Deep reinforcement learning-based approach to tackle positive influence maximization in signed social networks

Longxiang Li[1], Yanhao Sun[2], Tianwei Lin[2], Nuan Song[2,4*], Liming Wu[2], Tianqi Sun[2], Wei Si[2], Yongjian Zhao[2], Dong Li[2,3,4*]

[1]Shandong Qilu Electrical Machinery Manufacture Co., Ltd, Jinan 250100, China

[2]School of Mechanical, Electrical & Information Engineering, Shandong University, Weihai 264209, China

[3]Shenzhen Research Institute of Shandong University, Shenzhen 518000, China

[4]Shandong Key Laboratory of Intelligent Electronic Packaging Testing and Application, Shandong University, Weihai 264209, China

Keywords: Positive influence maximization; Signed social network; Deep reinforcement learning; Network embedding

Citation: Li L.X., Sun Y.H., Lin T.W., et al.: Deep reinforcement learning-based approach to tackle positive influence maximization in signed social networks. Data Intelligence, 2026, Vol. 8, Art. No.: 20250039. DOI: https://doi.org/10.3724/2096-7004.di.2025.0039

ABSTRACT

The Influence Maximization problem has garnered significant research interest since its introduction. In 2014, the problem was further extended to include signed social networks, resulting in the positive influence maximization problem and negative influence maximization problem. However, current solutions mainly rely on greedy algorithms that suffer from the Inefficient shortcoming, which do not leverage deep reinforcement learning. This paper introduces a novel approach to maximize the number of positively activated nodes by applying deep reinforcement learning to signed networks. Specifically, we extend SDGNN model for network representation learning and design a DQN-based seed node selection algorithm. The extensive experimental results on four real-world networks demonstrate our proposed model outperforms greedy algorithm and CELF algorithm, in terms of both time efficiency and influence spread quality. To our knowledge, this work is the first to leverage deep reinforcement learning to solve influence maximization problem in signed social networks.

*Corresponding authors: Nuan Song (E-mail: 202317543@mail.sdu.edu.cn); Dong Li (E-mial: dongli@sdu.edu.cn).

© 2026 Chinese Academy of Sciences. Published under a Creative Commons Attribution 4.0 International (CC BY 4.0) license.

1. INTRODUCTION

In recent years, social network modeling in real life has enjoyed great popularity, such as Twitter, Facebook and Twitch. Nodes on network represent people in the world and edges reflect the interactions between them. Social networks can promote the development of viral marketing first proposed by Domingos and Richardson [1], where a group of users are designed to accept some goods initially, and based on the diffusion process in the network, their friends can be influenced to determine whether to buy. If more reasonable strategy of selecting the initial group is adopted, the product sales can get increased more dramatically. This problem was defined by Kempe et al. [2] as Influence Maximization (IM) problem. They proved the problem NP-hard and proposed a greedy algorithm to solve the IM problem, which guarantees $1-\frac{1}{e}$ probability to obtain the optimal solution.

Sorted by the polarity of edges, social network can be divided into two categories: unsigned network (*i.e.* only positive weight edges) and signed network (*i.e.* both positive and negative weight edges), as shown in Figure 1. Taking Bitcoin-Alpha[①] as an example, it is a "Who-trust-Whom" signed network whose edge weights range between –10 (representing total distrust) and +10 (representing total trust). In signed networks, nodes can be activated into two states: positive state and negative state. Similar to IM problem in unsigned network, in signed network, there are positive influence maximization problem (PIM) and negative influence maximization problem (NIM). In this paper, we focus on PIM problem which describes how to select some seed nodes initially in a social network such that the final positive influence is maximized.

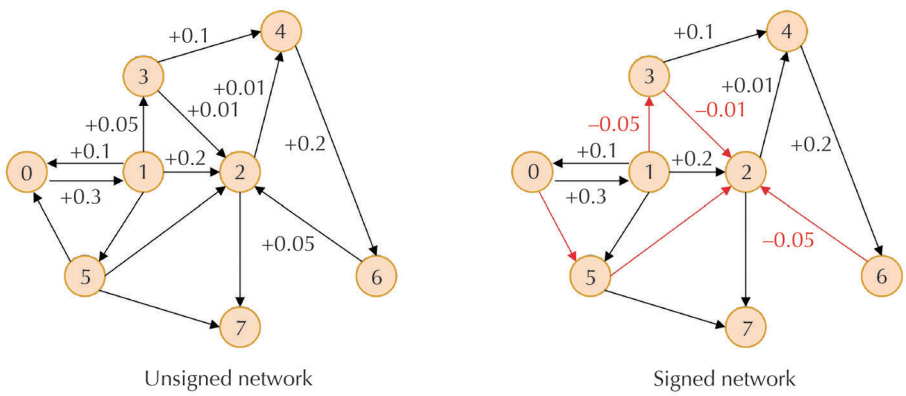

Figure 1. Unsigned network and Signed network.

IM problem was claimed by Kempe et al. [2] to have two significant properties: monotonicity and submodularity, which provided theoretical basis for the greedy algorithm. Based on the submodularity property, CELF [3] algorithm was proposed to improve the efficiency of greedy algorithm. Later, Li et al. [4] proved that PIM problem possesses the two properties of IM problem as well, so the greedy algorithm and

① http://snap.stanford.edu/data/soc-sign-bitcoin-alpha.html.

