# $A^2S^2$-GNN: Rigging GNN-Based Social Status by Adversarial Attacks in Signed Social Networks

Xiaoyan Yin, *Member, IEEE*, Wanyu Lin, *Member, IEEE*, Kexin Sun, Chun Wei, and Yanjiao Chen, *Senior Member, IEEE*

***Abstract*— Social status, the social influence of a user, plays an important role in many real-world applications, e.g., trust relations and information propagation in a social network. In this paper, we reveal the possibility of falsifying social status through adversarial attacks in graph neural networks (GNNs). Different from neural networks in the visual or speech domain, GNNs take the attributes of nodes and edges in a graph as features. To cater to the characteristics of GNNs, we design a new paradigm of adversarial example attack, named $A^2S^2$-GNN (GNN-based Adversarial Attacks on Social Status), aiming at manipulating the social status of a target node in social networks. The key idea is to establish relationships or break relationships between a set of compromised nodes and the target node. More specifically, we consider a signed directed graph representing complicated positive/negative asymmetric relationships between nodes. We design an efficient adversarial attack algorithm to determine the minimum set of signed links that should be created or deleted to reach the attack objective. We conduct extensive experiments on baseline datasets. Compared with the benchmark algorithms, $A^2S^2$-GNN can effectively promote or vilify the social status of the target node up to 89.36% and 192.38%, respectively, while keeping the modification to the social network to the minimum. Furthermore, the experimental results on six status evaluating algorithms verify the transferability of our proposed attack algorithm.**

***Index Terms*— Social computing, adversarial machine learning, white-box attack, graph neural networks.**

## I. Introduction

SOCIAL status [1] is the reflection of all kinds of social relations. Generally, a person acknowledged by others with high influence has high social status. Social status plays an important role in the formation of trust relations [2], [3], [4], [5], [6] and is particularly vital for information spreading to the whole social network [7]. More specifically, if high-status users rather than low status-users are selected as seed nodes, the scope and speed of information propagation will be greatly improved. Up to now, a line of works have been proposed to predict social status in a social network based on graph neural networks (GNNs) [8].

In recent years, various adversarial attacks have been proposed based on GNNs, most of which focus on classification tasks in unsigned or undirected graphs, e.g., link prediction [9], [10], node embedding [11], [12], [13], node classification [14], [15], [16], community detection [17] and graph classification [18]. Deep learning models, e.g, graph convolutional networks [19] and graph neural networks [14] can achieve good performance on graph data. As for social status prediction, Dai et al. [20] improved the credit of a target user by adding a connection between the user and an AAA-class user in a credit evaluation system. Li et al. [17] attacked the community detection algorithm by deleting links, such that the target user can hide from the real community and become a member of the desired community. However, there is a lack of works on manipulating social status in GNNs, especially for the complicated signed directed graphs. Higher social status implies higher visibility to others in online social platforms. Therefore, the manipulation of social status is of great significance, e.g., the official can dispel net-mediated public sentiment effectively and efficiently by maneuvering the social status of the rumor source, who is tracked down by the Internet public opinion monitoring system. In essence, the spread of rumors is demolished by reducing the visibility of rumor disseminators. Similarly, the official can improve the spreading effect of positive energy by increasing the social status of seed nodes.

A social network is usually modeled as a graph with the nodes denoting users and the edges denoting relationship between nodes. To characterize complicated asymmetric trust/distrust and like/dislike relationships (or friends/foes interchangeably) among users, it is necessary to use signed directed graphs. Firstly, different from undirected graphs that regard the relationship between two nodes as mutually the same, it is more realistic for directed graphs to differentiate relationships at different directions. For instance, a common user may like a celebrity user while the celebrity user may not return the favor. Secondly, signs of edges is important for social status evaluation since different signs will lead to different results. For instance, a positive link (with sign $+$)

Manuscript received 10 March 2022; revised 10 August 2022 and 22 October 2022; accepted 22 October 2022. Date of publication 3 November 2022; date of current version 7 December 2022. The work was supported in part by the National Natural Science Foundation of China under Grant 61872295 and in part by the Shaanxi Natural Science Foundation under Grant 2020JM-416. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Dusit Niyato. *(Corresponding author: Yanjiao Chen.)*

Xiaoyan Yin and Chun Wei are with the School of Information Science and Technology, Northwest University, Xi'an 710127, China, and also with the Shaanxi International Joint Research Centre for the Battery-Free Internet of Things, Xi'an 710127, China (e-mail: yinxy@nwu.edu.cn; weichun@stumail.nwu.edu.cn).

Kexin Sun is now with BYD Company, Xi'an 710065, China (e-mail: sunkexin@stumail.nwu.edu.cn).

Wanyu Lin is with the Department of Computing, The Hong Kong Polytechnic University, Hong Kong, China (e-mail: wan-yu.lin@polyu.edu.hk).

Yanjiao Chen is with the College of Electrical Engineering, Zhejiang University, Hangzhou 310007, China (e-mail: chenyanjiao@zju.edu.cn).

Digital Object Identifier 10.1109/TIFS.2022.3219342

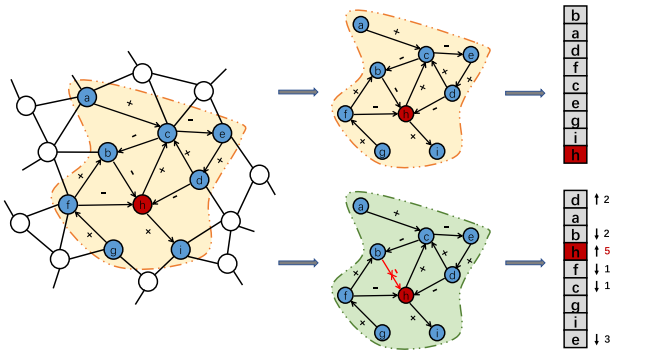



Fig. 1. An example of Status Attack. After deleting the edge from node $b$ to node $h$, $d$'s and $h$'s status have been raised by 2 and 5, respectively, meanwhile, $b$'s, $f$'s and $c$'s status have been lowered by 2, 1 and 1, respectively.

from a highly-influential user $A$ to $B$ will increase the social status of $B$ while a negative link (with sign $-$) from user $A$ to $B$ may decrease the social status of $B$. For the similar reason, a positive link (with sign $+$) from a lesser user $A$ to $B$ may sometimes decrease the social status of $B$, and vice versa.

In this paper, we consider the possibility of tampering with social status prediction by GNNs through white-box adversarial attacks. Nonetheless, to succeed in manipulating a signed directed graph in GNNs is challenging due to various reasons. First, different from learning in the visual or speech domain, the features in GNNs are attributes of nodes and links in the graph. While pixels and voices may be modified freely in a continuous manner, many features of graphs are discrete, e.g., the degree of nodes, and the sign of a link. Moreover, it is easy to constrain the modifications in image/speech adversarial examples by simply comparing the pixel values and voice signals. In contrast, the difference between the two social networks is hard to quantify. Therefore, how to conduct the modification of the graph and how to restrict the modification need to be redefined. This leads to the need for new ways to characterize the modification of graph. Second, for GNNs, signed directed graphs are much more difficult than unsigned or undirected graphs, as the feature space is expanded. Because of the asymmetric relationship between nodes and the different influences of nodes with similar status but different hops on a specific node, the corresponding feature space is more complicated than unsigned or undirected graphs. This makes it difficult to represent the social status in social networks in GNNs. A commonly-used strategy for modifying social networks is changing the network topology, e.g., adding or deleting edges, which is easy in unsigned or undirected graphs. In signed directed graphs, modifying a positive/negative edge not only influence the two nodes linked by the edge but propagate the influence to almost all nodes in the network. As shown in Fig. 1, deleting the negative connection from user b to user h causes the status change for multiple users. Thus, it is very difficult to formulate the impact of neighbor nodes on the social status of the specific node and the impact of attacks on the change of social status.

To address these challenges, we propose an **A**dversarial **A**ttack on **S**ocial **S**tatus based on GNNs, named $A^2S^2$-GNN, which aims to cheat the GNN model in evaluating the social status of a target node. To solve the first challenge, we restrict the modification of the adversary to adding or deleting edges between the target node and a set of nodes controlled by the adversary. We establish an optimization problem that targets at maximizing the changes of the social status of the target user while restricting the changes of the social status of all other users, which fulfills the imperceptibility characteristic of adversarial example attacks. To deal with the second challenge, we build a social status score learning model, which leverages a GNN to learn the social score for every user in the social network based on the features of the graph that represents the social network. The loss function for the learning process minimizes the difference between predicted social scores of nodes and corresponding social scores of nodes computed based on the status theory. Based on our social status evaluation results, we propose an attack algorithm that iteratively identifies the edges that should be added or removed between the compromised nodes and the target node.

We evaluate the performance of $A^2S^2$-GNN on seven most commonly-used datasets of signed directed graphs. By comparing $A^2S^2$-GNN with four baselines, we show that with the same budget constraint, $A^2S^2$-GNN is able to change the social status of the target node much more than the baselines. Experiments also show that $A^2S^2$-GNN has relatively lower overhead than baselines.

Our contributions are summarized as follows.

- We make the first attempt to launch an white-box adversarial attack on social status based on GNNs in social networks.
- We design a series of algorithms to facilitate the attack by accurately evaluating the influence of the attack strategy (modifying the network topology) on the status scores of not only the target nodes but all nodes in the social network.
- We evaluate the performance of $A^2S^2$-GNN using 7 public real-world datasets from various applications. Compared to four classical baselines, the experiment results validate the effectiveness and efficiency of $A^2S^2$-GNN.

The remainder of this paper is organized as follows. We introduce the threat model in Section II. The target problem is defined in details in Section III. The attack framework is described in Section IV. Simulation results are presented in Section V. We survey the related works in Section VI. Finally, we conclude this paper in Section VII.

## II. Threat Model

We describe the threat model with a white-box attack in terms of the adversary's knowledge, capability and goal.

### A. Knowledge

Under the white-box attack, we assume that the adversary knows the structure and the adjacency matrix of the directed signed social network. This is feasible since many social networks publicize the friendship relations between users. The adversary also knows that the status evaluation model is built on the status theory [6] in order to evaluate the attack performance and the influence of the attack on the social status.

### B. Capability

The adversary has control over a set of nodes in the signed directed network. To change the status of the target user, the adversary can carry out an adversarial attack to generate an adversarial graph by perturbing the graph structure, e.g., deleting links between the target node and its neighbors, adding links between the target node and its two-hop neighbors,[1] or rewiring links related to the target node.[2] This mimics the start or the end of friendship between users in signed directed social networks. In this paper, we focus on perturbing the interactions between users rather than the users themselves. To evaluate the different effects from different neighbors on the status of the target node, the adversary can distinguish between positive influence nodes and negative influence neighbors. Furthermore, the adversary can build a status evaluation model based on the status theory to calculate the status score for every node, and then rank all nodes in terms of the status score.

### C. Goal

The adversary's goal is to change the social status of the target node by imperceptible perturbations,[3] i.e., to promote the status of the target node by deleting edges between the target node and its negative influence nodes, or adding edges between the target node and its positive influence nodes, and vice versa. Intuitively, imperceptible perturbation constraints include the following three aspects: (i) imperceptible status changes of other nodes. The primary aim of the adversary is to maximize the status change (status upgrading/degrading) of the target node with little status alteration of other nodes in the network at the same time. (ii) imperceptible adversarial attack on graph. This implies that the adversarial graph is indistinguishable from the original graph in terms of the number of edges, the degree of nodes and so on. (iii) imperceptible changes in network structure. This means that there are limited changes (constrained number of operations for link deletion or addition) from the original graph to attain the final perturbed graph.

