recently, some proposed dynamic graph embedding methods can represent both topological and evolving information in the representations. Among them, several methods use random walk, which extends the random walk in static graphs by the addition of temporal constraints [10,25]. More methods use Recurrent Neural Network (RNN) to update node representations when changes on edge happened [22,31,36]. In this work, we apply this type of dynamic graph embedding method to consider the temporal evolution of social networks.

7.2 Methodology Summary

Following the above two subsections describe our design ideas and methodological principles, the pseudo-code of our method is shown in Algorithm 1.

Algorithm 1. Graph Embedding Based Dynamic Opinion Leader Detection

```
Require: dynamic graph: $G$, number of opinion leaders needed: $k$, number of cluster
    $c$, hyperparameter: $m_1$,$m_2$,$m_3$
Ensure: set of opinion leaders $S$
 1: $node\_embeddings \leftarrow TGNs(G)$
 2: $clusters \leftarrow k\text{-}means(node\_embeddings, num\_clusters)$
 3: $score\_clusters \leftarrow$ calculate ALScore of each cluster in $clusters$ with Eq. 2
 4: $S \leftarrow$ cluster with the highest ALScore
 5: if size of $S > k$ then
 6:     calculate LScore of each node in $S$ with Eq. 2
 7:     sort $S$ according to LScore
 8:     $S \leftarrow$ top $k$ nodes in $S$
 9: end if
10: return $S$
```

The dynamic social network is first transferred to node embeddings using TGNs. Then, the nodes are clustered based the node representations. Next, the ALScore of each cluster is calculated using Eq. 2. The cluster with the highest ALScore is selected as the cluster containing all opinion leaders. If the number of nodes in cluster is higher than the number of opinion leaders needed. The LScore of these nodes is calculated with Eq 1 and these nodes are ranked according to the LScore. The top k nodes are the final opinion leaders.

7.3 Dataset Description

Here is a detailed explanation of three used datasets.

- **UC-IRV** [29] UC Irvine messages dataset includes the users that sent or received at least one message in a Facebook-like Social Network originating from an online community for students at the University of California, Irvine. The database contains a total of 59,835 online messages, each message contains the sender, receiver and time of sending. The dataset is used to form a directed dynamic social network. Each edge represents a message which is directed from the sender of the message to the receiver of the message.
- **Bitcoinalpha**[4] Bitcoinalpha is a who-trusts-whom network of people who trade using Bitcoin on a platform called Bitcoin Alpha. The dataset contains

[4] http://snap.stanford.edu/data/soc-sign-bitcoin-alpha.html.

a total of 24,186 credit rating records from one user to another. Each record contains the person who made the credit rating and the person who was rated, the rating level and the time the rating was made. The dataset is used to form a directed dynamic social network. Each edge represents a credit rating record which is directed from the rating maker to the rate.

– **Bitcoinotc**[5] Bitcoinotc is similar to Bitcoinalpha, but it is collected from another Bitcoin platform, Bitcoin OTC. The number of records in this dataset is 35,592. The format of records is the same as that of Bitcoinalpha.

7.4 Parameters Optimization

In the proposed method, there are five important parameters: number of clusters c, m_1,m_2,m_3 and type of decay function. We tested different combinations of parameters in all three datasets.

We first optimize the number of clusters c. As we discussed in the last section, ALScore can roughly reflect the average probability that all nodes in a cluster might be opinion leaders. When we choose the value of c, we want the ALScore of the cluster with the highest ALScore to be the highest. The ALScore of the cluster with the highest ALScore varying with the value of c, other conditions being equal, is shown in Table 3. It can be seen that the best results are achieved when c is set to be 7 or 8. We recommend that in subsequent experiments c will be set to 7 or 8 for best results.

Table 3. ALScore of the cluster with the highest ALScore varying with the value of c

c	5	6	7	8	9
UC-IRV	0.41	0.42	0.43	**0.45**	0.44
Bitcoinalpha	0.46	0.48	**0.49**	0.48	0.46
Bitcoinotc	0.52	0.55	**0.56**	0.55	0.54

Following, we optimize m_1,m_2,m_3 and type of decay function. The test the influence of opinion leaders detected with combinations of different type of decay function and centrality metrics in dataset UC-IRV and Bitcoinalpha. The results are shown in Table 4. For the decay function, $decay_2$ can always bring the best performance. Therefore we suggest that the preferred decay function is $decay_2$, which will also be used in the next experiments. For the centrality indicator, closeness centrality performs significantly lower than the other two centrality indicators. Considering the time-consuming calculation of closeness centrality, we suggest ignoring closeness centrality, i.e., setting m_3 to 0. Betweenness centrality and degree centrality perform similarly, so we suggest setting both m_1 and m_2 to 1 so that they contribute equally to LScore. This will also be used in the following experiments.