CELF algorithm are still applicable. There are also some alternative methods available for PIM problem, such as ICWNP model proposed by Qiu et al. [5]. Li et al. [6] proposed an alternative method based on simulated annealing (SA) to solve the positive influence maximization problem, and proposed two heuristic methods to accelerate the convergence process of the proposed method. Yin et al. [7] designed a novel Signed-PageRank (SPR) algorithm to achieve influence maximization in signed social networks. Ju et al. [8] proposed an effective algorithm to maximize the positive influence of the signed network under the independent cascade model, and calculated the propagation probability to avoid the influence diffusion of the simulated candidate seed node selection, so as to obtain the maximum positive influence propagation in the signed network. Inspired by deep learning techniques, Kumar et al. [9] proposed a novel approach to solve the influence maximization problem as a classical regression task using transfer learning via graph-based long short-term memory (GLSTM). These algorithms can have a good trade-off between influence spread and consuming time. However, no one has taken deep reinforcement learning (DRL) into account. Inspired by DISCO [10] which was proposed to solve the IM problem using DRL, we apply DRL to the PIM problem and get a better performance than baselines. In our approach, we adopt SDGNN [11] to perform signed network embedding and use DQN to select nodes. Based on the original SDGNN algorithm, we make some improvements in the aggregator part so that it can be more suitable for weighted network. Furthermore, some tricks are utilized to optimize the training process. According to our empirical results, we find that our method outperforms the greedy algorithm and the CELF algorithm on sparse networks. Our paper is organized as follows: In Section 2, we make a summary of some state-of-the-art algorithms to solve PIM problem. Section 3 introduces some preliminaries and our model. Section 4 reports our experiment details. Finally, the conclusion is presented in Section 5.

2. RELATED WORK

In this section, we will introduce some novel and efficient algorithms for solving IM problem in unsigned networks and PIM problem in signed networks respectively. For IM problem, the most classical method is the greedy algorithm proposed by Kempe et al. [2]. However, one shortcoming is that it requires many time-consuming simulations. Based on the submodularity property of this problem, CELF algorithm [3] adopts the strategy of lazy updating to reduce the running time, which is about 700 times faster than the greedy algorithm. There are a great number of algorithms which can solve IM problem efficiently. For example, Chen et al. [12] improved both the original greedy algorithm and influence spread process. One useful idea is to pick out and eliminate the less influential nodes in the preliminary phase. Cui et al. [13] proposed a degree-descending search strategy (DDS), in which the nodes whose influence spread is not comparable to the degree centrality can be eliminated. Similarly, an iterative framework in descending order was used by He et al. [14] to determine the candidate nodes. Lately, Kianian and Rostamnia [15] adopted a practical heuristic method to prune nodes while Biswas et al. [16] utilized a multi-criteria decision making (MCDM) approach. Furthermore, some researchers set sights on the community property of social networks and have achieved good results. Li et al. [17] considered the community diversity which is ignored by most algorithms but plays an important role in elevating the influence effect in the future promotion. Kumar et al. [18] took community structure into account,

and used node seeding, label propagation and community detection to maximize influence. Apart from the community attributes, the distance between nodes is also proven to be a useful factor. For example, in [19], the selected nodes are limited to proper distances, which leads to less overlap and wider coverage area of the selected nodes. In other words, this operation can help maximize influence spread. In fact, the submodularity of the influence function can make a difference as well. Based on the characteristic of submodularity, Wu and Küçükyavuz [20] proposed a delayed constraint generation algorithm, which outperforms the basic greedy algorithm. With the development of DRL, some researchers find it practical to apply DRL to solving IM problem. The process usually consists of two main parts: node embedding and DRL model training. Li et al. [10] trained an end-to-end model, named DISCO, which adopts a variation of Structure2vec [21] to implement nodes embedding and deep Q network to estimate the influence. It claims to complete the cross-graph feature extraction so that the model can be applied to multiple homogeneous networks and present better generality. However, DISCO could not perform well on dynamic networks. In [22], Graph Attention Network (GAT) [23] is used to extract both the feature of nodes and the local structure so that the model can handle dynamic change of networks. Nevertheless, these algorithms using DRL cannot be directly applied to signed networks because the embedding process and the method to calculate influence of selected nodes set are different. The next discussion is concerned with some methods which can work well for PIM problem in signed networks.

PIM problem Compared with solving IM problem in unsigned networks, there are less researches about solving PIM problem in signed networks. In [6], an algorithm based on sample annealing was proven to be efficient to solve PIM problem. Subsequently, Ju et al. [8] proposed a new algorithm which firstly computes the activation probabilities between node pairs and then utilizes a propagation increment function to determine the candidate seeds. Later, ICWNP propagation model compatible with a greedy algorithm was proposed [5]. Conversely, Sheng et al. [24] paid more attention to the role of negative edges and set some propagation rules to model the influence spread process.

3. PROBLEM DEFINITION AND PROPOSED METHODS

In this section, we will show the formulation of PIM problem and give an explicit description of our framework.

3.1 Polarity-Related Influence Maximization

Suppose we represent a social network by a weighted directed network $G(V, E, W)$, where V denotes the set of nodes in the social network, E denotes the set of directed edges of the network, $|V| = n$ denotes the number of nodes, and $|E| = m$ denotes the number of edges. For each directed edge from node u to node v, $(u, v) \in E$, and each edge has a weight w. The value of this weight indicates the trust relationship (positive value) and distrust relationship (negative value) between the nodes, and the absolute magnitude indicates the strength of this relationship.

In addition, we define the $\sigma_+(.)$ function to denote the positive influence function, which receives a weighted directed network G and an initial set of seed nodes S, and returns the expected number of nodes that end up being positively activated under a particular diffusion model ψ. Similarly, we can also define the $\sigma_-(.)$ function, which denotes the negative influence function.

Accordingly, the **P**olarity-**R**elated **I**nfluence **M**aximization (PRIM) problem can be defined as follows.

Definition 1. *Given a certain weighted directed network $G = (V, E, W)$, a certain diffusion model ψ and an initial set of seed nodes of size $k (k > 0)$. The goal of* ***Polarity-Related Influence Maximization*** *problem is to select k seeds from the nodes set to compose an initial set S $S \subseteq V$, $|S| = k$ such that under the ψ diffusion model, the value of $\sigma_+(G, \psi)$ or $\sigma_-(G, \psi)$ is maximized.*

For PIM problem, we adopt the first definition, *i.e.*, let $\sigma_+(G, \psi)$ be maximized. We define $\sigma(S)$ to denote the final value of positive influence obtained by the initial seed set S, then the above definition can be expressed as follows:

$$S^* = \underset{S \subseteq V, |S| = k}{\arg\max} \sigma(S) \tag{1}$$

3.2 Polarity-related Diffusion Model

We will first introduce two influence diffusion models in unsigned social networks.

3.2.1 Linear Threshold (LT) model

We define $N(u)$ to denote the set of all neighbor nodes of node u and $N_{active}(u)$ to denote the set of activated neighbors. So $N_{active}(u)$ is a subset of $N(u)$.

