Knowledge and Information Systems (2020) 62:571–610
https://doi.org/10.1007/s10115-019-01364-z

REGULAR PAPER

Random walk-based ranking in signed social networks: model and algorithms

Jinhong Jung[1] · Woojeong Jin[2] · U Kang[1]

Received: 15 June 2017 / Revised: 6 April 2019 / Accepted: 12 April 2019 / Published online: 6 May 2019

Abstract

How can we rank nodes in signed social networks? Relationships between nodes in a signed network are represented as positive (trust) or negative (distrust) edges. Many social networks have adopted signed networks to express trust between users. Consequently, ranking friends or enemies in signed networks has received much attention from the data mining community. The ranking problem, however, is challenging because it is difficult to interpret negative edges. Traditional random walk-based methods such as PageRank and random walk with restart cannot provide effective rankings in signed networks since they assume only positive edges. Although several methods have been proposed by modifying traditional ranking models, they also fail to account for proper rankings due to the lack of ability to consider complex edge relations. In this paper, we propose SIGNED RANDOM WALK WITH RESTART (SRWR), a novel model for personalized ranking in signed networks. We introduce a signed random surfer so that she considers negative edges by changing her sign for walking. Our model provides proper rankings considering signed edges based on the signed random walk. We develop two methods for computing SRWR scores: SRWR- ITER and SRWR- PRE which are iterative and preprocessing methods, respectively. SRWR- ITER naturally follows the definition of SRWR, and iteratively updates SRWR scores until convergence. SRWR- PRE enables fast ranking computation which is important for the performance of applications of SRWR. Through extensive experiments, we demonstrate that SRWR achieves the best accuracy for link prediction, predicts trolls 4× more accurately, and shows a satisfactory performance for inferring missing signs of edges compared to other competitors. In terms of efficiency, SRWR- PRE preprocesses a signed network 4.5× faster and requires 11× less memory space than other preprocessing methods; furthermore, SRWR- PRE computes SRWR scores up to 14× faster than other methods in the query phase.

Keywords Signed networks · Signed random walk with restart · Personalized node ranking · Trustworthiness measure

✉ U Kang
ukang@snu.ac.kr

[1] Department of Computer Science and Engineering, Seoul National University, Seoul, Republic of Korea

[2] Department of Computer Science, University of Southern California, Los Angeles, USA

1 Introduction

How can we obtain personalized rankings for users in signed social networks? Many social networks have allowed users to express their trust or distrust to other users. For example, in online social networks such as Slashdot [23], a user is explicitly able to mark other users as friends or foes. The users are represented as nodes, and the expressions are represented as positive and negative edges in graphs which are called *signed networks* [37]. Ranking nodes in signed networks has received much interest from data mining community to reveal trust and distrust between users [23] inducing many useful applications such as link prediction [35], anomaly detection [23], sign prediction [26], and community detection [42] in signed networks.

Traditional ranking models, however, do not provide satisfactory node rankings in signed networks. Existing random walk-based ranking models such as PageRank [31] and Random Walk with Restart [17,18,34,39,43,44] assume only positive edges; thus, they are inappropriate in the signed networks containing negative edges. Many researchers have proposed heuristics on the classical methods to make them computable in signed networks [23,33]. However, those heuristic methods still have room to improve in terms of ranking quality since they do not consider complex social relationships such as friend of enemy or enemy of friend in their rankings as shown in Fig. 2. In addition, most existing ranking models in signed networks focus only on a global node ranking, although personalized rankings are more desirable for individuals in many contexts such as recommendation. Also, the fast ranking computation is important for the computational performance of applications in SRWR.

In this paper, we propose Signed Random Walk with Restart (SRWR), a novel model for effective personalized node rankings in signed networks. The main idea of SRWR is to introduce a sign into a random surfer in order to let the surfer consider negative edges based on structural balance theory [5,26]. Consequently, our model considers complex edge relationships and makes random walks interpretable in signed networks. We devise SRWR-Iter, an iterative method which naturally follows the definition of SRWR, and iteratively update SRWR scores until convergence. Furthermore, we propose SRWR- Pre, a preprocessing method for computing SRWR scores quickly which is useful for various applications in signed networks. Through extensive experiments, we demonstrate that our proposed approach offers improved performance for personalized rankings compared to alternative methods in signed social networks. Our main contributions are as follows:

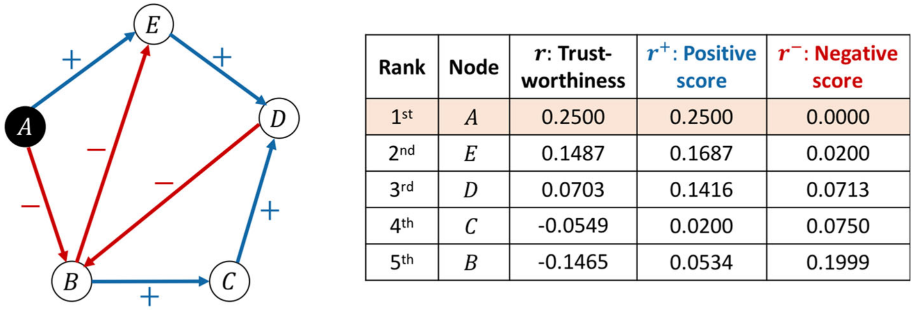
- **Novel ranking model** We propose Signed Random Walk with Restart (SRWR), a novel model for personalized rankings in signed networks (Definition 1). We show that our model is a generalized version of RWR working on both signed and unsigned networks (Property 2).
- **Algorithm** We propose SRWR- Iter and SRWR- Pre for computing SRWR scores. SRWR- Iter is an iterative algorithm which naturally follows the definition of SRWR (Algorithm 2). SRWR- Pre is a preprocessing method which employs a node reordering technique and block elimination to accelerate SRWR computation speed (Algorithms 3 and 4).
- **Experiment** We show that SRWR achieves higher accuracy for link prediction (Fig. 7), predicts trolls 4× more accurately (Fig. 9), and provides a good performance for sign prediction compared to other ranking models (Fig. 10). In terms of efficiency, SRWR-Pre preprocesses signed networks up to 4.5× faster, and requires 11× less memory space than baseline preprocessing methods. Furthermore, SRWR- Pre computes SRWR scores up to 14× faster than other methods including SRWR- Iter (Fig. 13).