[5] https://snap.stanford.edu/data/soc-sign-bitcoin-otc.html.

Table 4. The influence of opinion leaders detected with combinations of different types of decay functions and centrality metrics

Dataset	UC-IRV			Bitcoinalpha		
Centrality metric	Betweenness	Degree	Closeness	Betweenness	Degree	Closeness
decay_1	1247	1245	1219	1645	1636	1631
decay_2	**1261**	**1257**	**1183**	**1679**	**1664**	**1628**
decay_3	1232	1217	1213	1636	1630	1566
decay_4	1196	1191	1180	1597	1601	1513
decay_5	1200	1203	1187	1611	1608	1521

References

1. Aggarwal, C.C., Lin, S., Yu, P.S.: On influential node discovery in dynamic social networks. In: Proceedings of the 2012 SIAM International Conference on Data Mining, pp. 636–647. SIAM (2012)
2. Arvapally, R.S., Liu, X., Jiang, W.: Identification of faction groups and leaders in web-based intelligent argumentation system for collaborative decision support. In: 2012 International Conference on Collaboration Technologies and Systems (CTS), pp. 509–516. IEEE (2012)
3. Bamakan, S.M.H., Nurgaliev, I., Qu, Q.: Opinion leader detection: a methodological review. Expert Syst. Appl. **115**, 200–222 (2019)
4. Barros, C.D., Mendonça, M.R., Vieira, A.B., Ziviani, A.: A survey on embedding dynamic graphs. ACM Comput. Surv. (CSUR) **55**(1), 1–37 (2021)
5. Cai, H., Zheng, V.W., Chang, K.C.C.: A comprehensive survey of graph embedding: problems, techniques, and applications. IEEE Trans. Knowl. Data Eng. **30**(9), 1616–1637 (2018)
6. Chan, K.K., Misra, S.: Characteristics of the opinion leader: a new dimension. J. Advert. **19**(3), 53–60 (1990)
7. Chen, Y.C., Cheng, J.Y., Hsu, H.H.: A cluster-based opinion leader discovery in social network. In: 2016 Conference on Technologies and Applications of Artificial Intelligence (TAAI), pp. 78–83. IEEE (2016)
8. Chen, Y.C., Hui, L., Wu, C.I., Liu, H.Y., Chen, S.C.: Opinion leaders discovery in dynamic social network. In: 2017 10th International Conference on Ubi-media Computing and Workshops (Ubi-Media), pp. 1–6. IEEE (2017)
9. Cordeiro, M., Sarmento, R.P., Brazdil, P., Gama, J.: Evolving networks and social network analysis methods and techniques. In: Social Media and Journalism-Trends, Connections, Implications, pp. 101–134 (2018)
10. Du, L., Wang, Y., Song, G., Lu, Z., Wang, J.: Dynamic network embedding: an extended approach for skip-gram based network embedding. In: IJCAI, vol. 2018, pp. 2086–2092 (2018)
11. Duan, J., Zeng, J., Luo, B.: Identification of opinion leaders based on user clustering and sentiment analysis. In: 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), vol. 1, pp. 377–383. IEEE (2014)
12. Gao, C., Wei, D., Hu, Y., Mahadevan, S., Deng, Y.: A modified evidential methodology of identifying influential nodes in weighted networks. Phys. A **392**(21), 5490–5500 (2013)