## III. Problem Definition

We consider a signed directed social network, denoted as $G = (V, E^+, E^-)$, to model the real-world entities and their relationships, where $V$ is the set of individuals, and $E^+$, $E^-$ are the set of positive links as well as the set of negative links, respectively. We have $E = E^+ \cup E^- \subseteq V \times V$ and $E^+ \cap E^- = \emptyset$. The positive edges imply like/trust relationships (or friends interchangeably) between nodes, and the negative edges indicate dislike/distrust relationships (or foes interchangeably) between nodes. We use a $|V| \times |V|$ adjacency matrix $A$ to denote the interactions between users, where $|V|$ is the cardinality of $V$. $A_{ij} = 1$ if $(i, j) \in E^+$. $A_{ij} = -1$ if $(i, j) \in E^-$. $A_{ij} = 0$ otherwise.

The adversarial attack on social status aims to promote or lower the status of the target user via changing the structure of the graph. Corresponding to the above three imperceptibility constraints, we first quantify the change of node status. Given target user $u$, let $A'$ denote the perturbed adjacency matrix after attack, $F(\cdot)$ denote the status evaluation model, and $R(F(\cdot))$ is the ranking order of a specific node based on status score provided by $F(\cdot)$. Thus, mathematically, the first imperceptibility constraint can be defined as

$$\frac{\sum_{v \in V, v \neq u} |R(F_v(G', A')) - R(F_v(G, A))|}{|V| - 1} < |R(F_u(G', A')) - R(F_u(G, A))|, \tag{1}$$

where $|R(F_u(G', A')) - R(F_u(G, A))|$ is the status change for user $u$.

To measure the imperceptibility of adversarial attacks on graph, we focus on degree distribution of the network. We assess the changes in degree distribution based on the likelihood ratio test for the power-law degree distribution of the original graph $G$ and the modified graph $G'$ [14]. The perturbed graph $G'$ is acceptable only if the degree distribution satisfies

$$\Lambda(G, G') < \tau \approx 0.004, \tag{2}$$

where $\Lambda$ denotes the log-likelihood ratio test statistic according to the power-law degree distributions, which follow a $\chi^2$ distribution with one degree of freedom. $\tau$ is approximated using the critical $\rho - value$ setting in the $\chi^2$ distribution [9].

The evaluation of the imperceptibility of network structure alteration is founded on the change of the adjacency matrices. We use the $l_1$-norm distance of the adjacency matrices between the original graph $G$ and the perturbed graph $G'$ to describe the changes. Then, the imperceptible change in network structure can be expressed as

$$| A - A' | \leq \Delta, \tag{3}$$

where $\Delta$ is the budget for our adversarial attacks, i.e., the amount of deleting or adding links must not be larger than $\Delta$.

In the context of adversarial attacks on social status, our target problem is to upgrade/degrade status for a specific user. Given target user $u$, for status evaluation algorithm $F(\cdot)$, the adversarial attack problem on social status can be formulated as an optimization problem that aims to maximize the social status increment of the target node while keeping the perturbation within the imperceptibility constraints.

$$\begin{aligned} \max \quad & |R(F_u(G', A')) - R(F_u(G, A))|, \\ \text{s.t.} \quad & (1), (2), (3), \end{aligned} \tag{4}$$

where Eq. (1) specifies that the average status changes of other nodes in the social network is smaller than that of the target node, i.e., the target node has the largest social status change due to the adversarial attack, and Eq. (2) and Eq. (3) designate that the changes of the social network are within a limit.

We summarize the key notations in Table I.

[1] We consider adding links between the target node and its two-hop neighbors instead of any node, mainly because it is more feasible to become friends with friends of friends.

[2] In this paper, we make the simplifying assumption that rewiring a link is equivalent to deleting an existing link and then adding a corresponding new link. Therefore, rewiring links can be represented by adding and deleting links.

[3] We follow the existing definition of imperceptible perturbations [17]. The imperceptible perturbations mean that while achieving the attack goal, we should make the status change of other nodes as small as possible, modification of the graph as little as possible, and the number of modification operations as small as possible.

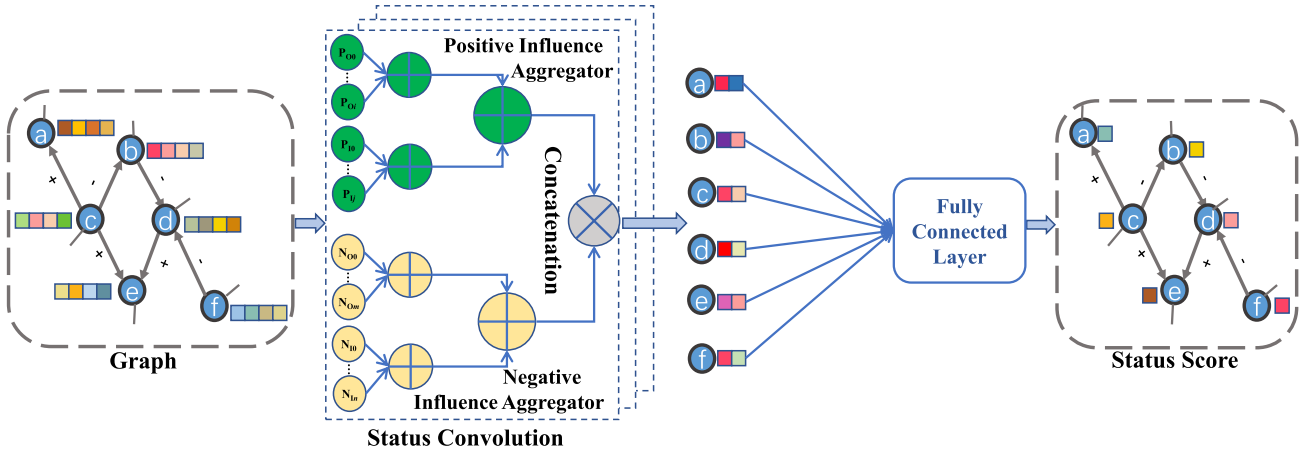



Fig. 2. Overview of our proposed status evaluation model. The model consists of three status convolution layers and a fully connected layer. Every status convolution layer leverages the influence of in-degree friends, out-degree friends, out-degree foes and in-degree foes on the social status of the target node, respectively, while the fully connected layer combines the positive influence and negative influence and obtain the status score for every node finally.

TABLE I
KEY NOTATIONS

| Notation | Definition |
|---|---|
| $E^+$ | the set of positive links |
| $E^-$ | the set of negative links |
| $G(V, E^+, E^-)$ | the original graph for a signed directed social network |
| $G'(V, E'^+, E'^-)$ | the corresponding adversarial graph |
| $\|V\|$ | the cardinality of V |
| $A$ | the adjacency matrix for graph $G$ |
| $A'$ | the perturbed adjacency matrix for graph $G'$ |
| $H_I(u)$ | the set of in-degree neighbours of node $u$ |
| $H_I^+(u)$ | the set of in-degree friends of node $u$ |
| $H_I^-(u)$ | the set of in-degree foes of node $u$ |
| $H_O(u)$ | the set of out-degree neighbors of node $u$ |
| $H_O^+(u)$ | the set of out-degree friends of node $u$ |
| $H_O^-(u)$ | the set of out-degree foes of node $u$ |
| $P(u)$ | positive influence for node $u$ |
| $P_I(u)$ | positive influence for $u$ from in-degree neighbours |
| $P_O(u)$ | positive influence for $u$ from out-degree neighbours |
| $N(u)$ | negative influence for $u$ |
| $N_I(u)$ | negative influence for $u$ from in-degree neighbours |
| $N_O(u)$ | negative influence for $u$ from out-degree neighbours |
| $C(u)$ | the status score of node $u$ |
| $R(u)$ | the status score-based ranking order of node $u$ |
| $V_{op}(u)$ | the set of operable nodes of node $u$ |

## IV. THE PROPOSED ATTACK FRAMEWORK

As stated in the problem definition, our adversarial attack includes two components: (1) a status evaluation algorithm $F(\cdot)$ that can calculate a status score for every user. (2) a greedy algorithm to solve the optimization problem described in section III. The two components are interconnected, $F(\cdot)$ can help verify effectiveness of the attack result of the greedy algorithm, while adversarial examples can improve robustness of status evaluation algorithm $F(\cdot)$.

In our framework, we design a GNN based status evaluation algorithm $F(\cdot)$ and a greedy algorithm based adversarial attack model. The adversarial attack model generates perturbed graph $G'$ under the control of the attack budget. The status evaluation model, which consists of four mean-aggregators and a full connection layer, as shown in Fig.2, iteratively computes the status scores for nodes. Furthermore, perturbed graph $G'$ can be optimized continuously with the aid of the calculation results of the status evaluation model.

In our framework, we calculate the status score of nodes according to their relationship with their neighbors. Specifically, positive influence from neighbors will upgrade the status score of the target node, while negative influence from neighbors will degrade its status score. At the same time, we use the difference in ranking to measure the status change of the target node, and convert the imperceptible perturbation into the attack budget. We then select the node that has the greatest impact on the status change of the target user, and generate the adversarial graph. Thus, our solution can also be applied to other influence related adversarial attack scenarios.

### A. Status Evaluation Model

According to the status theory [6] and the balance theory [19], people usually trust a user who has a higher status than themselves. More specifically, in signed directed graph $G$, nodes tend to establish positive links to nodes with higher social status and negative links to nodes with lower status. Therefore, the four types of relationships, as shown in Fig.3, between two nodes in a signed directed social network imply the ranking of the two corresponding nodes. For example, as node $v$ is an in-degree friend of $u$ in Fig. 3 (a), we can deduce that $u$ has higher social status than $v$. In other words, trust from one of in-degree friends will improve status ranking of a specific node. Similarly, the relationship between nodes in Fig. 3 (a) and Fig. 3 (c) will have a positive impact on the social status of node $u$, and the relationship between nodes in Fig. 3 (b) and Fig. 3 (d) will have a negative impact on the social status of node $u$.

We build a GNN-based status evaluation model to derive the status score for each node, as shown in Fig.2. Inspired by [21], we leverage the expanded node features (i.e., the negative and the positive influence) to generate node embedding, which can measure the local and the global importance of the node in the signed directed social network. Let $P(u)$ and $N(u)$

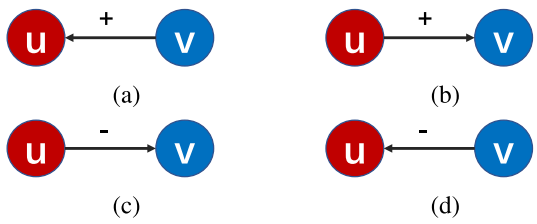



Fig. 3. Influence analysis of node $u$. (a) Positive influence from in-degree friend $v$. (b) Negative influence from out-degree friend $v$. (c) Positive influence from out-degree foe $v$. (d) Negative influence from in-degree foe $v$.

denote the positive and the negative influence of node $u$ from its neighbors, respectively. Let $H_I(u)$ and $H_O(u)$ denote the in-degree neighbors and the out-degree neighbors of $u$, respectively. To further distinguish the sign of edges, we use $H_I^+(u)$ and $H_I^-(u)$ to denote the in-degree neighbors of $u$ with positive (friends) and negative relationship (foes) with $u$, respectively. Similarly, $H_O^+(u)$ and $H_O^-(u)$ represent the set of out-degree friends and the set of out-degree foes, respectively.