Table 1 Table of symbols

Symbol	Definition
$G = (\mathbf{V}, \mathbf{E})$	Signed input graph
$\mathbf{V}$	Set of nodes in G
$\mathbf{E}$	Set of signed edges in G
n	Number of nodes in G
n_1	Number of spokes in G
n_2	Number of hubs in G
m	Number of edges in G
s	Seed node (= query node, source node)
c	Restart probability
ϵ	Error tolerance
$\overleftarrow{\mathbf{N}}_u$	Set of in-neighbors to nodes u
$\overrightarrow{\mathbf{N}}_u$	Set of out-neighbors from nodes u
$\mathbf{A}$	$(n \times n)$ signed adjacency matrix of G
$\lvert\mathbf{A}\rvert$	$(n \times n)$ absolute adjacency matrix of G
$\mathbf{D}$	$(n \times n)$ out-degree matrix of $\lvert\mathbf{A}\rvert$, $\mathbf{D}_{ii} = \sum_j \lvert\mathbf{A}\rvert_{ij}$
$\tilde{\mathbf{A}}$	$(n \times n)$ semi-row-normalized matrix of $\mathbf{A}$
$\tilde{\mathbf{A}}_+$	$(n \times n)$ positive semi-row-normalized matrix of $\mathbf{A}$
$\tilde{\mathbf{A}}_-$	$(n \times n)$ negative semi-row-normalized matrix of $\mathbf{A}$
$\lvert\tilde{\mathbf{A}}\rvert$	$(n \times n)$ absolute row-normalized matrix of $\lvert\mathbf{A}\rvert$
$\mathbf{q}$	$(n \times 1)$ starting vector (= sth unit vector)
$\mathbf{r}^+$	$(n \times 1)$ positive score vector
$\mathbf{r}^-$	$(n \times 1)$ negative score vector
$\mathbf{r}$	$(n \times 1)$ trustworthiness score vector, e.g., $\mathbf{r} = \mathbf{r}^+ - \mathbf{r}^-$
$\mathbf{p}$	$(n \times 1)$ $\mathbf{p} = \mathbf{r}^+ + \mathbf{r}^-$
$\lvert\mathbf{H}\rvert$	$(n \times n)$ $\lvert\mathbf{H}\rvert = \mathbf{I} - (1-c)\lvert\tilde{\mathbf{A}}\rvert^\top$
$\mathbf{T}$	$(n \times n)$ $\mathbf{T} = \mathbf{I} - (1-c)(\gamma\tilde{\mathbf{A}}_+^\top - \beta\tilde{\mathbf{A}}_-^\top)$
$\lvert\mathbf{H}\rvert_{ij}, \mathbf{T}_{ij}$	$(n_i \times n_j)$ (i, j)th partition of $\lvert\mathbf{H}\rvert$ or $\mathbf{T}$
$\mathbf{S}_{\lvert\mathbf{H}\rvert}, \mathbf{S}_{\mathbf{T}}$	$(n_2 \times n_2)$ Schur complement of $\lvert\mathbf{H}\rvert_{11}$ or $\mathbf{T}_{11}$
$\mathbf{q}_i, \mathbf{p}_i, \mathbf{r}_i^-$	$(n_i \times 1)$ ith partition of $\mathbf{q}$, $\mathbf{p}$ or $\mathbf{r}^-$

Boldface capital letters, such as $\mathbf{A}$, represent matrices. Boldface small letters, such as $\mathbf{r}$, represent vectors

The code of our method and datasets used in this paper are available at http://datalab.snu.ac.kr/srwrpre. The rest of this paper is organized as follows. We first introduce the formal definition of the personalized ranking problem in signed networks at Sect. 2. Then we provide a review of related works in Sect. 3. In Sect. 4, we describe our proposed model and algorithms for computing personalized rankings. After presenting experimental results in Sect. 5, we conclude in Sect. 6. Table 1 lists the symbols used in this paper.



Rank	Node	r: Trustworthiness	r^+: Positive score	r^-: Negative score
1st	A	0.2500	0.2500	0.0000
2nd	E	0.1487	0.1687	0.0200
3rd	D	0.0703	0.1416	0.0713
4th	C	-0.0549	0.0200	0.0750
5th	B	-0.1465	0.0534	0.1999

Input: a signed network & seed node A

Output: the trustworthiness score vector r w.r.t. the seed node

Fig. 1 Example of the personalized node ranking problem in Problem 1. Given a signed network and a seed node (in this example, node A is the seed node), our goal is to compute the trustworthiness score vector $\mathbf{r}$ w.r.t. the seed node. Our proposed model SRWR (see Definition 1 in Sect. 4) aims to compute $\mathbf{r}$ based on the positive and negative score vectors $\mathbf{r}^+$ and $\mathbf{r}^-$, i.e., $\mathbf{r} = \mathbf{r}^+ - \mathbf{r}^-$

2 Problem definition

We define the personalized ranking problem in signed networks as follows:

Problem 1 *(Personalized Node Ranking in Signed Networks)*

- ***Input:*** *a signed network* $G = (\mathbf{V}, \mathbf{E})$ *and a seed node* s *where* $\mathbf{V}$ *is the set of nodes, and* $\mathbf{E}$ *is the set of signed edges.*
- ***Output:*** *a trustworthiness score vector* $\mathbf{r} \in \mathbb{R}^n$ *of all other nodes for seed node* s *to rank those nodes w.r.t. seed node* s. □

In signed social networks, users are represented as nodes, and trust or distrust relations between users are represented as positive or negative edges. When a user u considers that a user v is trustworthy, a positive edge $u \rightarrow v$ is formed. On the contrary, a negative edge $u \rightarrow v$ is formed when u distrusts v. Given those signed edges between nodes and a seed node s, the personalized ranking problem is to rank all other nodes w.r.t. seed node s in the order of trustworthiness scores represented by $\mathbf{r}$ where $\mathbf{r}_u$ indicates how much seed node s should trust node u as depicted in Fig. 1. If the score $\mathbf{r}_u$ is high, then s is likely to trust u. Otherwise, s is likely to distrust u.

3 Related work

In this section, we review related works, which are categorized into four parts: (1) ranking in unsigned networks, (2) ranking in signed networks, (3) applications in signed networks, and (4) fast personalized ranking methods.