In the LT model, each node u has a threshold θ_u that serves as a criterion to determine whether it is activated or not and the weight of edges needs to meet the requirements shown in Equation (2). Given a positive-weighted directed network $G = (V, E, W)$ and an initial seed set S, the LT model first activates the initial seed nodes and starts propagation. In each propagation, node u which has not been activated before is activated only if $\sum_{v \in N_{active}(u)} w(v, u) \geqslant \theta_u$ is satisfied. In the next round of propagation, since u has already been activated, it will try to activate its inactive neighbor nodes. The LT model stops until no more nodes can be activated.

$$\begin{cases} w(v, u) \leqslant 1, \ v \in N(u) \\ \sum_{v \in N(u)} w(v, u) = 1 \end{cases} \tag{2}$$

3.2.2 Independent Cascade (IC) model

The diffusion process of the IC model starts with an initial set of nodes. When a node u is activated for the first time at timestamp t, it can activate an inactive neighbor with the success probability of $|w(v, u)|$, and the failure probability of $1-|w(v, u)|$. If node v is successfully activated in timestamp t, it will get the

ability to activating its neighbors only in timestamp $t + 1$ and the ability will vanish in subsequent steps. This process will continue until no new node can be activated.

Due to the involvement of polarity, we adopted a variant of the independent cascade model as the diffusion model, namely IC-P model [4]. The main differences between the IC-P model and the original IC model are described as follows.

- The states of node changes from active and inactive to inactive, positive activation and negative activation, where negative activation indicates the user's distrust of the information.
- Four basic sociological guidelines have been integrated, namely: "A friend of my friend is a friend", "An enemy of my friend is an enemy", "A friend of my enemy is an enemy", and "An enemy of my enemy is a friend".

Without loss of generality, we let all seed nodes be positively activated at the beginning. The Figure 2 provides an intuitive overview of the complete workflow, highlighting the interaction between three key components: (1) the signed graph embedding module (based on SDGNN), which encodes structural and polarity-aware features into dense representations; (2) the polarity-aware diffusion model (IC-P), which governs the simulation of influence propagation in signed networks; and (3) the DQN-based decision module, which iteratively selects seed nodes based on state-action values learned through environment feedback.

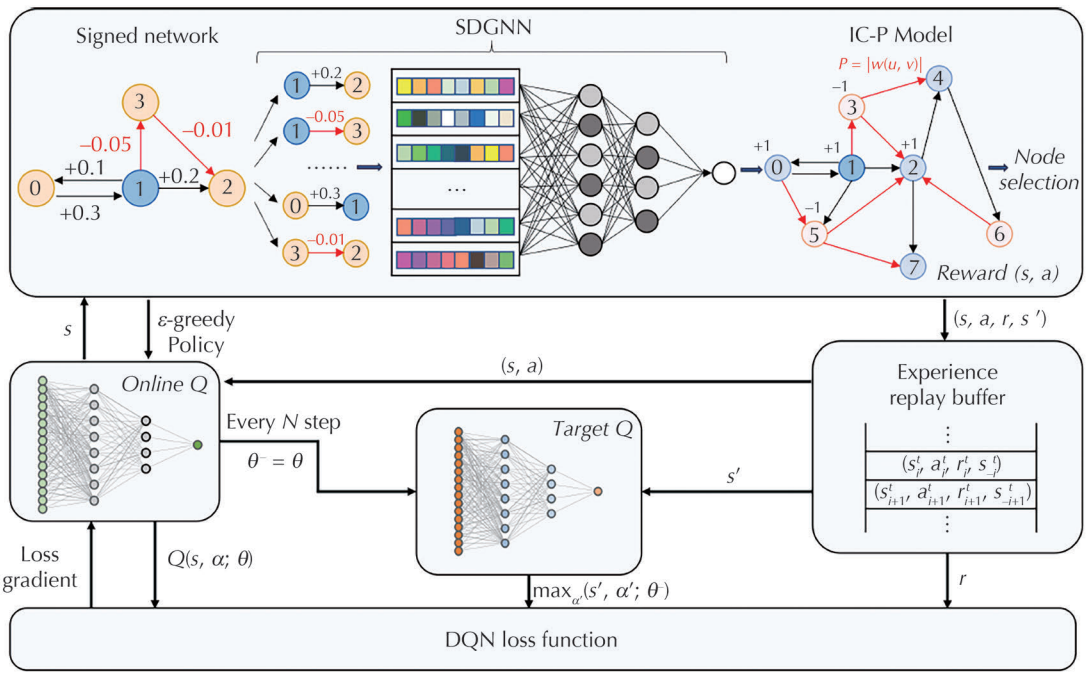

Figure 2. Frame diagram of our algorithm.

3.3 Network Embedding

Network embedding is a technology that can map nodes into low-dimensional vectors, which can reflect the information of the network and relations between nodes. For example, if two nodes are close in a network, their vectors may be similar. Generally, embedding technology is used to preprocess the network data. After that, the low-dimensional vectors can be used to solve different downstream tasks, such as link prediction, node classification. Nowadays, node2vec [25], GAT [23] and GCN [26] are popular network embedding algorithms. There are also some algorithms taking adversarial mechanism into consideration, such as SNEGAN [27] and ASiNE [28]. In this paper, we adopt SDGNN [11] to complete the embedding part for the following main reasons. Firstly, it can be applied to signed networks; Secondly, it considers sociology theory which is significant in social networks. However, the original SDGNN can not fully satisfied our demands, because it only takes edge weight as 1 or –1. In our application scenario, the absolute value of edge weight represents the trust (positive) or distrust (negative) level (i.e., the level has positive correlation with the absolute value). So we can not simply fix them as only ±1 and we extend the SDGNN by adding edge weights into the algorithm.

Here come some details of the extended SDGNN: There are 4 types of edge relations as shown in Figure 3. For every node u, we can get 4 types of neighbors grouped by the 4 relations, respectively. Then it uses mean aggregator to aggregate the information of each group and uses MLP to encode these messages, as shown in Equations (3) and (4). The aggregate-encode process can be execute several times continuously.