*1) Influence-Based Status Aggregation:* Accordingly, given target node $u$, we use four mean aggregators to exploit influence on social status from its neighbors. Let $P_I(u)$, $N_O(u)$, $P_O(u)$, $N_I(u)$ denote the influence on node $u$ from $v$ corresponding to Fig.3 (a)-(d). Firstly, we take $[1\ 0]^T$ and $[0\ 1]^T$ as the one-hot encoding for positive edges and negative edges, respectively. Then we convert the one-hot representations into four corresponding dense vectors via a linear transformation. The latent factors of four types of neighbours can be computed as

$$k_{H_I^+(u)} = W_{H_I^+(u)} \cdot [1\ 0]^T, \tag{5}$$

$$k_{H_O^+(u)} = W_{H_O^+(u)} \cdot [1\ 0]^T, \tag{6}$$

$$k_{H_O^-(u)} = W_{H_O^-(u)} \cdot [0\ 1]^T, \tag{7}$$

$$k_{H_I^-(u)} = W_{H_I^-(u)} \cdot [0\ 1]^T, \tag{8}$$

where $W_{H_I^+(u)} \in \Re^{D_e \times 2}$, $W_{H_O^+(u)} \in \Re^{D_e \times 2}$, $W_{H_O^-(u)} \in \Re^{D_e \times 2}$, $W_{H_I^-(u)} \in \Re^{D_e \times 2}$ are four transformation matrices, and $D_e$ is the density of $u$'s feature vector.

Accordingly, the influence on status from $v$ to $u$ corresponding to Fig.3 (a)-(d) can be modeled as

$$P_I(u) = x[v] \otimes k_{H_I^+(u)}, \tag{9}$$

$$N_O(u) = x[v] \otimes k_{H_O^+(u)}, \tag{10}$$

$$P_O(u) = x[v] \otimes k_{H_O^-(u)}, \tag{11}$$

$$N_I(u) = x[v] \otimes k_{H_I^-(u)}, \tag{12}$$

where $\otimes$ represents the concatenation of $v$'s feature vector and the latent factor vector.

Then, to carry out the propagation and aggregation of social status throughout the network, we combine the four classes of influence in terms of positive influence and negative influence based on a linear approximation of a localized spectral convolution [22] as follows

$$C_P(u) = \frac{1}{|H_I^+(u)|} \sum_{v \in H_I^+(u)} P_I(u) + \frac{1}{|H_O^-(u)|} \sum_{v \in H_O^-(u)} P_O(u), \tag{13}$$

$$C_N(u) = \frac{1}{|H_I^-(u)|} \sum_{v \in H_I^-(u)} N_I(u) + \frac{1}{|H_O^+(u)|} \sum_{v \in H_O^+(u)} N_O(u), \tag{14}$$

where $C_P(u)$ and $C_N(u)$ can capture positive influence-based status and negative influence-based status for user $u$, respectively. We let $|H_I^+(u)|$, $|H_O^-(u)|$, $|H_I^-(u)|$ and $|H_O^+(u)|$ denote the number of four types of neighbours.

The performance of the status evaluation model largely depends on the latent factors. To achieve better performance, the negative and the positive influence from individual neighbors should not be considered separately. Therefore, we use a standard fully-connected layer to combine negative influence-based status and positive influence-based status.

$$C(u) = W \cdot (C_P(u) \otimes C_N(u)) + b, \tag{15}$$

where $C(u)$ is the result of the concatenated operation of $C_P(u)$ and $C_N(u)$, and we take $C(u)$ as input of the fully connected layer.

*2) Higher-Order Status Propagation:* Thanks to propagation and aggregation between users, we can capture the influence (both positive influence and negative influence) of $m$-hop neighbors on the status of user $u$ with $m$ convolutional layers. In the $m$-th iteration, the status score of $u$ is recursively calculated through Eq.(16)-Eq.(22) as

$$P_I^m(u) = C^{m-1}[v] \otimes \{W_{H_I^+(u)}^m \cdot [1\ 0]^T\}, \tag{16}$$

$$P_O^m(u) = C^{m-1}[v] \otimes \{W_{H_O^-(u)}^m \cdot [0\ 1]^T\}, \tag{17}$$

$$N_I^m(u) = C^{m-1}[v] \otimes \{W_{H_I^-(u)}^m \cdot [0\ 1]^T\}, \tag{18}$$

$$N_O^m(u) = C^{m-1}[v] \otimes \{W_{H_O^+(u)}^m \cdot [1\ 0]^T\}, \tag{19}$$

$$C_P^m(u) = \frac{1}{|H_I^+(u)|} \sum_{v \in H_I^+(u)} P_I^m(u) + \frac{1}{|H_O^-(u)|} \sum_{v \in H_O^-(u)} P_O^m(u), \tag{20}$$

$$C_N^m(u) = \frac{1}{|H_I^-(u)|} \sum_{v \in H_I^-(u)} N_I^m(u) + \frac{1}{|H_O^+(u)|} \sum_{v \in H_O^+(u)} N_O^m(u), \tag{21}$$

$$C^m(u) = W^m \cdot (C_P^m(u) \otimes C_N^m(u)) + b^m, \tag{22}$$

where $C^0[u] = x[u]$ is the feature vector of $u$ such that the information of the target node can also be aggregated, $W_{H_I^+(u)}^m$, $W_{H_O^-(u)}$, $W_{H_I^-(u)}^m$, $W_{H_O^+(u)}^m$ and $W^m$ are weight matrices, and $b^m$ is bias.

*3) Status Score Calculation:* Because of status convolutional layers involved, the scope of status propagation across the network can be controlled by regulating parameter $m$. Thanks to the small-world phenomenon (or six degrees of separation interchangeably) [23], we assume that the maximum number of hops between any two nodes of a network is 5, so five iterations are enough to propagation and then aggregate the impact of other nodes on the status of the target node. After five iterations, the status of all nodes will be stable,

and more iterations will not alter the status of nodes. Note that, stable status means that the ranking order of every node remains unchanged, but not the status score. Theoretically, status stability can be formulated as $\sum_v |R_{m+1}(v) - R_m(v)| = 0$, $\forall v \in V$, where $R_m(v)$ is the ranking order of node $v$ based on status score at the $m$-th iteration. In this paper, the termination condition of model learning is controlled by the loss function defined in the next subsection. Thus, the optimal number of convolutional layer-based status score can be calculated as

$$C(u) = W_{final} \cdot C^M(u) + b_{final}, \tag{23}$$

where $W_{final}$ and $b_{final}$ are the model parameters of the last convolutional layer, $M$ is the optimal number of convolutional layers that is also the minimum number of layers to reach status stability. We take C(u) as input of the fully connected layer. Then, the fully connected layer is utilized to calculate the status score for every user.

*4) The Loss Function:* We define a loss function to learn all parameters in $A^2S^2$-GNN. According to four types of relationships between users, the loss function consists of four terms, all of which enable the status evaluation model to learn the status relationship between nodes based on status theory. The first term is to minimize the status difference between in-degree foes and the target node. The second term is to minimize the status difference between out-degree friends and the target node. The third term is to maximize the status difference between in-degree friends and the target node. The fourth term is to maximize the status difference between out-degree foes and the target node. Mathematically, the loss function can be expressed as

$$\begin{aligned} L = & \frac{1}{|H_I^-(u)|} \sum_{v \in H_I^-(u)} [C(u) - C(v)] \\ & + \frac{1}{H_O^+(u)} \sum_{v \in H_O^+(u)} [C(u) - C(v)] \\ & - \frac{1}{|H_I^+(u)|} \sum_{v \in H_I^+(u)} [C(u) - C(v)] \\ & - \frac{1}{|H_O^-(u)|} \sum_{v \in H_O^-(u)} [C(u) - C(v)] + \lambda \cdot ||\Theta||_2^2, \end{aligned} \tag{24}$$

where $\Theta = \{\{W^m_{H_I^+(u)}, W_{H_O^-(u)}, W^m_{H_I^-(u)}, W^m_{H_O^+(u)}, W^m\}_{m=1}^M, W_{final}\}$ represent all weight matrices, and $\lambda$ controls the $L_2$ regularization intensity to avoid over-fitting.

### B. Adversarial Attack Model

Intuitively, there are many ways to perturb a graph, including adding edges, deleting edges and changing the properties of edges. To minimize the changes to the social network, we only perturb the graph by adding edges and deleting edges between the target node and its neighbours. We consider deleting harmful edges and adding beneficial edges to achieve the attack target. It is important to note that harmful edges are not negative edges, and beneficial edges are not positive edges. In Fig. 4, both the negative edge from $c$ to $d$ and the positive edge from $a$ to $c$ are positive influence edges for node $c$, because they both can upgrade status of node $c$. On the contrary, both the positive edge from $b$ to $c$ and the negative edge from $a$ to $b$ are negative influence edges, because they both can degrade status of node $b$.

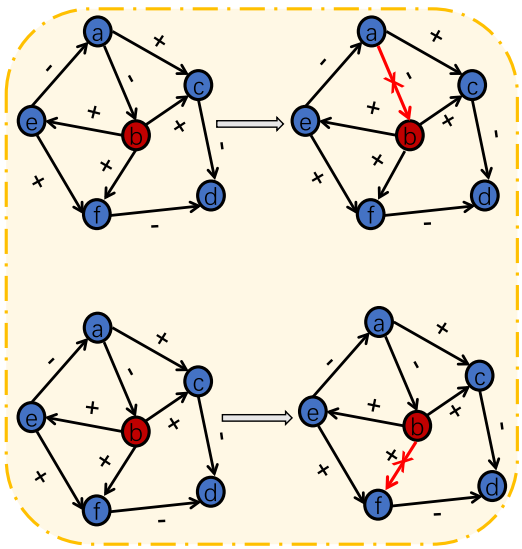



Fig. 4. Deletion of negative influence edge. Deleting negative influence edge can improve target node's status. The negative edge from node $a$ to node $b$ and the positive edge from node $b$ to node $f$ are both negative influence nodes for node $b$ in this figure.

Fig. 4 and Fig. 5 show a couple of ways we can manipulate the edges around node $b$, e.g., create a positive link from $d$ to $b$ or a negative link from $b$ to $d$ as beneficial edges to improve social status for $b$. The examples of deleting harmful edges for promoting $b$'s status include removing the negative edge from $a$ to $b$ or the positive edge from $b$ to $f$.

As we discussed earlier, our goal is to maximize status changes through imperceptible perturbations to the network structure. The change of status includes the promotion and the demotion of status. More specifically, deleting negative influence edges and adding positive influence edges can improve the status of the target node, while deleting positive influence links and adding negative influence links can lower the status of the target node. Here we take the promotion of status as an example, and the demotion of status can be achieved in a similar way.

*1) Operable Nodes Selection:* Given the target node, to improve its status, the operable nodes include its out-degree friends and its in-degree foes for deleting negative influence edges, as well as its 2-hop neighbors for adding positive influence links.

*a) Toy example:* We use the graph before change in Fig. 4 as the original graph, and node $b$ as the target node. To improve the social status of $b$, we first need to determine the set of operable nodes $V_{op}(b)$. For negative influence edge deletion, the operable nodes include $a, c, e$ and $f$. For positive influence link addition, the only 2-hop neighbor $d$ should be added to the set of operable nodes. Thus, we have $V_{op}(b) = \{a, c, d, e, f\}$.

*2) Influence Based Node Ranking:* Due to the diverse relationship with the target node, each operable node has different

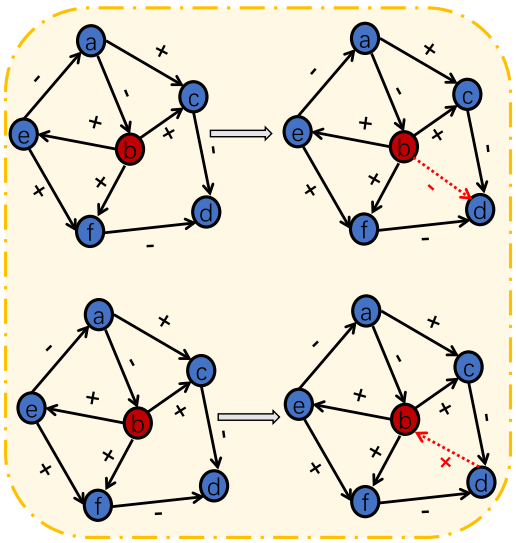



Fig. 5. Addition of positive influence edge. Adding positive influence edge can also improve target node's status. The negative edge from node $b$ to node $d$ and the positive edge from node $d$ to node $b$ are both positive influence nodes for node $b$ in this figure.

influence on the status of the target node. Based on the status evaluation model, we compute the impact of each operable node on status of the target node in terms of status change of the target node. Given target node $u$, the status change brought by an operable node can be measured as

$$q_v(u) = R(F_u(G', A'(v))) - R(F_u(G, A)), v \in V_{op}(u), \tag{25}$$

where $A'(v)$ denotes the weight matrix obtained by modifying the link related to $v$. $F_u(G, A)$ and $R(F_u(G, A))$ are the status score and the status score-based ranking order of $u$ in graph $G$, respectively. $q_v(u)$ denotes the status change of $u$ caused by $v$.