Ranking in unsigned networks There are various global ranking measures based on link structure and random walk, e.g., PageRank (PR) [31], HITS [21], and SALSA [25]. Furthermore, personalized ranking methods are explored in terms of node-to-node relevance such as Random Walk with Restart (RWR) [40], Personalized PageRank (PPR) [13], Personalized SALSA (PSALSA) [2]. Among these measures, RWR has received much interests and has been applied to many applications [1,10,15,20]. Note that these methods are not applicable to signed graphs because they assume only positive edges; on the contrary, our model works on signed networks as well as on unsigned networks.

Ranking in signed networks Many researchers have made great efforts to design global node rankings in signed networks. Kunegis et al. [23] presented Signed spectral Ranking (SR) that heuristically computes PageRank scores based on a signed adjacency matrix. Wu et al. [41] proposed Troll-Trust model (TR-TR) which is a variant of PageRank. In the algorithm, the trustworthiness of an individual user is modeled as a probability that represents the underlying ranking values. Shahriari et al. [33] suggested Modified PageRank (MPR), which computes PageRank in a positive subgraph and a negative subgraph separately, and subtracts negative PageRank scores from positive ones. Although the idea of MPR is easily applicable to other personalized ranking models such as RWR by computing ranking scores on the positive and negative subgraphs, this results in many disconnections between nodes. Note that all those models mainly focus on global node rankings, and they do not consider complex relationships between negative and positive edges such as friend of enemy or enemy of friend; in contrast, our model SRWR provides an effective personalized ranking with considering complicated relationships between nodes based on a social theory such as structural balance theory [5].

Applications in signed networks Numerous applications in signed social networks such as link prediction, troll detection, and sign prediction have been studied in many literatures. Song et al. [35] proposed GAUC (Generalized AUC) to measure the quality of link prediction in signed networks where the link prediction task is to predict nodes which will be positively or negatively linked by a node in the future. They devised a matrix factorization-based method GAUC-OPT which approximately maximizes GAUC for link prediction. Kunegis et al. [23] analyzed the Slashdot dataset from the perspective of troll detection, and proposed Negative Rank (NR) as a variant of PageRank for detecting trolls who behave abnormally in the social network. Leskovec et al. [26] proposed LOGIT which is specially designed for sign prediction classifying the sign between two arbitrary nodes. They exploited a logistic classifier trained by node and edge features such as node degrees and common neighbors between those two nodes. Guha et al. [12] also studied sign prediction and devised TRUST measuring trustworthiness between two source and target nodes by propagating trust and distrust from the source node to the target node. Note that our model SRWR shows better performance in link prediction, troll detection, and sign prediction tasks compared to those methods as demonstrated in Sect. 5.

Fast personalized ranking methods Many researchers have emphasized the importance of fast computation for personalized rankings such as RWR to reduce their computational cost and boost the performance of applications based on ranking in terms of efficiency. Tong et al. [40] proposed an approximate method which exploits a low-rank approximation based on matrix decomposition in the preprocessing phase, and computes an RWR query from the decomposed matrices in the query phase. Fujiwara et al. utilized LU factorization [9] with degree ordering to speed up the RWR computation. Shin et al. [34] proposed a block elimination approach based on node reordering to accelerate RWR computation speed. Although those approaches significantly increase the performance of ranking in terms of running time, they only focus on ranking in unsigned networks. In our previous work [16], we designed a random surfer model for ranking nodes in signed networks, and developed an iterative method for computing trust and distrust scores. However, the iterative method is not appropriate for real-time applications since the method is not fast in large signed networks as shown in Fig. 13c. In this work, we also aim to develop an efficient preprocessing method to accelerate the query speed of SRWR in signed networks. We will demonstrate that our

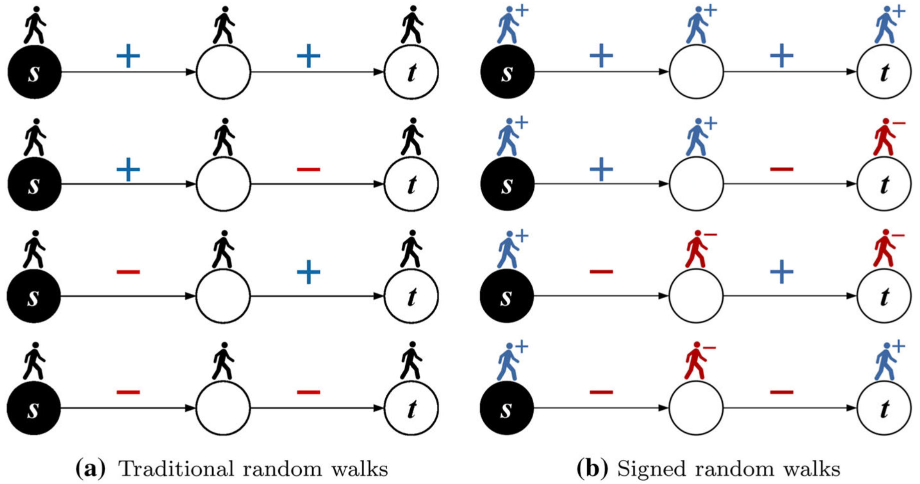


Fig. 2 Examples of traditional random walks and signed random walks. Each case represents (1) friend's friend, (2) friend's enemy, (3) enemy's friend, or (4) enemy's enemy from the top. A random surfer has either a positive (blue) or a negative (red) sign on each node in (**b**). When the signed surfer traverses a negative edge, she changes her sign from positive to negative or vice versa (color figure online)

preprocessing method SRWR- PRE is the fastest for computing SRWR scores among other baselines as presented in Fig. 13c.

4 Proposed methods

We propose SIGNED RANDOM WALK WITH RESTART (SRWR), a novel ranking model for signed networks in Sect. 4.1. Then we first develop an iterative algorithm SRWR- ITER for computing SRWR scores w.r.t. a seed node in Sect. 4.2, and then propose a preprocessing algorithm SRWR- PRE to accelerate SRWR computation speed in Sect. 4.3.