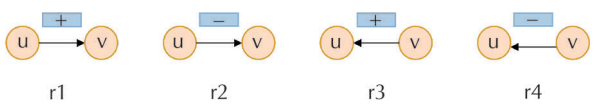

Figure 3. Four different kinds of edge relations.

$$X_{r_i}^k(u) = \sigma\left(W_{r_i}^k \cdot Mean\left(\left\{z_v^k,\ v \in N_{r_i}(u)\right\}\right)\right) \quad i = 1,\ 2,\ 3,\ 4 \tag{3}$$

$$z_u^{k+1} = MLP\left(z_u^k \left\| X_{r1}^k(u) \right\| \ldots \left\| X_{r4}^k(u)\right.\right) \tag{4}$$

where, z_u^k denotes the k-th layer's embedding result of node u, $N_{r_i}(u)$ is the neighbor set of node u under relation r_i, $X(u)$ represents the aggregated information of us neighbors, and "||" denotes the operation of concatenation.

What's more, $Mean(.)$ is a function that can calculate the element-wise weighted average of neighbors' embedding. In the original SDGNN algorithm, the coefficient corresponding to each neighborhood node is the same value of $\frac{1}{|N(u)|}$, as shown in Equation 5.

$$Mean\left(\left\{z_v,\ v \in N(u)\right\}\right) = \frac{\sum_{v \in N(u)} z_v}{|N(u)|} \tag{5}$$

The *Mean*(.) function in our extended SDGNN works as Equation 6. For node u, the coefficient of neighbor v is the product of d_v and $abs\ (w_{u,\ v})$, where d_v is the degree of v and $abs\ (w_{u,\ v})$ is the absolute value of the edge weight between u and v. The role of d_v is representing the number of vs neighbors. And $abs\ (w_{u,\ v})$ can reflect the strength of relationship between u and v. Similar to the attention mechanism, the product of the two items can be rated as the importance of neighbor v.

$$Mean\left(\left\{z_v,\ v \in N(u)\right\}\right) = \frac{\sum_{v \in N(u)} d_v \times abs\left(w_{u,\ v}\right) \times z_v}{\sum_{v \in N(u)} d_v \times abs\left(w_{u,\ v}\right)} \tag{6}$$

The detail of our extended SDGNN model is depicted in Figure 4. As can be seen, nodes numbered 1 and 2 are both node u's neighbors. Supposing $w_{1,\ u}$ equals to $w_{2,\ u}$, since the number of neighbors of node 1 is greater than that of node 2, node 1 is more important than node 2, which reflects in the different thickness of the two edges.

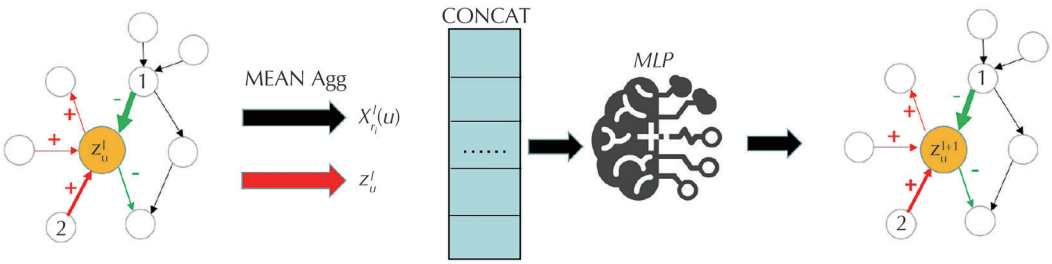

Figure 4. The extended SDGNN model.

Based on the above process, we can obtain the embedding of each node which can be applied into DQN.

3.4 Reinforcement Learning Model

3.4.1 The motivation of choosing Deep Q-network

Limitations of traditional influence maximization methods. Conventional approaches to influence maximization primarily fall into two categories: approximation algorithms and heuristic strategies. While approximation methods offer provable performance guarantees, they often involve intensive computations, making them unsuitable for large-scale or time-constrained environments. On the other hand, heuristic algorithms—although computationally more efficient—are typically tailored to specific diffusion models and lack adaptability to dynamic network changes. Furthermore, both approaches generally require

re-optimization or manual adjustment when the network structure evolves, limiting their scalability and responsiveness in real-world applications.

Suitability of reinforcement learning for combinatorial optimization. Influence maximization is fundamentally a combinatorial optimization problem, where the objective is to select an optimal subset of nodes from a large candidate set. Deep reinforcement learning (DRL) is particularly well-suited to such problems due to its ability to learn decision-making policies through trial-and-error interactions with the environment. Once trained, a DRL agent can make seed selection decisions efficiently without the need to rerun the optimization process, making it especially advantageous in large-scale or dynamic network settings.

No requirement for labeled data. In contrast to supervised learning, which depends on large quantities of annotated data, reinforcement learning operates in environments where learning is driven by reward signals. This makes DRL particularly attractive in the influence maximization domain, where ground-truth labels are generally unavailable. The agent learns from simulated diffusion outcomes, adjusting its policy over time to maximize positive influence—without requiring external supervision or manually labeled data.

3.4.2 Deep Q Network

Deep Q Network (DQN) refers to a deep-learning-based Q-learning algorithm, whose key points are how to represent action, state and reward. Combined with the PIM problem, we have the following definitions.

1. Representation of the state s. We use a vector z_u as the embedding of each node u, and the embedding of the currently selected seed nodes set is summed up as the current state as Equation 7.

$$s = \sum_{u \in S} z_u \tag{7}$$

2. Representation of action a. In the IC-P model, an action is represented as "select a node and add it to the set of selected seed nodes". After executing action a, such as selecting node u, z_u is added to the current state to get the new state as shown in Equation 8.

$$s_{i+1} = s_i + z_u \tag{8}$$

3. Representation of reward r. We define $r(s_i, a)$ as the increased value of positive influence from state s_i to new state s_{i+1} by making action a. Since the activating process has randomness, we obtain the estimated value of influence with high probability by Monte-Carlo simulations. Let r_i represents the positive influence obtained by seed nodes set at state s_i, and r_{i+1} represents that at s_{i+1}, then $r(s, a)$ can be expressed as Equation 9.