Furthermore, we rank all nodes in the set of operable nodes in a descending order based on the status change of the target node to obtain a sorted set of operable nodes.

*3) Adversarial Graph Generation:* We propose an adversarial attack algorithm as summarized in Algorithm 1. For given target node $u$, the algorithm first obtains the corresponding set of operable nodes, as shown in lines 3-6. The nodes in $V_{op}(u)$ are sorted based on their influence on the target node, as shown in lines 7-10. Then, the algorithm selects the most influential operable node and generates a perturbed graph by deleting or adding an edge. For the newly generated graph, the algorithm judges whether it meets the three constraints Eq.(1)-(3) of the objective function. If not, we undo the operation (line 24 and line 26). These steps are repeated until the budget runs out. In short, the algorithm iteratively adds or deletes edges between target node $u$ and the nodes in $V_{op}(u)$ to maximize social status improvement of $u$. The resulting graph is adversarial graph $G'$.

*4) Time Complexity Analysis:* As mentioned above, the proposed framework comprises of two parts: status evaluation and adversarial attack. We use the localized graph convolutions [21] to compute the status scores, which implies that each status convolution layer conduct node-wise feature aggregation

**Algorithm 1** $A^2S^2$-GNN

**Input:** Original graph $G$, weight matrix $A$, $\tau$, target node $u$, attack budget $\Delta$, status evaluation model $F(\cdot)$ and ranking function $R(\cdot)$.

**Output:** Adversarial graph $G'$.

1: $V_{op}(u) = \emptyset$.
2: $G' = G$.
3: **for** every positive influence neighbour $v$ of $u$ **do**
4: $\quad V_{op}(u) = V_{op}(u) \bigcup \{v\}$.
5: **for** every 2-hop neighbour $v$ of $u$ **do**
6: $\quad V_{op}(u) = V_{op}(u) \bigcup \{v\}$.
7: **for** $\forall v \in V_{op}(u)$ **do**
8: $\quad q_v(u) = R(F_u(G', A'(v))) - R(F_u(G, A))$.
9: $\quad$ rank every node $v$ in $V_{op}(u)$ in a descending order in terms of $q_v(u)$.
10: $\quad$ obtain a sorted $V_{op}(u)$.
11: **while** $\Delta > 0$ and $V_{op}(u) \neq \emptyset$ **do**
12: $\quad G'' = G'$.
13: $\quad v$ = the highest ranked node in $V_{op}(u)$.
14: $\quad$ Remove $v$ from $V_{op}(u)$.
15: $\quad$ **if** a link exists between $u$ and $v$ **then**
16: $\quad\quad$ delete the link.
17: $\quad\quad$ then generate $G'$ and $A'$.
18: $\quad$ **else**
19: $\quad\quad$ create a link between $u$ and $v$.
20: $\quad\quad$ then generate $G'$ and $A'$.
21: $\quad$ **if** $\Lambda(G, G') < \tau$ **then**
22: $\quad\quad$ **if** Eq. (1) satisfies **then**
23: $\quad\quad$ **else**
24: $\quad\quad\quad G' \leftarrow G''$.
25: $\quad$ **else**
26: $\quad\quad G' \leftarrow G''$.
27: $\quad \Delta = \Delta - 1$.
**return** $G'$.

only from the immediate neighbors. All nodes share the parameters of the status convolution layer, the complexity of status evaluation is thus determined by the complexity of model parameters, independent of the size of the graph. Therefore, the status evaluation cost for each node is $O(J^M Q^2)$, where $J$ represents the average number of neighbors, $M$ is the optimal number of status convolution layers, and $Q$ denotes the hidden features of nodes. The key calculation of adversarial attacks include the status score-based ranking and the attack budget-constrained attack (i.e., the edge adding or deleting operation). Since the adversarial attack only considers the immediate neighbors and 2-hop neighbors, the number of nodes in the operable node set is $(J + J^2)$. specifically, the cost of our proposed framework is $[(J+J^2)*O(J^M Q^2)+\Delta]$, i.e., $O(J^{M+2}Q^2)$.

## V. Experimental Evaluation

In this section, we conduct experiments and compare the performance of the proposed attack model and baselines on the benchmark datasets.

TABLE II
STATISTICS OF TYPE I DATASETS

| Dataset | # of nodes | # of edges | +edges(%) | -edges(%) |
|---|---|---|---|---|
| EPINIONS | 131,828 | 841,372 | 85.30 | 14.70 |
| SLASHDOT | 82,140 | 549,202 | 77.40 | 22.60 |
| REDDIT | 55,863 | 858,490 | 90.42 | 9.58 |
| WIKIRFA | 11,258 | 179,418 | 77.92 | 22.08 |
| WIKIELEC | 7,126 | 104,167 | 78.78 | 21.22 |
| BITCOIN-OTC | 5,881 | 35,592 | 89.90 | 10.01 |
| BITCOIN-ALPHA | 3,783 | 24,186 | 93.65 | 6.35 |

TABLE III
STATISTICS OF TYPE II DATASETS

| Dataset | # of nodes | # of edges | Density |
|---|---|---|---|
| DBLP | 35,135 | 941,936 | 0.076 |
| S-Epinions | 18,089 | 355,217 | 0.109 |
| Ciao | 2,342 | 57,544 | 1.049 |

### A. Datasets

We perform experiments on two types of datasets, i.e., signed directed social network datasets and unsigned undirected social network datasets. The former is used to validate the effectiveness and efficiency of the proposed attack algorithm, and the latter is utilized to verify the transferability of our attack algorithm. Type I datasets include Epinions, Slashdot [24], WikiRfa [25], WikiElect [26], Reddit, Bitcoin-OTC and Bitcoin-Alpha, and the statistics of datasets are shown in Table II. Type II datasets are composed of DBLP [27], Simplified Epinions (S-Epinions) [28] and Ciao [29], and the detail of datasets are shown in Table III.

**Epinions**. Epinions is a network of who-trust-whom relationships between users, where trust is represented by positive links and distrust is represented by negative links.

**Slashdot**. Slashdot is a technology-related news website with a specific community. Users can tag other users as "friends" or "foe", which forms a signed directed network.

**WikiRfa**. WikiRfa consists of the voting information of Wikipeida administrator candidates. Any user can vote as supporting, neutral, or opposing for a Wikipedia editor.

**WikiElec**. WikiElec is the voting data related to Wikipedia administrator, and its definition is similar to that of Wikirfa.

**Reddit**. Reddit captures directed connections between two subreddits. A subreddit is a community on Reddit. The network is directed, signed, temporal, and attributed.

**Bitcoin-OTC**. Bitcoin-OTC is a who-trusts-whom network of people who trade using Bitcoin on a platform called Bitcoin OTC. Members of Bitcoin OTC rate other members on a scale of $-10$ (total distrust) to $+10$ (total trust).

**Bitcoin-Alpha**. Bitcoin-Alpha is similar to Bitcoin-OTC, but on another platform called Bitcoin Alpha.

**DBLP**. DBLP is a citation network of academic papers, which is built by focusing on the citation relationships among the authors in the six research fields.

**S-Epinions**. S-Epinions is a simplified version of Epinions. The resulting social network has only positive links.

**Ciao**. Ciao is a review network where users can write comments and rate the comments of other users. When building a trust network, we add a edge between user A and user B if user A trusts B.

### B. Experiments Settings

All the experiments are conducted on a computer with Intel Core i5-10600K 6-core 4.10GHz CPU, GeForce RTX 3070 Ti GPU, 16G RAM, and 1T SSD. The status evaluation model is implemented in PyTorch3. For each dataset, we use 80% edges for training and 20% for test. The training dataset and the test dataset are randomized in each run. There is no feature in the datasets, thus we randomly generate the node feature matrix with 64 dimensions. The learning rate is tuned amongst $\{0.001, 0.005, 0.01, 0.05\}$ according to a grid search, and the coefficient of $l_2$-norm is in $\{10^{-5}, 10^{-4}\}$. We stop training if the training loss does not increase in 10 epochs or we reach the given maximum number of rounds, which is 600 for our experiments. By default, our status evaluation model adopts three status convolution layers [32, 64, 32], and set the learning rate as 0.01 and the regularization coefficient as $10^{-5}$, respectively.

For the attack model, we focus on exploring the impact of several elements on algorithm performance, including the value of budget and status level of the target node. We run each attack algorithm for 10 times on each dataset, and get the average of these 10 results as a final result. The best effect for an attack is that the target node status increases while the neighbors' and global status remain stable. To realize scalability, we adopt different strategies for graphs with different sizes. For large graphs, we only consider deleting edges, and we delete and insert edges for small graphs at the same time.

We adjust the budget according to the degree of the target node (i.e., the number of neighbors of the target node). This is inspired by the observation that nodes with a high degree are more difficult to attack than nodes with a low degree. This means that attacks on high degree nodes need a larger budget. On the contrary, attacks on low degree nodes only need a small budget. We also study the impact of budget size on the performance of the attack.

### C. Metrics

To evaluate the the effectiveness and efficiency of the attacks, we define three quantitative metrics as follows:

$$\Delta_{S[u]} = \frac{\Delta_{status}[u]}{I_{status}[u]},$$
$$\Delta_{N[u]} = \frac{1}{\mid N(u) \mid} \sum_{v \in N(u)} \Delta_{S[v]},$$
$$\Delta_G = \frac{1}{N} \sum_{v \in V} \Delta_{S[v]},$$

where $\Delta_{S[u]}$ represents the change of the relative status of node $u$, $\Delta_{status}[u]$ is the difference between status ranking orders of node $u$ before and after the attack, and $I_{status}[u]$ is

the initial status of node $u$. $\Delta_{N[u]}$ denotes the mean change of node $u$'s neighbors' status. $\Delta_G$ is the mean change of the global status. We hope to improve the status of the target node as much as possible and keep the status of the whole graph as smooth as possible, which means we expect a large $\Delta_{S[u]}$, as well as small $\Delta_G$ and $\Delta_{N[u]}$, where $\Delta_{N[u]}$ is a metric for the status change of nodes around the target node, which is more targeted than $\Delta_G$. Note that a larger $\Delta_{S[u]}$, a smaller $\Delta_G$ and a smaller $\Delta_{N[u]}$ indicate better attack performance.

### D. Performance Analysis of $A^2S^2$-GNN

*1) Baselines:* For effectiveness comparison, since we are the first attempt to adversarial attack on social status in signed directed networks, and there are no available algorithms as benchmarks, therefore, we use the following approaches as our baselines.