4.1 Signed random walk with restart model

As discussed in Sect. 1, complicated relationships of signed edges are the main obstacles for providing effective rankings in signed networks. Most existing works on signed networks have not focused on personalized rankings. In this work, our goal is to design a novel ranking model which resolves those problems in signed networks. The main ideas of our model are as follows:

- We introduce a signed random surfer. The sign of the surfer is either positive or negative, which means favorable or adversarial to a node, respectively.
- When the random surfer encounters a negative edge, she changes her sign from positive to negative, or vice versa. Otherwise, she keeps her sign.
- We introduce balance attenuation factors into the surfer to consider the uncertainty for friendship of enemies.

There are four cases according to the signs of edges as shown in Fig. 2: (1) friend's friend, (2) friend's enemy, (3) enemy's friend, and (4) enemy's enemy. Suppose a random surfer starts at node s toward node t. A traditional surfer just moves along the edges without considering signs as seen in Fig. 2a since there is no way to consider the signs on the edges. Hence, classical models cannot distinguish those edge relationships during her walks. For instance, the model considers that node s and node t are friends for the second case (friend's enemy), even though node t are more likely to be an enemy w.r.t. node s.

On the contrary, our model in Fig. 2b has a signed random surfer who considers those complex edge relationships. If the random surfer starting at node s with a positive sign encounters a negative edge, she flips her sign from positive to negative, or vice versa. Our model distinguishes whether node t is the friend of node s or not according to her sign at node t. As shown in Fig. 2b, the results for all cases from our model are consistent with structural balance theory [5]. Thus, introducing a signed random surfer enables our model to discriminate those edge relationships.

Trust or distrust relationships between a specific node s and other nodes are revealed as the surfer is allowed to move around a signed network starting from node s. If the positive surfer visits a certain node u many times, then node u is trustable for node s. On the other hand, if the negative surfer visits node u many times, then node s is not likely to trust node u. Thus, rankings are obtained by revealing a degree of trust or distrust between people based on the signed random walks. Here, we formally define our model on signed networks in Definition 1. Note that Definition 1 involves the concept of restart which provides personalized rankings w.r.t. a user.

Definition 1 (*Signed Random Walk with Restart*) A *signed random surfer* has a sign, which is either positive or negative. At the beginning, the surfer starts with $+$ sign from a seed node s because she trusts s. Suppose the surfer is currently at node u, and c is the restart probability of the surfer. Then, she takes one of the following actions:

- **Action 1: Signed Random Walk** The surfer randomly moves to one of the neighbors from node u with probability $1 - c$. The surfer flips her sign if she encounters a negative edge. Otherwise, she keeps her sign.
- **Action 2: Restart** The surfer goes back to the seed node s with probability c. Her sign should become $+$ at the seed node s because she trusts s. □

We measure two probabilities on each node through Signed Random Walk with Restart (SRWR) starting from the seed node s. The two probabilities are represented as follows:

- $\mathbf{r}_u^+ = P(u, +)$: the probability that the positive surfer visits node u after SRWR from seed node s.
- $\mathbf{r}_u^- = P(u, -)$: the probability that the negative surfer visits node u after SRWR from seed node s.

Note that $\mathbf{r}_u^+$ (or $\mathbf{r}_u^-$) corresponds to a ratio of how many times the positive (or negative) surfer visits node u during SRWR. If the positive surfer visits node u much more than the negative one, then s is likely to trust u. Otherwise, s is likely to distrust u. In other words, s would consider u as a positive node if $\mathbf{r}_u^+$ is greater than $\mathbf{r}_u^-$. On the contrary, s would treat u as a negative one if $\mathbf{r}_u^-$ is greater than $\mathbf{r}_u^+$. Based on this intuition, we define the relative trustworthiness score $\mathbf{r}_u = \mathbf{r}_u^+ - \mathbf{r}_u^-$ between s and u. For all nodes, $\mathbf{r}^+$ is a positive score vector and $\mathbf{r}^-$ is a negative score vector of SRWR. Then, the trustworthiness score vector for SRWR is represented as $\mathbf{r} = \mathbf{r}^+ - \mathbf{r}^-$, the output of Problem 1. Many researchers have dealt with trust and distrust between nodes through such representation for trustworthiness [12,23,29,33]. Particularly, the interpretation of the resulting values from $\mathbf{r}_u = \mathbf{r}_u^+ - \mathbf{r}_u^-$ is consistent with what Kunegis et al. said as follows:

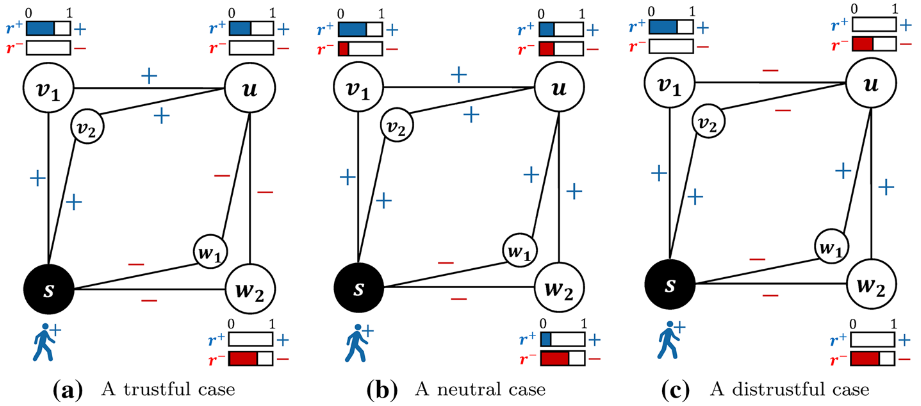


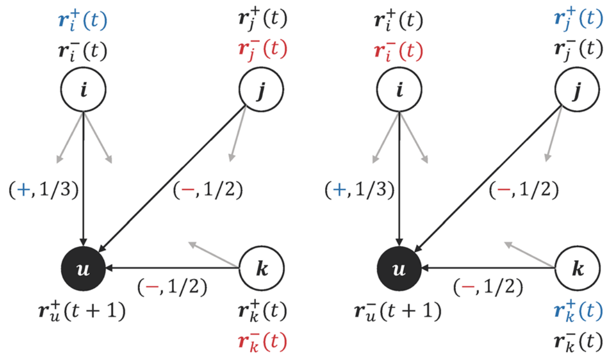
Fig. 3 Examples of how to interpret positive and negative scores of our model between nodes s and u. The bars on node u depict how many the signed surfer visits that node, indicating positive and negative scores between s and u. **a** and **c** represent trustful and distrustful cases between those nodes: s is likely to trust u in (**a**), and s is likely to distrust u in (**c**). However, if the those scores are similar as in (**b**), it is difficult for node s to decide whether to trust node u or not. Hence, s is likely to be neutral about node u in (**b**)

- *"The resulting popularity (based on trustworthiness) measure admits both positive and negative values, and represents a measure of popularity in the network, with positive edges corresponding to a positive endorsement and negative edges to negative endorsements. This interpretation is consistent with the semantics of the 'friend' and 'foe' relationships* [23]."