$$r(s, a) = r_{i+1} - r_i \tag{9}$$

To better tackle the dilemma between exploration and exploitation, we adopt the ε-greedy strategy which means that DQN has the probability of ε to select a new node randomly from nodes that are not in current selected nodes set, and the probability of $1-\varepsilon$ to select the node whose Q value is the biggest. The ε-greedy strategy is formulated as Equation 10.

$$a^* = \begin{cases} random() & \varepsilon \\ \arg\max_{a \in V - S_i} Q(s_i, a) & 1-\varepsilon \end{cases} \tag{10}$$

When it comes to the training of DQN, there are two different ideas: Monte-Carlo based and Temporal-Difference Based. In this paper, we adopt the later one. We define Q as the predict network where θ is the parameters of the neural network and it can be trained by minimizing the loss function formulated in Equation 11.

$$\begin{gathered} L(\theta) = \left[r(s_i, a) + \gamma * Q'(s_{i+1}, a'; \theta') - Q(s_i, a; \theta) \right]^2 \\ a' = \arg\max_{t \in (V - S_{i+1})} Q(s_{i+1}, t) \end{gathered} \tag{11}$$

where, Q' is target network with the parameters of θ', which copy from θ regularly. To diminish the overestimate of Q value, we utilize Double-DQN (DDQN) technology [29] where the selection of a' depends on predict network Q rather than target network Q'. After training DQN, the strategy to select nodes in sequence is greedy-based, *i.e.*, scoring each action. It makes an action by selecting the node that is not in the currently selected nodes set and has the highest score, which can be seen in Equation 12.

$$v^* = \arg\max_{v \in (V - S_i)} Q(s_i, v) \tag{12}$$

The whole work process of DQN is illustrated as Algorithm 1.

4. EXPERIMENTS

In this section, we introduce our datasets firstly and then discuss our experiment in detail. Subsequently, some useful tricks for experiments are presented. All the experiments are carried out on the machine with the following configurations: seven cores Intel(R) Xeon(R) CPU E5-2680 v4 2.40GHz, 16GB RAM, Titan Xp, Linux system.

4.1 Experimental setup

In this part, we will introduce the setup of our experiment, including the datasets, the diffusion model we adopt, the parameters of our model, the training issues and the involving algorithms.

Algorithm 1. Pre-training based DQN for node selection.

Require:

network $G = (V, E, W)$; a positive number k (the size of set S); embedding $Z \in \mathbb{R}^{|V| \times d}$; other hyper parameters: *episode*, *batchsize*, C

Ensure:

The weight of eval-network θ

for all u such that $u \in V$ **do**
  Use node u as the seed node under IC-P diffusion process;
  Calculate the influence r of node u using Monte-Carlo simulation;
  Store transition $(s_0, u, r, s_1, \{u\})$ to D;
end for
for $i = 1$ to *episode* **do**
  Clear the seed set S;
  Set $lastR = 0$ to record the last total influence;
  Initialize the state to empty $s_0 = (0, 0 \ldots 0) \in \mathbb{R}^d$;
  for $j = 1$ to k **do**
    Select node u_j by ε-greedy policy;
    Add u_j to current seed set $S_j \leftarrow S_{j-1} \cup u_t$;
    Update State $s_j = s_{j-1} + z_{uj}$;
    Calculate the influence of S_j using Monte-Carlo simulation, marked as $now\boldsymbol{R}$;
    Let $r_j = now\boldsymbol{R} - last\boldsymbol{R}$ and set $last\boldsymbol{R} = now\boldsymbol{R}$;
    Store transition $(s_{j-1}, u_j, r_j, s_j, S_j)$ to D;
    if $j \geq 2 *$ *batchsize* **then**
      Sample *batchsize* transitions randomly from D;
      Learn and update θ by Adam;
      Set $\theta' = \theta$ every C steps;
    end if
  end for
end for

4.1.1 Dataset

We use four datasets to carry out experiments: Bitcoin-Alpha (Alpha)[②], Bitcoin-OTC (OTC)[③], Slashdot[④], Advogato[⑤]. The four datasets are originally weighted directed graphs with edge weights between [–10, 10]. Due to the large edge weights, the number of diffusion is large and the training time is long. Therefore, we statistically scale the edge weights to [–0.1, 0.1]. The details of the four datasets are shown in Table 1.

Table 1. Summary of datasets.

Name	Nodes	Edges	Type	Percentage of positive edges
Alpha	3783	24186	Directed, sparse	93%
OTC	5881	35592	Directed, sparse	89%
Slashdot	4999	128760	Directed, sparse	77%
Advogato	6551	51332	Undirected, sparse	80%

4.1.2 Diffusion Model

As our application scenario is in signed network, we adopt the IC-P model.

4.1.3 Parameter Settings

For our methods, we choose a 20-dimensional vector as the embedding for each node. We set the total experience capacity to 50,000 and the batch size to 2,048. In addition, we set the total number of training rounds to *max* (15,000, $|V|$ + 10,000), where $|V|$ represents the total number of nodes. We use the Adam optimizer and set the learning rate to 0.001. For DQN training, we set the decay factor γ in temporal difference strategy to 0.95 and set the update frequency of the network to $|V|$ * 3500. To solve the exploration-exploitation dilemma, we use a decaying ε, which makes the DQN biased towards exploring more new states in the early training rounds and biased towards using the learned experience in the later rounds. And the range of ε-decay is from 0.95 to 0.02. To avoid over-fitting, we adopt the early-stop strategy, which stops training when the running episode is greater than $|V|$ * 3500 and the average influence of the last 50 episodes is 5% more than the influence of CELF algorithm.

4.1.4 Training issues

For better training, we use some tricks summarized as follows:

② http://snap.stanford.edu/data/soc-sign-bitcoin-alpha.html.
③ http://snap.stanford.edu/data/soc-sign-bitcoin-otc.html.
④ https://slashdot.org/.
⑤ https://networkrepository.com/soc-advogato.php.