- Random Target Attack (RTA). Following the idea of [14], for a given target node, RTA randomly samples a neighbor of the target node in every step. If the neighbor has a negative impact on the target node, then the link is deleted; otherwise, sample again. If the target node has no neighbor, RTA randomly samples a node and creates a link between them. RTA repeats these steps until the budget is exhausted.
- DICE. We adapt the original DICE algorithm in [30] by using a heuristic strategy. Improved DICE first removes some edges that negatively affect the target node's status, and then spends the rest of the budget inserting edges between the target node and the rest of the graph.
- Edge Feature based Attack (EFBA). EFBA modifies the signs (i.e., positive and negative) and directions of the edges that have a negative impact on the target node within a given budget.
- Adapted TDGIA (A-TDGIA). The algorithm, TDGIA, designed in [31] aims to fool the node classification model via finding the existing nodes that are most helpful to attack based on the topology vulnerability and then injecting adversarial nodes around these nodes sequentially. We adapt TDGIA to our social status attack scenario, called as A-TDGIA, and treat the immediate neighbors and 2-hop neighbors of the target node as important nodes.

*2) Attack Effectiveness:* Table IV and Table V list attack results of status of five algorithms on seven datasets with the target node in different status levels, respectively. The attack includes upgrading and degrading the status of the target node, where the target node is selected randomly from the specified status level, and the attack budget is set as the degree of the target nodes, i.e., the number of neighbors of the target node. We group the nodes as high, middle, and low level based on the status ranking according to the ratio of 1:3:6. We put the maximum of $\Delta_{S[u]}$, the minimum of $\Delta_{N[u]}$, and the minimum of $\Delta_G$ in bold to highlight the performance comparison of attack. We have the following observations:

For metric $\Delta_{S[u]}$, we observe that $A^2S^2$-GNN achieves the best performance over seven datasets, which demonstrates $A^2S^2$-GNN's superiority. For metrics $\Delta_{N[u]}$ and $\Delta_G$, $A^2S^2$-GNN can not always attain the minimum status change but close to the minimum value.

The difficulty of attack varies with the social level of the target node. The experimental results in Table IV and Table V verify that improving status is harder for nodes with higher status and reducing status is harder for users with low-level status. When the attack objective is to improve the status of the target node, the status change can be as high as 89.36% to upgrade the target node with low status, however, the minimum status change is only 7.42% to upgrade the target node with high status. When the attack aim is to reduce the status of the target node, while the status change can be as high as 192.38% to degrade the target node with high status, the minimum status change is 19.30% to degrade the target node with low status. Compared with improving status experiments, reducing status experiments also have a greater range of variation under the attacks.

*3) Effect of the Attack Budget:* In this set of experiments, we investigate the impact of budget on the attack performance. Here, we use $\Delta_{S[u]}$ to measure attack performance of improving status, as well as set attack budget as 20%, 40%, 60%, 80% and 100% of the degree of the target node, respectively. Because the target node is randomly selected, it may be at any status level. Naturally, the performance increases along with attack budget, as shown in Fig. 6 (a)-(g). A larger budget means more number of times of deleting or adding links, eventually leading to greater status changes. At the beginning, the performance of the five algorithms had little difference when the attack budget is small. $A^2S^2$-GNN increases sharply with the budget, the performance difference of the five algorithms becomes larger and larger. Among the five algorithms, RTA has the worst performance. Compared with RTA, $A^2S^2$-GNN can achieve up to 2 times improvement. In essence, $A^2S^2$-GNN always chooses the node with the greatest influence every attack, thus the target node can achieve the biggest status change. In Fig. 6 (a)-(g), we only report the performance comparison of improving status on seven datasets, and the performance comparison of reducing status are omitted as they present a similar performance trend.

*4) Effect of the Number of Hops:* Fig. 7 illustrates the impact of attacking neighbors with different hop counts on the status change of the target node, where the target node is selected randomly. The results of our attack algorithm on 7 datasets confirm that the impact of the node on the target node shows a rapid downward trend with the increase in the number of hops. Starting from 3-hop neighbors, the influence is close to 0, which can be ignored. This is why our attack framework only considers neighbors and 2-hop neighbors. Furthermore, the impact on the status of the target node is related to the number of nodes in the network. By attacking neighbors of the target node, the resulting performance on Bitcoin-Alpha with 3,783 nodes is far better than that on Epinions with 131,828 nodes. The influence of neighbors with other hops on the target node status has a similar trend. Therefore, we can conclude that it is more difficult to attack a larger scale network.

*5) Attack Cost:* The running time of five algorithms on seven datasets is shown in Fig. 8. Consistent with intuition,

TABLE IV
ATTACK PERFORMANCE COMPARISON OF IMPROVING SOCIAL STATUS

| Dataset | | low-status | | | middle-status | | | high-status | | | random | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $\Delta_{S[u]}\uparrow$ | $\Delta_{N[u]}\downarrow$ | $\Delta_G\downarrow$ | $\Delta_{S[u]}\uparrow$ | $\Delta_{N[u]}\downarrow$ | $\Delta_G\downarrow$ | $\Delta_{S[u]}\uparrow$ | $\Delta_{N[u]}\downarrow$ | $\Delta_G\downarrow$ | $\Delta_{S[u]}\uparrow$ | $\Delta_{N[u]}\downarrow$ | $\Delta_G\downarrow$ |
| WIKIELEC | RTA | 23.81% | 7.53% | 4.42% | 11.23% | **2.07%** | **1.82%** | 6.80% | 4.45% | 3.72% | 7.39% | 4.04% | **3.19%** |
| | DICE | 48.77% | 6.38% | 4.19% | 31.92% | 4.53% | 4.05% | 15.04% | 4.26% | 4.21% | 33.19% | 5.27% | 5.73% |
| | EFBA | 32.06% | 4.76% | **3.62%** | 22.27% | 9.66% | 4.45% | 10.66% | 8.93% | 3.37% | 26.24% | 5.53% | 3.56% |
| | A-TDGIA | 33.54% | 9.50% | 3.79% | 31.40% | 5.68% | 1.97% | 15.45% | 9.41% | 4.38% | 32.33% | 4.47% | 6.72% |
| | $A^2S^2$-Attack | **55.13%** | **2.85%** | 3.81% | **40.96%** | 2.75% | 2.02% | **23.83%** | **2.14%** | **2.80%** | **43.76%** | **2.51%** | 3.52% |
| WIKIRFA | RTA | 33.09% | **4.76%** | 7.63% | 20.37% | 9.52% | 7.83% | 10.17% | 6.71% | 5.95% | 24.95% | **3.74%** | **4.13%** |
| | DICE | 66.94% | 7.40% | 7.09% | 50.16% | 9.16% | 8.03% | 13.27% | 3.82% | 6.37% | 61.15% | 7.22% | 6.31% |
| | EFBA | 57.13% | 6.96% | 4.51% | 38.68% | 6.40% | 7.51% | 10.81% | 8.76% | 9.86% | 2.20% | 8.02% | 8.68% |
| | A-TDGIA | 59.73% | 7.64% | 4.33% | 41.56% | **6.05%** | 8.07% | 18.24% | **2.78%** | **5.80%** | 51.60% | 6.04% | 5.03% |
| | $A^2S^2$-Attack | **85.16%** | 5.16% | **4.28%** | **63.55%** | 6.96% | **7.03%** | **55.11%** | 8.07% | 8.10% | **73.18%** | 4.99% | 6.76% |
| SLASHDOT | RTA | 28.79% | 8.34% | 6.03% | 22.89% | 4.12% | 5.77% | 6.28% | **2.09%** | 5.61% | 20.36% | 6.02% | 5.61% |
| | DICE | 58.21% | 8.95% | 7.57% | 15.32% | 4.39% | 5.63% | 13.66% | 5.53% | 5.33% | 21.45% | **3.70%** | 4.71% |
| | EFBA | 57.33% | 9.46% | **5.47%** | 24.94% | 4.77% | 6.67% | 14.66% | 3.98% | **5.20%** | 37.29% | 8.55% | 4.73% |
| | A-TDGIA | 62.35% | **7.29%** | 5.01% | 23.55% | 3.05% | 6.13% | 16.18% | 2.52% | 6.01% | 30.90% | 4.07% | 6.12% |
| | $A^2S^2$-Attack | **77.05%** | 7.73% | 5.66% | **40.90%** | **1.95%** | **2.83%** | **32.87%** | 5.78% | 5.67% | **41.80%** | 4.91% | **4.31%** |
| EPINIONS | RTA | 22.03% | **4.41%** | **2.25%** | 7.09% | **3.14%** | 7.41% | 6.95% | 6.67% | 9.88% | 16.64% | 2.81% | 2.45% |
| | DICE | 30.23% | 8.01% | 3.06% | 17.97% | 3.23% | 8.42% | 7.40% | **3.59%** | 4.16% | 23.72% | **1.14%** | **2.11%** |
| | EFBA | 40.86% | 9.35% | 2.34% | 13.75% | 8.73% | 3.49% | 8.58% | 6.39% | 5.47% | 9.62% | 2.25% | 2.28% |
| | A-TDGIA | 34.90% | 5.40% | 3.00% | 20.75% | 4.56% | 6.19% | 12.82% | 7.58% | 5.16% | 32.56% | 4.52% | 6.72% |
| | $A^2S^2$-Attack | **69.48%** | 5.18% | 2.35% | **37.41%** | 3.64% | **2.07%** | **22.53%** | 8.27% | **2.03%** | **68.96%** | 4.71% | 2.42% |
| BITCOIN-ALPHA | RTA | 34.21% | 9.26% | 5.86% | 13.52% | **4.64%** | **2.53%** | 7.88% | 6.15% | **3.93%** | 26.48% | 5.31% | 4.06% |
| | DICE | 63.11% | 6.92% | 4.45% | 55.04% | 5.36% | 4.36% | 12.97% | 3.47% | 4.33% | 57.05% | 5.54% | 3.93% |
| | EFBA | 24.00% | **5.24%** | **3.67%** | 19.12% | 9.21% | 4.03% | 13.21% | 8.36% | 4.36% | 21.59% | 8.65% | **3.56%** |
| | A-TDGIA | 50.71% | 7.35% | 6.72% | 32.63% | 4.86% | 7.25% | 15.64% | 6.39% | 7.13% | 32.68% | 6.20% | 7.25% |
| | $A^2S^2$-Attack | **89.36%** | 9.35% | 4.26% | **57.27%** | 7.07% | 7.05% | **15.64%** | **2.10%** | 4.23% | **84.31%** | **4.29%** | 4.30% |
| BITCOIN-OTC | RTA | 6.52% | 5.32% | 6.01% | 4.61% | 3.69% | 9.85% | 2.98% | 5.20% | 2.22% | 6.42% | **3.50%** | 9.31% |
| | DICE | 15.97% | 6.81% | 5.79% | 9.23% | **2.37%** | 2.75% | 4.53% | 8.16% | 8.63% | 11.81% | 8.69% | 6.84% |
| | EFBA | 17.47% | 7.37% | **1.63%** | 9.37% | 2.51% | 4.98% | 4.97% | 9.55% | **2.12%** | 7.49% | 4.26% | 4.25% |
| | A-TDGIA | 24.59% | 5.36% | 3.11% | 16.25% | 8.89% | 7.51% | 15.45% | **3.36%** | 4.12% | 22.33% | 6.45% | 3.12% |
| | $A^2S^2$-Attack | **55.43%** | **1.35%** | 1.85% | **20.21%** | 4.16% | **1.35%** | **7.42%** | 6.86% | 8.75% | **31.58%** | 5.72% | **1.32%** |
| REDDIT | RTA | 38.65% | 8.53% | 8.95% | 19.29% | 9.22% | 6.21% | 7.33% | 5.70% | **1.47%** | 20.50% | 6.14% | **3.45%** |
| | DICE | 68.96% | 6.72% | 4.25% | 46.09% | 9.70% | **4.10%** | 34.34% | 6.92% | 3.70% | 51.68% | **4.38%** | 9.47% |
| | EFBA | 49.55% | **3.40%** | 9.32% | 40.64% | 8.48% | 5.32% | 19.17% | 3.62% | 5.73% | 42.71% | 7.13% | 3.69% |
| | A-TDGIA | 44.59% | 5.36% | 4.51% | 36.25% | 8.89% | 5.15% | 28.68% | **3.36%** | 4.12% | 37.83% | 4.40% | 6.12% |
| | $A^2S^2$-Attack | **81.29%** | 8.12% | **1.59%** | **61.27%** | **4.89%** | **4.10%** | **56.06%** | 6.74% | 3.19% | **65.43%** | 7.53% | 5.15% |

the running time of the five algorithms increases linearly with the scale of the dataset. Since Epinions owns more than hundreds of thousands of nodes and Bitcoin-Alpha has only a few thousand nodes at the same time, while the attack on Epinions corresponds to the longest running time, the attack on Bitcoin-Alpha corresponds to the shortest running time. Furthermore, RTA has the worst performance on every dataset, while DICE shows the best performance for every dataset. As we expected, $A^2S^2$-GNN and DICE exhibit similar properties in running time. The difference between these two algorithms mainly results from influence calculation and influence-based ranking, and is negligible. Taking the results in Fig. 6 and Fig. 8 into consideration, it is obvious that the performance improvement is at the cost of running time, More specifically, $A^2S^2$-GNN achieves better attack performance and takes slightly longer running time than DICE. Therefore, we have to tradeoff between attack efficiency and the running time.