Note that from the viewpoint of measure theory, the relative trustworthiness $\mathbf{r}_u$ is also an acceptable measure as *signed measure* [38] if we consider $\mathbf{r}_u^+$ and $\mathbf{r}_u^-$ as nonnegative measures (i.e., $\mathbf{r}_u^+ \geq 0$ and $\mathbf{r}_u^- \geq 0$). We discuss this in detail in Appendix A.6.

Discussion on positive and negative SRWR scores We explain how to interpret positive and negative SRWR scores using an example in Fig. 3. Suppose the signed surfer starts at node s, and performs SRWR to measure the trustworthiness between nodes s and u. Note that the trustworthiness score depends on which signed surfer stays at node u more frequently. Then, there would be three cases depending on the link structure between s and u as shown in Fig. 3. For the case in Fig. 3a, s is likely to trust u since the positive surfer visits u much more than the negative surfer through paths from s to u (i.e., the positive score is larger than the negative one at u). For the opposite case in Fig. 3c, s is likely to distrust u because the negative surfer frequently visits u. However, if those scores on u are similar as shown in Fig. 3b, then it is hard for s to determine whether to trust u or not. In this case s is likely to be neutral about node u. Thus, the trustworthiness score $\mathbf{r}_u$ of the trustful case is high (and positive in SRWR), and that of the distrustful case is low (and negative in SRWR). For the neutral case, the score would be in the middle (and around zero between −1 and 1 in SRWR).

Connection to balance theory According to balance theory [5,8], Fig. 3a, c is the balanced networks because the graphs are divided into two sets of users with mutual antagonism between the sets. For example, the set of nodes $\{v_1, v_2, s\}$ and the other set of nodes $\{w_1, w_2, u\}$ in Fig. 3c are connected with negative edges, and nodes in each set are positively connected. In the balanced networks, each node has either a positive score or a negative one. Because the signed surfer changes her sign walking negative edges linking the two groups, the positive surfer stays and walks only in one group and the negative surfer stays and walks only in the other group. However, Fig. 3b is an unbalanced network because it cannot be



(a) An example of a positive probability, $\mathbf{r}_u^+(t+1)$ **(b)** An example of a negative probability, $\mathbf{r}_u^-(t+1)$

Fig. 4 Examples of how $\mathbf{r}_u^+$ and $\mathbf{r}_u^-$ are defined in SRWR

divided into two sets that are negatively connected each other. Hence, positive and negative surfers visits the same node, i.e., each node has both positive and negative scores. In this case, the trustworthiness score on a node is determined by which signed surfer visits the node more frequently, which is represented by the difference between positive and negative scores.

4.1.1 Formulation for signed random walk with restart

We formulate the probability vectors, $\mathbf{r}^+$ and $\mathbf{r}^-$, following Signed Random Walk with Restart. First, we explain how to define $\mathbf{r}^+$ and $\mathbf{r}^-$ using the example shown in Fig. 4. In the example, we label a (sign, transition probability) pair on each edge. For instance, the transition probability for the positive edge from node i to node u is $1/3$ because node i has 3 outgoing edges. This edge is denoted by $(+, 1/3)$. Other pairs of signs and transition probabilities are also similarly defined. In order that the random surfer has a positive sign on node u at time $t+1$, a positive surfer on one of u's neighbor at time t must move to node u through a positive edge, or a negative surfer must move through a negative edge according to the signed random walk action in Definition 1. Considering the restart action of the surfer with the probability c, $\mathbf{r}_u^+(t+1)$ in Fig. 4a is represented as follows:

$$\mathbf{r}_u^+(t+1) = (1-c)\left(\frac{\mathbf{r}_i^+(t)}{3} + \frac{\mathbf{r}_j^-(t)}{2} + \frac{\mathbf{r}_k^-(t)}{2}\right) + c\mathbf{1}(u=s)$$

where $\mathbf{1}(u=s)$ is 1 if u is the seed node s and 0 otherwise. In Fig. 4b, $\mathbf{r}_u^-(t+1)$ is defined similarly as follows:

$$\mathbf{r}_u^-(t+1) = (1-c)\left(\frac{\mathbf{r}_i^-(t)}{3} + \frac{\mathbf{r}_j^+(t)}{2} + \frac{\mathbf{r}_k^+(t)}{2}\right)$$

Note that we do not add the restarting score $c\mathbf{1}(u=s)$ to $\mathbf{r}_u^-(t+1)$ in this case because the surfer's sign must become positive when she goes back to the seed node s. The recursive equations of our model are defined as follows:

$$\begin{aligned}\mathbf{r}_u^+ &= (1-c)\left(\sum_{v \in \overleftarrow{\mathbf{N}}_u^+} \frac{\mathbf{r}_v^+}{|\overrightarrow{\mathbf{N}}_v|} + \sum_{v \in \overleftarrow{\mathbf{N}}_u^-} \frac{\mathbf{r}_v^-}{|\overrightarrow{\mathbf{N}}_v|}\right) + c\mathbf{1}(u = s)\\ \mathbf{r}_u^- &= (1-c)\left(\sum_{v \in \overleftarrow{\mathbf{N}}_u^-} \frac{\mathbf{r}_v^+}{|\overrightarrow{\mathbf{N}}_v|} + \sum_{v \in \overleftarrow{\mathbf{N}}_u^+} \frac{\mathbf{r}_v^-}{|\overrightarrow{\mathbf{N}}_v|}\right)\end{aligned} \tag{1}$$

where $\overleftarrow{\mathbf{N}}_i$ is the set of in-neighbors of node i, and $\overrightarrow{\mathbf{N}}_i$ is the set of out-neighbors of node i. Superscripts of $\overleftarrow{\mathbf{N}}_i$ or $\overrightarrow{\mathbf{N}}_i$ indicate signs of edges between node i and its neighbors (e.g., $\overleftarrow{\mathbf{N}}_i^+$ indicates the set of positively connected in-neighbors of node i). We need to introduce several symbols related to an adjacency matrix $\mathbf{A}$ to vectorize Eq. (1).