- Fix network. When we interact with the environment, in order to ensure that r is monotonically increasing when selecting nodes and to reduce the computation, we fix the network before selecting seed nodes. The method is as follows: for each edge, we keep it if and only if the edge weight is larger than a random float number ranging between 0 and 1 and then set the weights of all the retained edges as 1. In other words, the new network is a spanning network of the original one. As a result, the activation process becomes deterministic as the activation probabilities of all edges are 100%.
- Pre-training. We first store transition $(s_0, u, r_0, s_1, \{u\})$ for each node u in the network, *i.e.*, we will store $|V|$ transitions in total before training. In the next 200 training rounds, we use these experiences to train our network. In this way, the estimate of the first node will be much more accurate than no pre-training, which lays the foundation for the subsequent training.

4.1.5 Used methods

In our experiments, we compare the effects of the following six algorithms:

- Our deep reinforcement learning model.
- The greedy algorithm [2].
- The CELF algorithm [3].
- Out-Degree. It selects k nodes with the largest out degrees from V.
- Positive Out-Degree. The difference between this method and Out-Degree is just considering the positive out degree.
- Random. This method selects k nodes from V randomly.

And we choose the greedy algorithm and the CELF algorithm as baselines.

4.2 Experimental Results

To comprehensively evaluate the performance of our proposed DRL-based method, we conduct experiments on four real-world signed social networks: Bitcoin-Alpha, Bitcoin-OTC, Slashdot, and Advogato. The evaluation focuses on two key aspects: (1) the quality of influence spread, and (2) computational efficiency.

Baseline Selection and Runtime Estimation. We compare our method against five representative baselines: Greedy, CELF, Out-Degree, Positive Out-Degree, and Random. Due to the well-known computational intensity of the Greedy algorithm, and following Leskovec et al. [3], who demonstrated that CELF can be up to 700 times faster than Greedy, we implement Greedy in C++ and use CELF's runtime as a proxy reference for Greedy's expected computational cost. For Out-Degree, Positive Out-Degree, and Random methods, the runtime is negligible since they only require simple node sorting based on degree or polarity.

Influence Spread Results. Figure 5 presents the influence spread results on the four datasets. Our model consistently achieves competitive or superior influence propagation compared to all baselines. Notably:

- The Random baseline performs the worst across all networks, serving as a valid lower-bound reference.
- Out-Degree and Positive Out-Degree exhibit similar performance due to the high ratio of positive edges (e.g., 93% in Alpha and 89% in OTC), which diminishes the distinction between total and positive degree.
- Our DQN-based method outperforms the baselines in most cases when the number of selected seed nodes is $\geqslant 30$. When the seed set is small (e.g., $\leqslant 20$), performance differences are less significant due to sparse diffusion and reward sparsity in training.

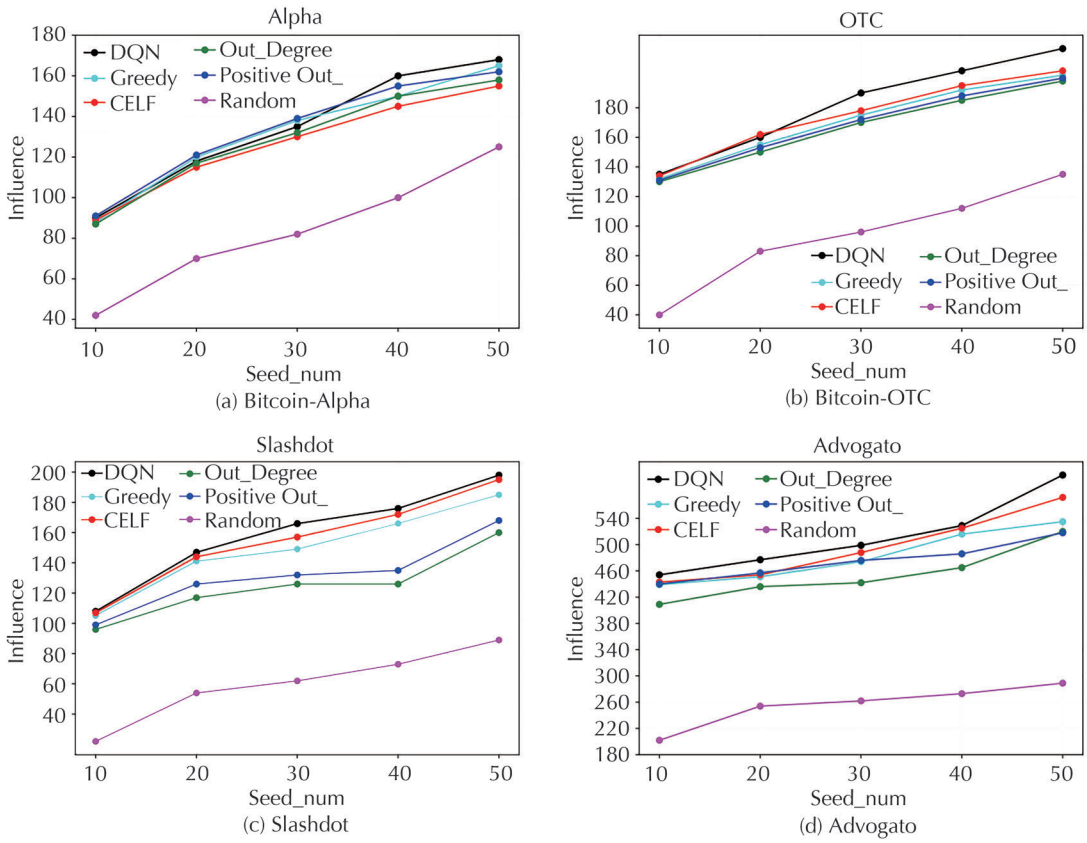

Figure 5. Positive influence spread across four signed social networks.

Although our DRL-based method does not consistently outperform the Greedy algorithm in terms of influence spread on the Bitcoin-Alpha dataset, further analysis reveals that Alpha is a relatively sparse

and structurally simple network. Such characteristics favor the Greedy algorithm, which leverages precise marginal gain computations and performs particularly well in less complex topologies. In contrast, deep reinforcement learning methods are designed to generalize across diverse network conditions and tend to exhibit greater advantages in scenarios involving higher structural complexity, denser connectivity, or temporal dynamics—where traditional methods often encounter limitations. Importantly, on the other three datasets (OTC, Slashdot, and Advogato), our method achieves competitive or superior influence spread, demonstrating its stronger adaptability and robustness in broader applications.