### *E. Transferability of the Proposed Attack Algorithm*

*1) Baselines:* To corroborate transferability of the proposed algorithm, we implement our adversarial attack on MPR [32] and its benchmark algorithms, i.e., IND, BET [33], CLO [34], BPR and WPR, which are diverse social status ranking algorithms. We use the released source code of [32], and all the parameter settings follow [32].

- Incoming Degree-based Ranking (IND). IND believes that nodes with larger incoming degree have higher influence.
- BETweenness-based Ranking (BET). BET thinks that nodes with bigger betweenness scores have higher social status.
- CLOseness-based Ranking (CLO). CLO deems that nodes with bigger closeness scores have greater importance.
- Binary PageRank (BPR). BPR implements PageRank on a binary network, and sets the weights of all edges to 1.

TABLE V
ATTACK PERFORMANCE COMPARISON OF REDUCING SOCIAL STATUS

| Dataset | | low-status | | | middle-status | | | high-status | | | random | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $\Delta_{S[u]}\uparrow$ | $\Delta_{N[u]}\downarrow$ | $\Delta_G\downarrow$ | $\Delta_{S[u]}\uparrow$ | $\Delta_{N[u]}\downarrow$ | $\Delta_G\downarrow$ | $\Delta_{S[u]}\uparrow$ | $\Delta_{N[u]}\downarrow$ | $\Delta_G\downarrow$ | $\Delta_{S[u]}\uparrow$ | $\Delta_{N[u]}\downarrow$ | $\Delta_G\downarrow$ |
| WIKIELEC | RTA | 3.42% | 9.09% | **3.49%** | 18.14% | 8.66% | 4.76% | 22.09% | 7.69% | 4.53% | 19.16% | 4.03% | 3.30% |
| | DICE | 30.62% | 6.65% | 4.56% | 55.37% | **4.34%** | 5.00% | 72.6% | 6.01% | 6.14% | 48.23% | 3.78% | 3.31% |
| | EFBA | 6.97% | 4.63% | 4.07% | 20.74% | 9.02% | 4.01% | 44.48% | 7.38% | 3.69% | 25.17% | **3.34%** | **3.18%** |
| | A-TDGIA | 22.14% | 4.68% | 4.07% | 39.03% | 4.66% | 4.41% | 69.07% | 4.74% | 3.39% | 43.88% | 3.57% | 4.77% |
| | $A^2S^2$-Attack | **31.52%** | **4.31%** | 4.39% | **55.82%** | 4.52% | **3.55%** | **92.69%** | **4.06%** | **3.38%** | **55.33%** | 3.35% | 3.92% |
| WIKIRFA | RTA | 3.51% | 5.10% | **4.05%** | 34.32% | 9.81% | **3.84%** | 43.02% | 7.76% | 4.69% | 33.09% | 8.07% | 4.96% |
| | DICE | 16.05% | 8.82% | 6.55% | 44.72% | 5.91% | 4.53% | 75.15% | 5.24% | 4.07% | 30.20% | 4.03% | 4.95% |
| | EFBA | 18.19% | 4.23% | 4.71% | 27.34% | 8.89% | 5.09% | 47.04% | 8.08% | 7.19% | 24.75% | 2.59% | 4.17% |
| | A-TDGIA | 19.30% | **4.21%** | 6.58% | 43.78% | 6.67% | 4.47% | 50.41% | **2.32%** | 5.29% | 43.27% | 2.99% | 4.01% |
| | $A^2S^2$-Attack | **19.30%** | 5.02% | 4.45% | **52.35%** | **5.48%** | 3.97% | **99.27%** | 2.45% | **4.05%** | **49.61%** | **2.25%** | **3.90%** |
| SLASHDOT | RTA | 6.70% | 7.42% | **5.55%** | 8.41% | 9.58% | 5.77% | 10.34% | 7.76% | 5.61% | 7.58% | 6.74% | **4.21%** |
| | DICE | 9.44% | 7.68% | 5.69% | 34.57% | **3.59%** | 6.08% | 75.38% | 8.23% | 5.56% | 33.51% | **5.42%** | 6.18% |
| | EFBA | 9.80% | 9.71% | **5.55%** | 33.89% | 7.63% | 5.49% | 55.84% | 8.33% | 5.82% | 24.88% | 9.30% | 5.66% |
| | A-TDGIA | 23.49% | 7.01% | 6.02% | 45.28% | 6.81% | 7.01% | 64.38% | **4.92%** | 4.95% | 42.01% | 5.59% | 6.47% |
| | $A^2S^2$-Attack | **24.95%** | **6.29%** | 5.74% | **64.07%** | 5.04% | **5.41%** | **126.93%** | 7.27% | **4.81%** | **91.75%** | 8.63% | 5.76% |
| EPINIONS | RTA | 2.89% | 7.30% | 5.80% | 14.06% | 9.11% | 6.02% | 18.83% | 5.02% | 2.53% | 13.89% | 3.78% | **4.39%** |
| | DICE | 17.41% | 9.30% | 5.72% | 18.74% | **5.31%** | 9.51% | 22.40% | 8.09% | 6.56% | 13.50% | **1.62%** | 5.71% |
| | EFBA | 19.65% | 9.48% | 7.13% | 40.43% | 8.48% | 7.60% | 59.86% | 5.07% | 6.42% | 36.78% | 2.83% | 5.38% |
| | A-TDGIA | 21.83% | 4.12% | 6.01% | 36.45% | 6.85% | 5.01% | 66.09% | 8.85% | 4.01% | 36.09% | 2.01% | 7.01% |
| | $A^2S^2$-Attack | **32.21%** | **2.83%** | **2.76%** | **52.04%** | 6.22% | **2.38%** | **163.24%** | **2.25%** | **2.30%** | **74.87%** | 2.65% | 4.68% |
| BITCOIN-ALPHA | RTA | 12.96% | 6.18% | 8.02% | 15.71% | 6.37% | 7.59% | 17.64% | 8.33% | 5.12% | 13.52% | **8.40%** | 5.50% |
| | DICE | 19.32% | 9.34% | 5.36% | 26.33% | 7.84% | 5.32% | 59.60% | **6.83%** | 9.22% | 20.33% | 9.05% | 7.87% |
| | EFBA | 22.15% | 8.97% | **5.33%** | 26.17% | 8.47% | **3.58%** | 49.96% | 9.50% | 5.44% | 36.91% | 9.61% | 5.38% |
| | A-TDGIA | 16.80% | 7.86% | 6.74% | 26.25% | 5.86% | 6.47% | 48.86% | 6.88% | **3.45%** | 34.54% | 8.48% | 7.75% |
| | $A^2S^2$-Attack | **27.36%** | **4.32%** | 8.73% | **30.56%** | **5.04%** | 4.02% | **99.38%** | 8.03% | 5.23% | **95.85%** | 9.13% | **3.16%** |
| BITCOIN-OTC | RTA | 14.05% | 9.51% | 5.35% | 20.92% | 6.33% | 9.93% | 38.30% | 9.71% | **4.21%** | 19.05% | 9.04% | 8.21% |
| | DICE | 37.70% | 8.96% | 3.21% | 71.55% | 9.78% | **5.53%** | 81.89% | **5.58%** | 8.62% | 70.34% | 9.18% | 8.75% |
| | EFBA | 27.33% | 7.15% | 2.91% | 29.39% | 3.63% | 8.02% | 48.21% | 7.24% | 7.88% | 72.85% | 6.28% | 8.84% |
| | A-TDGIA | 38.91% | 8.74% | 4.20% | 58.93% | 6.67% | 7.33% | 84.22% | 6.08% | 7.19% | 65.35% | **6.16%** | 8.23% |
| | $A^2S^2$-Attack | **53.54%** | **5.28%** | **2.72%** | **96.39%** | **3.41%** | 6.37% | **122.74%** | 7.67% | 6.21% | **157.24%** | 7.55% | **6.13%** |
| REDDIT | RTA | 9.48% | **8.57%** | 9.19% | 13.21% | 6.56% | 7.97% | 19.94% | 8.36% | **5.86%** | 13.84% | 9.97% | 7.87% |
| | DICE | 39.70% | 9.30% | 9.37% | 60.18% | 9.37% | 7.77% | 98.62% | 9.40% | 7.99% | 60.64% | 9.66% | 7.10% |
| | EFBA | 28.63% | 9.06% | **6.02%** | 29.70% | **6.52%** | 7.86% | 78.27% | 9.20% | 7.15% | 60.21% | 9.03% | 7.60% |
| | A-TDGIA | 42.39% | **5.69%** | 8.02% | 60.50% | 6.61% | **5.01%** | 98.15% | **5.03%** | 9.01% | 69.26% | 9.40% | 8.01% |
| | $A^2S^2$-Attack | **53.09%** | 9.55% | 7.63% | **74.02%** | 7.01% | 6.10% | **192.38%** | 8.76% | 7.49% | **131.76%** | **8.50%** | **5.80%** |

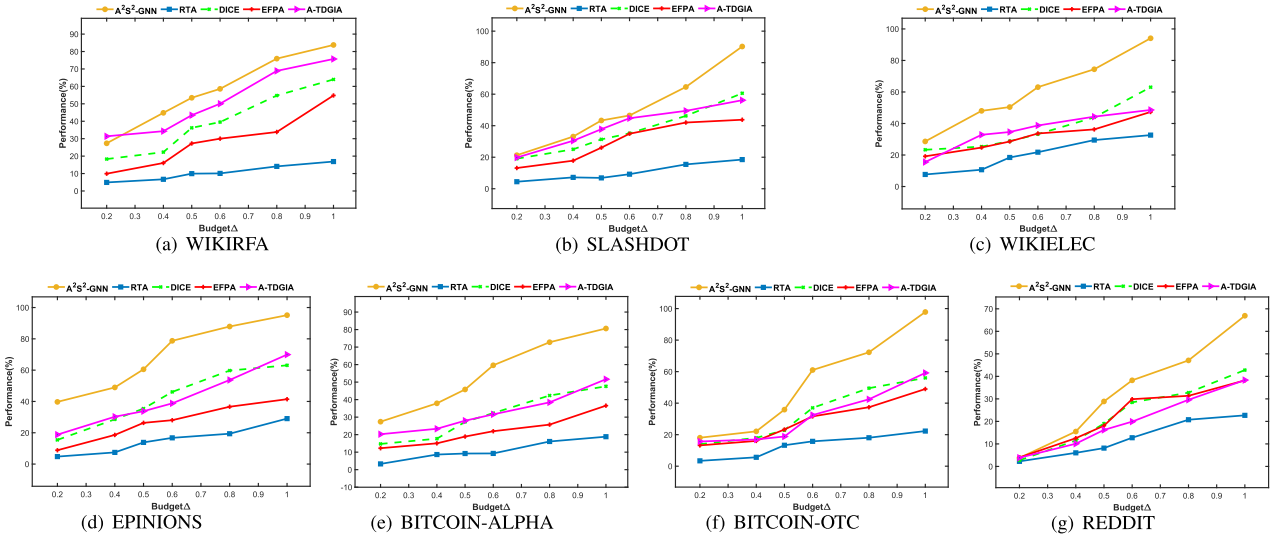
- Weighted PageRank (WPR). WPR carries out PageRank on a weighted network, where the weight of an link rates the interaction frequency between two nodes.
- Motif-based PageRank (MPR). Combining the motif-based relations and edge-based relations, MPR use higher-order relations to optimize the ranking calculation.