Definition 2 (*Signed adjacency matrix*) The signed adjacency matrix $\mathbf{A}$ of G is a matrix such that $\mathbf{A}_{uv}$ is positive or negative when there is a positive or a negative edge from node u to node v, respectively, and zero otherwise. □

Definition 3 (*Semi-row-normalized matrix*) Let $|\mathbf{A}|$ be the absolute adjacency matrix of $\mathbf{A}$, and $\mathbf{D}$ be the out-degree diagonal matrix of $|\mathbf{A}|$ (i.e., $\mathbf{D}_{ii} = \sum_j |\mathbf{A}|_{ij}$). Then semi-row-normalized matrix of $\mathbf{A}$ is $\tilde{\mathbf{A}} = \mathbf{D}^{-1}\mathbf{A}$. □

Definition 4 (*Positive or negative semi-row-normalized matrix*) The positive semi-row-normalized matrix $\tilde{\mathbf{A}}_+$ contains only positive values in the semi-row-normalized matrix $\tilde{\mathbf{A}}$. The negative semi-row-normalized matrix $\tilde{\mathbf{A}}_-$ contains absolute values of negative elements in $\tilde{\mathbf{A}}$. In other words, $\tilde{\mathbf{A}} = \tilde{\mathbf{A}}_+ - \tilde{\mathbf{A}}_-$, and $|\tilde{\mathbf{A}}| = \tilde{\mathbf{A}}_+ + \tilde{\mathbf{A}}_-$. □

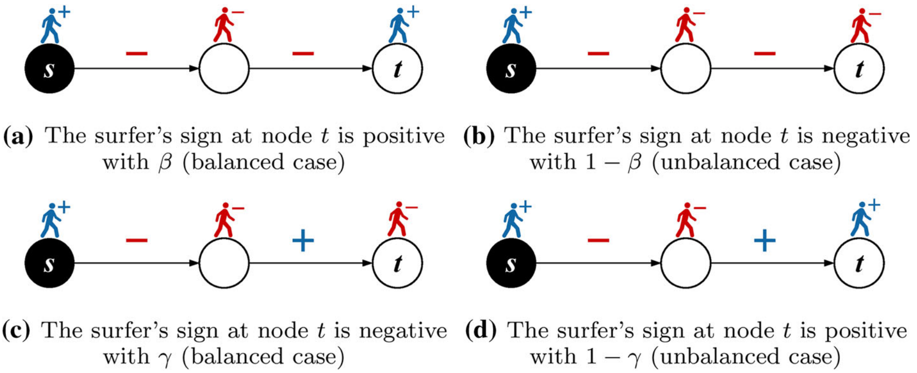
Based on Definitions 3 and 4, Eq. (1) is represented as follows:

$$\begin{aligned}\mathbf{r}^+ &= (1-c)\left(\tilde{\mathbf{A}}_+^\top \mathbf{r}^+ + \tilde{\mathbf{A}}_-^\top \mathbf{r}^-\right) + c\mathbf{q}\\ \mathbf{r}^- &= (1-c)\left(\tilde{\mathbf{A}}_-^\top \mathbf{r}^+ + \tilde{\mathbf{A}}_+^\top \mathbf{r}^-\right)\end{aligned} \tag{2}$$

where $\mathbf{q}$ is a vector whose sth element is 1 and all other elements are 0.

4.1.2 Balance attenuation factors

The signed surfer measures positive and negative scores of nodes w.r.t. a seed node in terms of trust and distrust according to edge relationships as discussed in Sect. 4.1. Our model in Definition 1 strongly supports the four cases between nodes in Fig. 2b where those cases represent *strong balance theory* [5,14]. However, recent works [27] have argued that the strong balance theory is unsatisfactory for fully supporting real-world signed networks, since unbalanced relationships frequently appear. Thus, this limitation would be naturally inherent in our model. To alleviate this limitation, many researchers have studied *weak balance theory* [6,27] which generalizes the strong balance theory by allowing several unbalanced cases such as "the enemy of my enemy is my enemy." Similarly, we adopt the generalization strategy of the weak balance theory to make our model flexible on unbalanced networks through dealing with both balanced and unbalanced cases.



(a) The surfer's sign at node t is positive with β (balanced case) **(b)** The surfer's sign at node t is negative with $1-\beta$ (unbalanced case)

(c) The surfer's sign at node t is negative with γ (balanced case) **(d)** The surfer's sign at node t is positive with $1-\gamma$ (unbalanced case)

Fig. 5 Examples of balance attenuation factors. **a** and **b** represent the uncertainty for "the enemy of my enemy is my friend" with probability β, and **c** and **d** represent the uncertainty for "the friend of my enemy is my enemy" with probability γ

We consider that the relationship of enemies of a seed user is uncertain since the user cannot believe the information provided by her enemies. We reflect the uncertainty of the relationship of those enemies into our ranking model by introducing stochastic parameters, β and γ, called *balance attenuation factors*. Note that we assume that the positive and negative relationship of friends of the seed user is reliable since the user trusts her friends. β is a parameter for the uncertainty of "the enemy of my enemy is my friend," and γ is for "the friend of my enemy is my enemy." We first explain β using the fourth case (enemy's enemy) in Fig. 2b. Suppose a surfer with a positive sign starts at node s toward node t and encounters two consecutive negative edges. Based on strong balance theory, her sign becomes negative at the intermediate node m and positive at node t in Fig. 5a. However, some people might think that the enemy of my enemy is my enemy as shown in Fig. 5b. In this case, her sign will be negative at nodes m and t. To consider this uncertainty, we introduce a parameter β so that if the negative surfer at node m encounters a negative edge, her sign becomes positive with probability β or negative with $1-\beta$ at node t. The other parameter γ is also interpreted similarly to β. When the negative surfer at node m encounters a positive edge, her sign will be negative with probability γ or positive with $1-\gamma$ at node t as in Fig. 5c, d. SRWR with the balance attenuation factors is represented as follows:

$$\begin{aligned}\mathbf{r}^{+} &= (1-c)\left(\tilde{\mathbf{A}}_{+}^{\top}\mathbf{r}^{+} + \beta\tilde{\mathbf{A}}_{-}^{\top}\mathbf{r}^{-} + (1-\gamma)\tilde{\mathbf{A}}_{+}^{\top}\mathbf{r}^{-}\right) + c\mathbf{q}\\ \mathbf{r}^{-} &= (1-c)\left(\tilde{\mathbf{A}}_{-}^{\top}\mathbf{r}^{+} + \gamma\tilde{\mathbf{A}}_{+}^{\top}\mathbf{r}^{-} + (1-\beta)\tilde{\mathbf{A}}_{-}^{\top}\mathbf{r}^{-}\right)\end{aligned} \tag{3}$$