Runtime Comparison. As shown in Figure 6, our method demonstrates a significant advantage in computational efficiency. As the seed-set size increases, our method generally exhibits a substantial computational advantage over CELF across the four datasets. It completes inference much faster than CELF, while achieving comparable or higher influence spread. This highlights the practical value of our approach in real-time or large-scale scenarios, where traditional Greedy-based methods become computationally prohibitive.

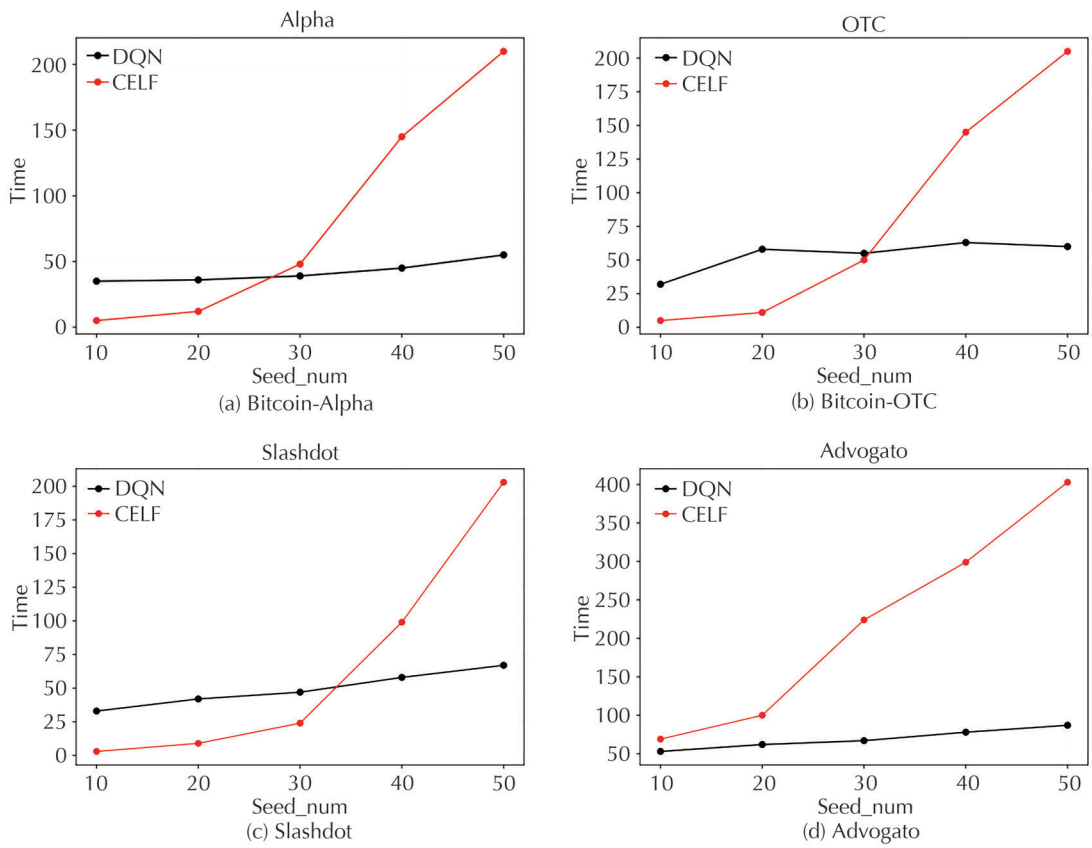

Figure 6. Running time across four signed social networks.

Summary. In summary, our model demonstrates comparable or superior influence spread compared to baseline methods, while significantly reducing computational time, particularly when the seed set size is relatively large. These results validate the practical value of the proposed approach in terms of both effectiveness and efficiency.

5. CONCLUSION

This paper presents a deep reinforcement learning-based framework for solving the problem of positive influence maximization in signed social networks. By combining polarity-aware diffusion modeling, SDGNN-based network embedding, and DQN-driven seed selection, our approach effectively captures the structural and semantic features of signed relationships. Experimental results on four real-world signed networks show that the proposed model consistently achieves competitive or superior influence spread while offering substantial runtime improvements over traditional methods such as CELF and Greedy. The model exhibits strong adaptability across networks of varying sparsity and polarity. Future work will focus on further improving performance in small-seed scenarios and extending the framework to large-scale dynamic signed networks.

ACKNOWLEDGEMENTS

This work is funded by the National Natural Science Foundation of China (No. 62376136 and No. 62076149), Guangdong Basic and Applied Basic Research Foundation (No. 2024A1515011935).

AUTHOR CONTRIBUTIONS

Longxiang Li: Writing-original draft, Validation. Yanhao Sun: Validation, Formal analysis, Data curation. Tianwei Lin: Formal analysis, Validation. Nuan Song: Writing-review & editing, Data curation, Conceptualization, Software, Validation. Liming Wu: Software, Validation. Tianqi Sun: Software, Validation. Wei Si: Conceptualization. Yongjian Zhao: Conceptualization. Dong Li: Writing-review & editing, Conceptualization, Supervision, Methodology.

REFERENCES

[1] P. M. Domingos and M. Richardson, "Mining the network value of customers," in *Proc. 7th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD '01)*, 2001, pp. 57–66, doi: 10.1145/502512.502525.

[2] D. Kempe, J. Kleinberg, and É. Tardos, "Maximizing the spread of influence through a social network," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD '03)*, 2003, pp. 137–146, doi: 10.1145/956750.956769.

[3] J. Leskovec, A. Krause, C. Guestrin, C. Faloutsos, J. M. VanBriesen, and N. Glance, "Cost-effective outbreak detection in networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD '07)*, 2007, pp. 420–429, doi: 10.1145/1281192.1281239.

[4] D. Li, Z.-M. Xu, N. Chakraborty, A. Gupta, K. Sycara, and S. Li, "Polarity related influence maximization in signed social networks," *PLoS ONE*, vol. 9, no. 7, Art. no. e102199, 2014, doi: 10.1371/journal.pone.0102199.