*2) Transferability Analysis:* The attack includes upgrading and degrading the status of the target node. Table VI and Table VII illustrate attack results of six social status evaluation models on three datasets with the target node in different status levels, respectively. We have the following observations:

To improve the status of the target node, it can be seen from the results of six algorithms on three datasets that the status change of the target node with low-level status is always the largest, and all algorithms can achieve acceptable performance in terms of $\Delta_{N[u]}$ and $\Delta_G$. However, the status of the target node with high status is improved by only 14.58%, 12.06% and 10.15% on DBLP, S-Epinions and Ciao, respectively. For metric $\Delta_{S[u]}$, compared with the other five algorithms, IND always achieves the largest status improvement on the three datasets.

To reduce the status of the target node, we can notice that the status change of the target node with high-level status is always the largest. Consistent with our attack expectations, all algorithms can achieve acceptable performance in terms of $\Delta_{N[u]}$ and $\Delta_G$. However, lowering the status of the target node with low status is surprisingly difficult. While the status change of the target node with low-status by MPR on Ciao is only 7.66%, the maximum status change of the target node with high-status by IND on S-Epinion is 98.91%.

Since the six ranking algorithms are based on different key properties, the impact of edge addition or deletion caused by the attack algorithm on the target nodes under these algorithms is also different. Nevertheless, the status of target nodes at




(a) WIKIRFA (b) SLASHDOT (c) WIKIELEC

(d) EPINIONS (e) BITCOIN-ALPHA (f) BITCOIN-OTC (g) REDDIT

Fig. 6. Performance of attack algorithm with different budgets on four datasets.

TABLE VI
TRANSFERABILITY COMPARISON OF THE ATTACK ALGORITHM IN IMPROVING SOCIAL STATUS

| Dataset | | DBLP | | | S-Epinion | | | Ciao | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | $\Delta_{S[u]} \uparrow$ | $\Delta_{N[u]} \downarrow$ | $\Delta_G \downarrow$ | $\Delta_{S[u]} \uparrow$ | $\Delta_{N[u]} \downarrow$ | $\Delta_G \downarrow$ | $\Delta_{S[u]} \uparrow$ | $\Delta_{N[u]} \downarrow$ | $\Delta_G \downarrow$ |
| IND | high-status | 33.55% | 6.88% | **1.74%** | 18.16% | **4.01%** | 3.39% | 39.12% | 5.71% | **1.22%** |
| | middle-status | 52.47% | **5.09%** | 3.93% | 38.84% | 5.73% | **2.88%** | 48.63% | 9.21% | 2.40% |
| | low-status | **76.31%** | 7.52% | 2.47% | **62.81%** | 5.89% | 3.67% | **71.22%** | **3.17%** | 1.48% |
| BET | high-status | 17.77% | 7.51% | 2.23% | 12.06% | 6.85% | **1.08%** | 15.16% | 5.43% | 7.42% |
| | middle-status | 52.73% | **4.57%** | **1.08%** | 37.36% | 4.99% | 2.43% | 34.96% | 5.38% | **2.10%** |
| | low-status | **70.46%** | 6.35% | 1.53% | **58.55%** | **4.22%** | 3.27% | **53.75%** | **2.34%** | 4.67% |
| CLO | high-status | 14.58% | **1.85%** | **3.80%** | 15.93% | 5.92% | **3.80%** | 29.43% | 4.85% | 6.67% |
| | middle-status | 25.12% | 2.51% | 5.43% | 34.39% | 3.59% | 5.43% | 39.40% | 8.56% | **5.42%** |
| | low-status | **68.06%** | 5.11% | 5.42% | **42.52%** | **1.46%** | 5.42% | **42.31%** | **4.29%** | 5.43% |
| BPR | high-status | 28.33% | 5.73% | 5.34% | 22.19% | 4.66% | 6.39% | 10.15% | **5.44%** | **5.07%** |
| | middle-status | 53.43% | **5.11%** | 7.29% | 41.89% | 4.20% | **5.34%** | 44.05% | 6.48% | 5.34% |
| | low-status | **74.29%** | 8.63% | **4.86%** | **50.84%** | **4.19%** | **5.34%** | **59.58%** | 6.39% | 6.25% |
| WPR | high-status | 22.63% | 5.85% | **4.23%** | 18.89% | 5.38% | 6.67% | 15.36% | 6.58% | 5.35% |
| | middle-status | 35.98% | **4.32%** | 5.35% | 31.43% | **5.08%** | 6.36% | 24.60% | 6.40% | **3.96%** |
| | low-status | **52.48%** | 4.73% | 5.07% | **61.08%** | 8.41% | **4.19%** | **30.78%** | **6.33%** | 5.34% |
| MPR | high-status | 34.88% | **4.54%** | 5.49% | 25.23% | 4.83% | 5.34% | 21.36% | **3.07%** | 5.36% |
| | middle-status | 41.36% | 6.86% | 5.33% | 47.32% | **4.81%** | **4.48%** | 32.12% | 5.91% | **5.31%** |
| | low-status | **63.53%** | 8.26% | **4.87%** | **51.32%** | 5.69% | 5.34% | **47.99%** | 7.36% | 5.62% |

different status levels is improved or reduced by adversarial attack in a similar trend to $A^2S^2$-GNN. The results prove the transferability of our attack method on other ranking models.

### *F. Insight*

Based on experimental confirmation and performance analysis, we realize that the farther the distance between nodes, the smaller the influence, and the larger the network scale, the more difficult the attack is. More importantly, the top status level users own almost no negative influence neighbors. Furthermore, it is difficult to upgrade their social status and easy to degrade their status. The lower-ranked nodes possess lots of negative influence edges, and their status is difficult to reduce, but it is easy to improve their status. The insight is in

TABLE VII
TRANSFERABILITY COMPARISON OF THE ATTACK ALGORITHM IN LOWERING SOCIAL STATUS

| Dataset | | DBLP | | | S-Epinion | | | Ciao | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | $\Delta_{S[u]} \uparrow$ | $\Delta_{N[u]} \downarrow$ | $\Delta_G \downarrow$ | $\Delta_{S[u]} \uparrow$ | $\Delta_{N[u]} \downarrow$ | $\Delta_G \downarrow$ | $\Delta_{S[u]} \uparrow$ | $\Delta_{N[u]} \downarrow$ | $\Delta_G \downarrow$ |
| IND | high-status | **89.10%** | 5.82% | **5.26%** | **98.91%** | 5.47% | 6.79% | **95.66%** | 4.32% | 5.65% |
| | middle-status | 60.83% | **4.26%** | 8.67% | 58.25% | **4.29%** | **4.33%** | 65.93% | 5.23% | 5.83% |
| | low-status | 12.36% | 6.51% | 6.33% | 18.03% | 5.41% | 6.23% | 17.80% | **2.03%** | **4.53%** |
| BET | high-status | **90.91%** | 4.32% | **5.94%** | **90.04%** | 8.56% | 7.23% | **97.11%** | **4.89%** | **5.64%** |
| | middle-status | 72.56% | **4.31%** | 7.44% | 56.83% | 7.44% | 6.05% | 78.40% | 7.44% | 7.88% |
| | low-status | 29.56% | 5.48% | 7.29% | 14.92% | **5.11%** | **5.66%** | 28.93% | 5.51% | 6.56% |
| CLO | high-status | **82.74%** | 6.13% | 6.15% | **76.63%** | 7.90% | 5.73% | **96.73%** | 7.57% | **5.42%** |
| | middle-status | 45.85% | 5.99% | **5.42%** | 59.67% | **4.56%** | 5.43% | 54.48% | 6.24% | 5.43% |
| | low-status | 27.23% | **5.19%** | 5.43% | 26.65% | 5.13% | **5.42%** | 14.23% | **5.62%** | 6.16% |
| BPR | high-status | **98.11%** | **5.15%** | **4.98%** | **48.87%** | 5.93% | **6.55%** | **81.84%** | 7.24% | 4.73% |
| | middle-status | 62.66% | 5.74% | 7.55% | 38.06% | 6.65% | 6.58% | 61.73% | **5.79%** | 7.69% |
| | low-status | 19.51% | 6.95% | 5.20% | 11.37% | **4.71%** | 6.57% | 33.35% | 6.67% | **3.05%** |
| WPR | high-status | **96.52%** | 6.25% | 7.24% | **61.60%** | 5.07% | 6.16% | **73.88%** | 9.55% | **5.89%** |
| | middle-status | 62.81% | **5.67%** | 7.31% | 27.12% | **4.28%** | 5.19% | 60.46% | **5.16%** | 6.29% |
| | low-status | 21.96% | 9.70% | **6.67%** | 12.21% | 6.90% | **4.82%** | 24.13% | 7.52% | 5.91% |
| MPR | high-status | **68.20%** | 3.93% | 4.14% | **42.88%** | 7.15% | 7.52% | **60.61%** | 2.84% | 4.58% |
| | middle-status | 44.69% | 5.93% | 3.20% | 38.80% | **5.65%** | 7.23% | 22.30% | **1.32%** | **2.71%** |
| | low-status | 17.63% | **2.63%** | **2.16%** | 14.27% | 7.81% | **5.42%** | 7.66% | 5.09% | 3.67% |

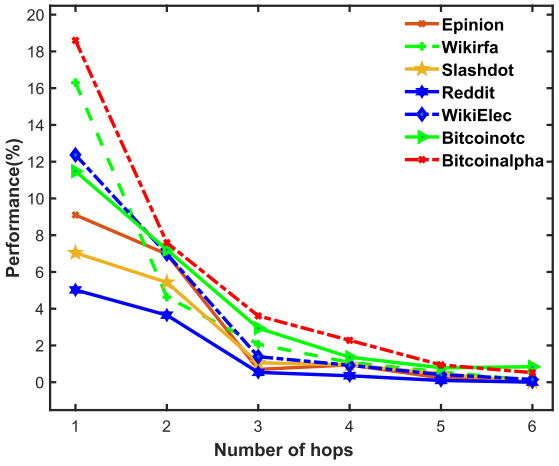



Fig. 7. Performance of attack algorithm with different number of hops.

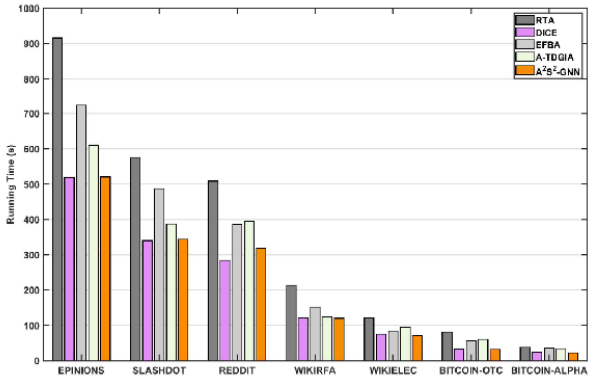



Fig. 8. Running time of different attack algorithms.

line with our general knowledge: the upward passage must be difficult, and the downward door will always open.