Discussion on other balance attenuation factors Note that other parameters for the uncertainties of "enemy of friend" and "friend of friend" could be easily adopted into our model. However, we do not reflect those parameters on our model with the following reasons:

- As described in this subsection, we assume that the positive and negative relationship of friends of a seed user is reliable and stable. If the seed user's friends distrust a user, then she is unlikely to believe the user since the user trusts her friends.
- Introducing the additional parameters could improve the performance of applications in signed networks, but it increases the complexity of our model considering too many uncertain cases. We consider that introducing β and γ achieves a good trade-off between the model complexity and the performance of each application as shown in Sect. 5.

Algorithm 1: Normalization phase of SRWR- ITER

```
Input: signed adjacency matrix: $\mathbf{A}$
Output: positive semi-row-normalized matrix: $\tilde{\mathbf{A}}_+$, and negative semi-row-normalized matrix: $\tilde{\mathbf{A}}_-$
1: compute out-degree matrix $\mathbf{D}$ of $|\mathbf{A}|$, $\mathbf{D}_{ii} = \sum_j |\mathbf{A}|_{ij}$
2: compute semi-row-normalized matrix, $\tilde{\mathbf{A}} = \mathbf{D}^{-1}\mathbf{A}$.
3: split $\tilde{\mathbf{A}}$ into $\tilde{\mathbf{A}}_+$ and $\tilde{\mathbf{A}}_-$ such that $\tilde{\mathbf{A}} = \tilde{\mathbf{A}}_+ - \tilde{\mathbf{A}}_-$
4: return $\tilde{\mathbf{A}}_+$ and $\tilde{\mathbf{A}}_-$
```

Discussion on the initial sign In Definition 1, we initialize the signed surfer as positive when she restarts at a seed node s. One might consider that our model is easily extendable to probabilistically initializing the signed surfer as negative for the restart action. Let p denote the probability of being the positive surfer for the restart action. Then, the extended version is established by changing $c\mathbf{q}$ to $(c \times p)\mathbf{q}$ in the first equation and adding $(c \times (1 - p))\mathbf{q}$ into the second equation of Eq. (3). However, we do not consider such case with the following reason:

– If the negative surfer starts at s, the surfer becomes positive at nodes negatively connected from s and negative at those positively connected from s. This implies that the surfer recognizes the friends of s as enemies and the enemies of s as friends. Thus, it is hard to interpret the scores measured by the negative initial surfer in terms of trustworthiness for s based on balance theory.

4.2 SRWR-ITER: iterative algorithm for signed random walk with restart

We present an iterative algorithm SRWR- ITER for computing SRWR scores based on Eq. (3). Note that the solution of a linear system with recursive structure is typically and efficiently obtained via an iterative manner such as power iteration and Jacobi method [36]. We also adopt such iterative strategy to solve the recursive equations in Eq. (3). We describe how SRWR- ITER obtains the trustworthiness SRWR score vector $\mathbf{r}$ given a signed network and a seed node in Algorithms 1 and 2 . Moreover, we prove that the iterative approach in SRWR-ITER converges, and returns a unique solution for the seed node in Theorem 1 of Sect. 4.2.1.

Normalization phase (Algorithm 1) Our proposed algorithm first computes the out-degree diagonal matrix $\mathbf{D}$ of $|\mathbf{A}|$, which is the absolute adjacency matrix of $\mathbf{A}$ (line 1). Then, the algorithm computes the semi-row-normalized matrix $\tilde{\mathbf{A}}$ using $\mathbf{D}$ (line 2). We split $\tilde{\mathbf{A}}$ into two matrices: the positive semi-row-normalized matrix ($\tilde{\mathbf{A}}_+$) and the negative semi-row-normalized matrix ($\tilde{\mathbf{A}}_-$) (line 3) satisfying $\tilde{\mathbf{A}} = \tilde{\mathbf{A}}_+ - \tilde{\mathbf{A}}_-$.

Iteration phase (Algorithm 2) Our algorithm computes the SRWR score vectors $\mathbf{r}^+$ and $\mathbf{r}^-$ for the seed node s with the balance attenuation factors (β and γ) in the iteration phase. We set $\mathbf{q}$ to sth unit vector, and initialize $\mathbf{r}^+$ to $\mathbf{q}$ and $\mathbf{r}^-$ to $\mathbf{0}$ (lines 1 and 2). Our algorithm iteratively computes Eq. (3) (lines 4 and 5). We concatenate $\mathbf{r}^+$ and $\mathbf{r}^-$ vertically (line 6) into $\mathbf{h}$. We then compute the error δ between $\mathbf{h}$ and $\mathbf{h}'$ which is the result in the previous iteration (line 7). We update $\mathbf{h}$ into $\mathbf{h}'$ for the next iteration (line 8). The iteration stops when the error δ is smaller than a threshold ϵ (line 9). We finally return the trustworthiness score vector $\mathbf{r}$ used for the personalized ranking w.r.t. s by computing $\mathbf{r} = \mathbf{r}^+ - \mathbf{r}^-$ (lines 10 and 11).

The space and time complexities of Algorithms 1 and 2 are analyzed in Lemma 5 of Appendix A.3.