13. Hafiene, N., Karoui, W., Romdhane, L.B.: Influential nodes detection in dynamic social networks: a survey. Expert Syst. Appl. **159**, 113642 (2020)
14. Huang, B., Yu, G., Karimi, H.R., et al.: The finding and dynamic detection of opinion leaders in social network. Math. Prob. Eng. **2014**, 7 (2014)
15. Jain, L., Katarya, R., Sachdeva, S.: Opinion leaders for information diffusion using graph neural network in online social networks. ACM Trans. Web **17**(2), 1–37 (2023)
16. Katz, E., Lazarsfeld, P.F.: Personal Influence: The Part Played by People in the Flow of Mass Communications. Routledge (2017)
17. Li, Y., Chen, W., Wang, Y., Zhang, Z.L.: Influence diffusion dynamics and influence maximization in social networks with friend and foe relationships. In: Proceedings of the Sixth ACM International Conference on Web Search and Data Mining, pp. 657–666 (2013)
18. Li, Y., Fan, J., Wang, Y., Tan, K.L.: Influence maximization on social graphs: a survey. IEEE Trans. Knowl. Data Eng. **30**(10), 1852–1872 (2018)
19. Lloyd, S.: Least squares quantization in PCM. IEEE Trans. Inf. Theory **28**(2), 129–137 (1982)
20. Lü, L., Zhang, Y.C., Yeung, C.H., Zhou, T.: Leaders in social networks, the delicious case. PLoS ONE **6**(6), e21202 (2011)
21. Luo, J., Du, Y., Li, R., Cheng, F.: Identification of opinion leaders by using social network embedding. In: 2019 IEEE 5th International Conference on Computer and Communications (ICCC), pp. 1412–1416. IEEE (2019)
22. Ma, Y., Guo, Z., Ren, Z., Tang, J., Yin, D.: Streaming graph neural networks. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 719–728 (2020)
23. Van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**(11), 2579–2605 (2008)
24. Michalski, R., Jankowski, J., Bródka, P.: Effective influence spreading in temporal networks with sequential seeding. IEEE Access **8**, 151208–151218 (2020)
25. Nguyen, G.H., Lee, J.B., Rossi, R.A., Ahmed, N.K., Koh, E., Kim, S.: Continuous-time dynamic network embeddings. In: Companion Proceedings of the the Web Conference 2018, pp. 969–976 (2018)
26. Osawa, S., Murata, T.: Selecting seed nodes for influence maximization in dynamic networks. In: Mangioni, G., Simini, F., Uzzo, S.M., Wang, D. (eds.) Complex Networks VI. SCI, vol. 597, pp. 91–98. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16112-9_9
27. Oueslati, W., Arrami, S., Dhouioui, Z., Massaabi, M.: Opinion leaders' detection in dynamic social networks. Concurrency Comput. Pract. Exp. **33**(1), e5692 (2021)
28. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: bringing order to the web. Tech. rep. Stanford InfoLab (1999)
29. Panzarasa, P., Opsahl, T., Carley, K.M.: Patterns and dynamics of users' behavior and interaction: network analysis of an online community. J. Am. Soc. Inform. Sci. Technol. **60**(5), 911–932 (2009)
30. Risselada, H., Verhoef, P.C., Bijmolt, T.H.: Indicators of opinion leadership in customer networks: self-reports and degree centrality. Mark. Lett. **27**(3), 449–460 (2016)
31. Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, F., Bronstein, M.: Temporal graph networks for deep learning on dynamic graphs. arXiv preprint arXiv:2006.10637 (2020)

32. Sankar, A., Wu, Y., Gou, L., Zhang, W., Yang, H.: DySAT: deep neural representation learning on dynamic graphs via self-attention networks. In: Proceedings of the 13th International Conference on Web Search and Data Mining, pp. 519–527 (2020)
33. Sharma, K., Bajaj, M., et al.: A review on opinion leader detection and its applications. In: 2022 7th International Conference on Communication and Electronics Systems (ICCES), pp. 1645–1651. IEEE (2022)
34. Song, K., Wang, D., Feng, S., Yu, G.: Detecting opinion leader dynamically in Chinese news comments. In: Wang, L., Jiang, J., Lu, J., Hong, L., Liu, B. (eds.) WAIM 2011. LNCS, vol. 7142, pp. 197–209. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28635-3_19
35. Tang, J., Lou, T., Kleinberg, J.: Inferring social ties across heterogenous networks. In: Proceedings of the fifth ACM International Conference on Web Search and Data Mining, pp. 743–752 (2012)
36. Trivedi, R., Farajtabar, M., Biswal, P., Zha, H.: DyRep: learning representations over dynamic graphs. In: International Conference on Learning Representations (2019)
37. Valente, T.W., Pumpuang, P.: Identifying opinion leaders to promote behavior change. Health Educ. Behav. **34**(6), 881–896 (2007)
38. Wu, L., Zhao, H., Li, Z., Huang, Z., Liu, Q., Chen, E.: Learning the explainable semantic relations via unified graph topic-disentangled neural networks. ACM Trans. Knowl. Discov. Data **17**(8), 1–23 (2023)
39. Xue, G., Zhong, M., Li, J., Chen, J., Zhai, C., Kong, R.: Dynamic network embedding survey. Neurocomputing **472**, 212–223 (2022)
40. Yang, L., Qiao, Y., Liu, Z., Ma, J., Li, X.: Identifying opinion leader nodes in online social networks with a new closeness evaluation algorithm. Soft. Comput. **22**(2), 453–464 (2018)
41. Yang, X.H., et al.: Identifying influential spreaders in complex networks based on network embedding and node local centrality. Phys. A **573**, 125971 (2021)
42. Yu, W., Cheng, W., Aggarwal, C.C., Zhang, K., Chen, H., Wang, W.: NetWalk: a flexible deep embedding approach for anomaly detection in dynamic networks. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2672–2681 (2018)
43. Zhuang, H., Sun, Y., Tang, J., Zhang, J., Sun, X.: Influence maximization in dynamic social networks. In: 2013 IEEE 13th International Conference on Data Mining, pp. 1313–1318. IEEE (2013)