[5] L. Qiu, S. Zhang, and J. Yu, "Positive influence maximization in the signed social networks considering polarity relationship and propagation probability," *Int. J. Softw. Eng. Knowl. Eng.*, vol. 31, no. 2, pp. 249–267, 2021, doi: 10.1142/S0218194021500078.

[6] D. Li, C. Wang, S. Zhang, G. Zhou, D. Chu, and C. Wu, "Positive influence maximization in signed social networks based on simulated annealing," *Neurocomputing*, vol. 260, pp. 69–78, 2017, doi: 10.1016/j.neucom.2017.03.003.

[7] X. Yin, X. Hu, Y. Chen, X. Yuan, and B. Li, "Signed-PageRank: An efficient influence maximization framework for signed social networks," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 5, pp. 2208–2222, 2021, doi: 10.1109/TKDE.2019.2947421.

[8] W. Ju, L. Chen, B. Li, W. Liu, J. Sheng, and Y. Wang, "A new algorithm for positive influence maximization in signed networks," *Inf. Sci.*, vol. 512, pp. 1571–1591, 2020, doi: 10.1016/j.ins.2019.10.061.

[9] S. Kumar, A. Mallik, and B. S. Panda, "Influence maximization in social networks using transfer learning via graph-based LSTM," *Expert Syst. Appl.*, vol. 212, Art. no. 118770, 2023, doi: 10.1016/j.eswa.2022.118770.

[10] H. Li, M. Xu, S. S. Bhowmick, C. Sun, Z. Jiang, and J. Cui, "DISCO: Influence maximization meets network embedding and deep learning," *arXiv preprint arXiv:1906.07378*, 2019, doi: 10.48550/arXiv.1906.07378.

[11] J. Huang, H. Shen, L. Hou, and X. Cheng, "SDGNN: Learning node representation for signed directed networks," in *Proc. 35th AAAI Conf. Artif. Intell. (AAAI-21)*, vol. 35, no. 1, 2021, pp. 196–203, doi: 10.1609/aaai.v35i1.16093.

[12] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD '09)*, 2009, pp. 199–208, doi: 10.1145/1557019.1557047.

[13] L. Cui *et al.*, "DDSE: A novel evolutionary algorithm based on degree-descending search strategy for influence maximization in social networks," *J. Netw. Comput. Appl.*, vol. 103, pp. 119–130, 2018, doi: 10.1016/j.jnca.2017.12.003.

[14] Q. He, X. Wang, Z. Lei, M. Huang, Y. Cai, and L. Ma, "TIFIM: A two-stage iterative framework for influence maximization in social networks," *Appl. Math. Comput.*, vol. 354, pp. 338–352, 2019, doi: 10.1016/j.amc.2019.02.056.

[15] S. Kianian and M. Rostamnia, "An efficient path-based approach for influence maximization in social networks," *Expert Syst. Appl.*, vol. 167, Art. no. 114168, 2021, doi: 10.1016/j.eswa.2020.114168.

[16] T. K. Biswas, A. Abbasi, and R. K. Chakrabortty, "An MCDM integrated adaptive simulated annealing approach for influence maximization in social networks," *Inf. Sci.*, vol. 556, pp. 27–48, 2021, doi: 10.1016/j.ins.2020.12.048.

[17] J. Li, T. Cai, K. Deng, X. Wang, T. Sellis, and F. Xia, "Community-diversified influence maximization in social networks," *Inf. Syst.*, vol. 92, Art. no. 101522, 2020, doi: 10.1016/j.is.2020.101522.

[18] S. Kumar, L. Singhla, K. Jindal, K. Grover, and B. S. Panda, "IM-ELPR: Influence maximization in social networks using label propagation based community structure," *Appl. Intell.*, vol. 51, no. 11, pp. 7647–7665, 2021, doi: 10.1007/s10489-021-02266-w.

[19] A. Zareie, A. Sheikhahmadi, and K. Khamforoosh, "Influence maximization in social networks based on TOPSIS," *Expert Syst. Appl.*, vol. 108, pp. 96–107, 2018, doi: 10.1016/j.eswa.2018.05.001.

[20] H.-H. Wu and S. Küçükyavuz, "A two-stage stochastic programming approach for influence maximization in social networks," *Comput. Optim. Appl.*, vol. 69, no. 3, pp. 563–595, 2018, doi: 10.1007/s10589-017-9958-x.

[21] H. Dai, B. Dai, and L. Song, "Discriminative embeddings of latent variable models for structured data," in *Proc. 33rd Int. Conf. Mach. Learn. (ICML)*, PMLR, vol. 48, 2016, pp. 2702–2711. [Online]. Available: https://proceedings.mlr.press/v48/daib16.html.

[22] C. Wang, Y. Liu, X. Gao, and G. Chen, "A reinforcement learning model for influence maximization in social networks," in *Database Systems for Advanced Applications (DASFAA 2021)*, Lecture Notes in Computer Science, vol. 12682, 2021, pp. 701–709, doi: 10.1007/978-3-030-73197-7_48.

[23] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, 2018. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ.

[24] J. Sheng, L. Chen, Y. Chen, B. Li, and W. Liu, "Positive influence maximization in signed social networks under independent cascade model," *Soft Comput.*, vol. 24, no. 19, pp. 14287–14303, 2020, doi: 10.1007/s00500-020-05195-x.

[25] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD '16)*, 2016, pp. 855–864, doi: 10.1145/2939672.2939754.

[26] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, 2017. [Online]. Available: https://openreview.net/forum?id=SJU4ayYgl.

[27] L. Ma, Y. Ma, Q. Lin, J. Ji, C. A. Coello Coello, and M. Gong, "SNEGAN: Signed network embedding by using generative adversarial nets," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 6, no. 1, pp. 136–149, Feb. 2022, doi: 10.1109/TETCI.2020.3035937.

[28] Y.-C. Lee, N. Seo, K. Han, and S.-W. Kim, "ASiNE: Adversarial signed network embedding," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval (SIGIR '20)*, 2020, pp. 609–618, doi: 10.1145/3397271.3401079.

[29] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with Double Q-learning," in *Proc. 30th AAAI Conf. Artif. Intell. (AAAI-16)*, vol. 30, no. 1, 2016, pp. 2094–2100, doi: 10.1609/aaai.v30i1.10295.