## VI. RELATED WORK

### A. Status Evaluation

Status theory helps understand the role of social status in various applications. In [1], Giddens et al. analyzed the influencing factors of social status from the perspective of sociology. Leskovec et al. [24] proposed the status theory as a substitute for the balance theory to predict the sign of edges and explore the underlying social mechanisms. Leskovec et al. [35] evaluated the relationship between the status of online users and individual preferences. Taking the motif-based relations and edge-based relations into consideration, MPR [32] used higher-order relations to optimize the ranking calculation. Deep neural networks (DNNs) are useful tools for graph analysis. Li et al. [8] presented an effective prediction method

for the future size of cascades based on end-to-end deep learning. Wang et al. [6] evaluated the role of status in building trust relationships and proposed a trust prediction frame based on status theory. Inspired by [6], starting with positive and negative effects from neighboring nodes, we quantify the status score and then propose an adversarial attack framework based on the status score to explore the status change of the target node.

### B. Adversarial Attacks

Dai et al. [36] surveyed GNNs from the aspects of privacy and robustness, and pointed out adversarial attacks with imperceptible perturbations can fool the detection model to achieve the attack objective through almost unnoticeable changes. Considerable research on adversarial attacks has been carried out recently, Carlini et al. [37] achieved the attack objective with imperceptible perturbation based on $L_p$ norm distance in image domain. As described in [38] and [39], adversarial attack on graph data can be divided into four categories: node-oriented, link-oriented, graph-oriented, and community-oriented. In [14], the adversarial attack on node classification was performed by modifying the node characteristics and graph structure. Adversarial attacks on node embedding have been examined [11], [12]. Wang et al. [15] formulated graph-based classification as an optimization problem and then achieved the attack goal by modifying the graph structure. Lin et al. [40] proposed an incremental computation-based adversarial attack algorithm to explore the possibility of adversarial attack against link prediction models. Zhang et al. [16] focused on the vulnerability of GNNs, and presented an attack model, LafAK, as well as a defense framework. Li et al. [17] proposed an adversarial attack solution on community detection to hide the target user. Zhan et al. [41] proposed a defense strategy with the validation set against Mettack, and then designed a black-box Attack algorithm. He et al. [42] realized link stealing attacks to infer whether a link between two nodes exists or not. Mu et al. [18] attacked GNNs to get the desired label for graph classification via perturbing the graph structure. Chang et al. [43] built an adversarial attacker, GF-Attack, to attack the graph embedding model in a black-box attack pattern. Until now, the adversarial attacks on social status have not been well explored. In essence, an adversarial attack on user status belongs to a node-oriented attack. However, different from the target of node classification attack, the purpose of node classification is to deceive the classification model and group the nodes into the desired category, while our attack on social status is to improve or reduce the status of the target node from the perspective of influence through imperceptible perturbation.

## VII. Conclusion

In this paper, we explore the feasibility of adversarial attacks on social status through the imperceptible distribution of network structure and propose a framework of adversarial attacks called $A^2S^2$-GNN. The key to $A^2S^2$-GNN is to find the neighbors within two hops that have the most influence on the target node. The experimental results and performance analysis on the benchmark datasets confirm the effectiveness and transferability of the proposed attack framework. Naturally, the attack algorithm can provide useful guidance for evaluating the influence of nodes in turn. To accurately evaluate the influence of nodes in complex networks, we can try various performance properties, such as considering the most influential nodes or edges selected by the attack algorithm. We leave this as our future work. In addition, the proposed attack model can be used to dispel rumors and spread network positive energy for online social networks by changing the status of the source node.

## References


[1] A. Giddens, M. Duneier, and R. Appelbaum, *Introduction to Sociology*. New York, NY, USA: WW Norton & Company, 2012.

[2] W. Lin and B. Li, "Medley: Predicting social trust in time-varying online social networks," in *Proc. IEEE Int. Conf. Comput. Commun.*, May 2021, pp. 1–10.

[3] W. Lin, Z. Gao, and B. Li, "Guardian: Evaluating trust in online social networks with graph convolutional networks," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, Jul. 2020, pp. 914–923.

[4] J. Yang and W. P. Tay, "An unsupervised Bayesian neural network for truth discovery in social networks," *Trans. Knowl. Data Eng.*, vol. 34, no. 11, pp. 5182–5195, Nov. 2022.

[5] T. Derr, C. Aggarwal, and J. Tang, "Signed network modeling based on structural balance theory," in *Proc. ACM CIKM*, 2018, pp. 557–566.

[6] Y. Wang, X. Wang, J. Tang, W. Zuo, and G. Cai, "Modeling status theory in trust prediction," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 1875–1881.

[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, May 2021.

[8] C. Li, J. Ma, X. Guo, and Q. Mei, "Deepcas: An end-to-end predictor of information cascades," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 577–586.

[9] W. Lin, S. Ji, and B. Li, "Adversarial attacks on link prediction algorithms based on graph neural networks," in *Proc. 15th ACM Asia Conf. Comput. Commun. Secur.*, 2020, pp. 370–380.

[10] J. Chen, X. Lin, Z. Shi, and Y. Liu, "Link prediction adversarial attack via iterative gradient attack," *Trans. Comput. Social Syst.*, vol. 7, no. 4, pp. 1081–1094, 2020.

[11] A. Bojchevski and S. Günnemann, "Adversarial attacks on node embeddings via graph poisoning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 695–704.

[12] M. Sun et al., "Data poisoning attack against unsupervised node embedding methods," 2018, *arXiv:1810.12881*.

[13] W. Lin and B. Li, "Status-aware signed heterogeneous network embedding with graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Feb. 28, 2022, doi: 10.1109/TNNLS.2022.3151046.

[14] D. Zügner, A. Akbarnejad, and S. Günnemann, "Adversarial attacks on neural networks for graph data," in *Proc. ACM SIGKDD*, 2018, pp. 2847–2856.

[15] B. Wang and N. Z. Gong, "Attacking graph-based classification via manipulating the graph structure," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 2023–2040.

[16] M. Zhang, L. Hu, C. Shi, and X. Wang, "Adversarial label-flipping attack and defense for graph neural networks," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2020, pp. 791–800.

[17] J. Li, H. Zhang, Z. Han, Y. Rong, H. Cheng, and J. Huang, "Adversarial attack on community detection by hiding individuals," in *Proc. Web Conf.*, 2020, pp. 917–927.

[18] J. Mu, B. Wang, Q. Li, K. Sun, M. Xu, and Z. Liu, "A hard label black-box adversarial attack against graph neural networks," in *Proc. 2021 ACM SIGSAC Conf. Comput. Commun. Secur.*, 2021, pp. 108–125.

[19] T. Derr, Y. Ma, and J. Tang, "Signed graph convolutional networks," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2018, pp. 929–934.

[20] H. Dai et al., "Adversarial attack on graph structured data," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1115–1124.

[21] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1025–1035.

[22] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2016, *arXiv:1609.02907*.
[23] E. Garfield, "Citation analysis as a tool in journal evaluation," *Science*, vol. 178, no. 4060, pp. 471–479, 1972.
[24] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Signed networks in social media," in *Proc. 28th Int. Conf. Hum. Factors Comput. Syst. (CHI)*, 2010, pp. 1361–1370.
[25] R. West, H. Paskov, J. Leskovec, and C. Potts, "Exploiting social network structure for person-to-person sentiment analysis," *Trans. Assoc. Comput. Linguistics*, vol. 2, pp. 297–310, Dec. 2014.
[26] J. Leskovec, D. Huttenlocher, and J. Kleinberg, "Predicting positive and negative links in online social networks," in *Proc. 19th Int. Conf. World Wide Web (WWW)*, 2010, pp. 641–650.
[27] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "ArnetMiner: Extraction and mining of academic social networks," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 990–998.
[28] J. Tang, H. Liu, H. Gao, and A. D. Sarmas, "ETrust: Understanding trust evolution in an online world," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2012, pp. 253–261.
[29] J. Tang, H. Gao, and H. Liu, "MTrust: Discerning multi-faceted trust in a connected world," in *Proc. 5th ACM Int. Conf. Web Search Data Mining (WSDM)*, 2012, pp. 93–102.
[30] M. Waniek, T. P. Michalak, M. J. Wooldridge, and T. Rahwan, "Hiding individuals and communities in a social network," *Nature Hum. Behav.*, vol. 2, no. 2, pp. 139–147, Jan. 2018.
[31] X. Zou et al., "TDGIA: Effective injection attacks on graph neural networks," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 2461–2471.
[32] H. Zhao, X. Xu, Y. Song, D. L. Lee, Z. Chen, and H. Gao, "Ranking users in social networks with higher-order structures," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, 2018, pp. 232–239.
[33] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, vol. 40, no. 1, pp. 35–41, 1977.
[34] G. Sabidussi, "The centrality index of a graph," *Psychometrika*, vol. 31, no. 4, pp. 581–603, Dec. 1966.
[35] J. Leskovec, "How status and reputation shape human evaluations: Consequences for recommender systems," in *Proc. 22nd Int. Conf. World Wide Web*, May 2013, pp. 937–938.
[36] E. Dai et al., "A comprehensive survey on trustworthy graph neural networks: Privacy, robustness, fairness, and explainability," 2022, *arXiv:2204.08570*.
[37] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2017, pp. 39–57.
[38] J. Tang, Y. Chang, C. Aggarwal, and H. Liu, "A survey of signed network mining in social media," *ACM Comput. Surv.*, vol. 49, no. 3, pp. 1–37, Dec. 2016.
[39] L. Sun et al., "Adversarial attack and defense on graph data: A survey," 2018, *arXiv:1812.10528*.
[40] W. Lin, S. Ji, and B. Li, "Adversarial attacks on link prediction algorithms based on graph neural networks," in *Proc. 15th ACM Asia Conf. Comput. Commun. Secur.*, Oct. 2020, pp. 370–380.
[41] H. Zhan and X. Pei, "Black-box gradient attack on graph neural networks: Deeper insights in graph-based attack and defense," 2021, *arXiv:2104.15061*.
[42] X. He, J. Jia, M. Backes, N. Z. Gong, and Y. Zhang, "Stealing links from graph neural networks," in *Proc. 30th Usenix Secur. Symp.*, 2021.
[43] H. Chang et al., "A restricted black-box adversarial framework towards attacking graph embedding models," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 4, 2020, pp. 3389–3396.

**Xiaoyan Yin** (Member, IEEE) received the Ph.D. degree in computer science from Northwestern Polytechnical University, Xi'an, China, in 2010. She is currently a Professor with the School of Information Science and Technology, Northwest University, Xi'an. Her research interests include the Internet of Things, network economy, social networks, and AI security.

**Wanyu Lin** (Member, IEEE) received the B.Eng. degree from the School of Electronic Information and Communications, Huazhong University of Science and Technology, China, in 2012, the M.Phil. degree from the Department of Computing, The Hong Kong Polytechnic University, in 2015, and the Ph.D. degree from the Department of Electrical and Computer Engineering, University of Toronto, in 2020. Her research interests include graph machine learning, trustworthy machine learning, data privacy, and model interpretability. She has served as an Associate Editor for IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.

**Kexin Sun** received the B.E. and M.S. degrees in software engineering from Northwest University, Xi'an, China, in 2019 and 2022, respectively. She is currently an Engineer at BYD Company. Her main research interests include social networks and data privacy protection.

**Chun Wei** received the B.E. degree in software engineering from the Xi'an University of Architecture and Technology, Xi'an, China, in 2019. She is currently pursuing the M.S. degree with the School of Information Science and Technology, Northwest University, Xi'an. Her main research interests include social networks and data privacy protection.

**Yanjiao Chen** (Senior Member, IEEE) received the B.E. degree in electronic engineering from Tsinghua University in 2010 and the Ph.D. degree in computer science and engineering from The Hong Kong University of Science and Technology in 2015. She is currently a Professor at Zhejiang University, Hangzhou, China. Her research interests include computer networks, wireless system security, and network economy.