Algorithm 2: Iteration phase of SRWR- ITER

```
Input: positive semi-row-normalized matrix: $\tilde{\mathbf{A}}_+$, and negative semi-row-normalized matrix: $\tilde{\mathbf{A}}_-$, and
    seed node: $s$, restart probability: $c$, balance attenuation factors: $\beta$ and $\gamma$, and error tolerance: $\epsilon$.
Output: trustworthiness SRWR score vector: $\mathbf{r}$
set the starting vector $\mathbf{q}$ from the seed node $s$
set $\mathbf{r}^+ = \mathbf{q}$, $\mathbf{r}^- = \mathbf{0}$, and $\mathbf{h}' = [\mathbf{r}^+; \mathbf{r}^-]$
repeat
    $\mathbf{r}^+ \leftarrow (1-c)(\tilde{\mathbf{A}}_+^\top \mathbf{r}^+ + \beta \tilde{\mathbf{A}}_-^\top \mathbf{r}^- + (1-\gamma)\tilde{\mathbf{A}}_+^\top \mathbf{r}^-) + c\mathbf{q}$
    $\mathbf{r}^- \leftarrow (1-c)(\tilde{\mathbf{A}}_-^\top \mathbf{r}^+ + \gamma \tilde{\mathbf{A}}_+^\top \mathbf{r}^- + (1-\beta)\tilde{\mathbf{A}}_-^\top \mathbf{r}^-)$
    concatenate $\mathbf{r}^+$ and $\mathbf{r}^-$ into $\mathbf{h} = [\mathbf{r}^+; \mathbf{r}^-]^\top$
    compute the error between $\mathbf{h}$ and $\mathbf{h}'$, $\delta = \|\mathbf{h} - \mathbf{h}'\|$
    update $\mathbf{h}' \leftarrow \mathbf{h}$ for the next iteration
until $\delta < \epsilon$
compute $\mathbf{r} = \mathbf{r}^+ - \mathbf{r}^-$
return $\mathbf{r}$
```

4.2.1 Theoretical analysis of iterative algorithm and signed random walk with restart

We theoretically analyze the iterative algorithm SRWR- ITER and the properties of Signed Random Walk with Restart.

Convergence Analysis of SRWR-ITER We show that the iteration in Algorithm 2 converges to the solution of a linear system as described in the following theorem.

Theorem 1 (Convergence of SRWR- ITER) *Suppose* $\mathbf{h} = [\mathbf{r}^+; \mathbf{r}^-]^\top$ *and* $\mathbf{q}_s = [\mathbf{q}; \mathbf{0}]^\top$. *Then the iteration for* $\mathbf{h}$ *in Algorithm 2 converges to the solution* $\mathbf{h} = c(\mathbf{I} - (1-c)\tilde{\mathbf{B}}^\top)^{-1}\mathbf{q}_s$ *where* $\tilde{\mathbf{B}}^\top = \begin{bmatrix} \tilde{\mathbf{A}}_+^\top & \beta\tilde{\mathbf{A}}_-^\top + (1-\gamma)\tilde{\mathbf{A}}_+^\top \\ \tilde{\mathbf{A}}_-^\top & (1-\beta)\tilde{\mathbf{A}}_-^\top + \gamma\tilde{\mathbf{A}}_+^\top \end{bmatrix}$.

Proof Equation (3) is represented as follows:

$$\begin{bmatrix} \mathbf{r}^+ \\ \mathbf{r}^- \end{bmatrix} = (1-c)\begin{bmatrix} \tilde{\mathbf{A}}_+^\top & \beta\tilde{\mathbf{A}}_-^\top + (1-\gamma)\tilde{\mathbf{A}}_+^\top \\ \tilde{\mathbf{A}}_-^\top & (1-\beta)\tilde{\mathbf{A}}_-^\top + \gamma\tilde{\mathbf{A}}_+^\top \end{bmatrix}\begin{bmatrix} \mathbf{r}^+ \\ \mathbf{r}^- \end{bmatrix} + c\begin{bmatrix} \mathbf{q} \\ \mathbf{0} \end{bmatrix} \Leftrightarrow \mathbf{h} = (1-c)\tilde{\mathbf{B}}^\top\mathbf{h} + c\mathbf{q}_s$$

where $\tilde{\mathbf{B}}^\top = \begin{bmatrix} \tilde{\mathbf{A}}_+^\top & \beta\tilde{\mathbf{A}}_-^\top + (1-\gamma)\tilde{\mathbf{A}}_+^\top \\ \tilde{\mathbf{A}}_-^\top & (1-\beta)\tilde{\mathbf{A}}_-^\top + \gamma\tilde{\mathbf{A}}_+^\top \end{bmatrix}$, $\mathbf{h} = \begin{bmatrix} \mathbf{r}^+ \\ \mathbf{r}^- \end{bmatrix}$, and $\mathbf{q}_s = \begin{bmatrix} \mathbf{q} \\ \mathbf{0} \end{bmatrix}$. Thus, the iteration in Algorithm 2 is written as in the following equation:

$$\begin{aligned} \mathbf{h}^{(k)} &= (1-c)\tilde{\mathbf{B}}^\top\mathbf{h}^{(k-1)} + c\mathbf{q}_s \\ &= \left((1-c)\tilde{\mathbf{B}}^\top\right)^2\mathbf{h}^{(k-2)} + \left((1-c)\tilde{\mathbf{B}}^\top + \mathbf{I}\right)c\mathbf{q}_s \\ &= \cdots \\ &= \left((1-c)\tilde{\mathbf{B}}^\top\right)^k\mathbf{h}^{(0)} + \left(\sum_{j=0}^{k-1}\left((1-c)\tilde{\mathbf{B}}^\top\right)^j\right)c\mathbf{q}_s \end{aligned} \tag{4}$$

The spectral radius $\rho((1-c)\tilde{\mathbf{B}}^\top) = (1-c) < 1$ when $0 < c < 1$ since $\tilde{\mathbf{B}}^\top$ is a column stochastic matrix and its largest eigenvalue is 1 [36]. Therefore, $\lim_{k\to\infty}((1-c)\tilde{\mathbf{B}}^\top)^k\mathbf{h}^{(0)} = \mathbf{0}$ and $\lim_{k\to\infty}\mathbf{h}^{(k)}$ converges as follows:

$$\lim_{k\to\infty} \mathbf{h}^{(k)} = \mathbf{0} + \lim_{k\to\infty} \left(\sum_{j=0}^{k-1} \left((1-c)\tilde{\mathbf{B}}^\top \right)^j \right) c\mathbf{q}_s = c\left(\mathbf{I} - (1-c)\tilde{\mathbf{B}}^\top\right)^{-1} \mathbf{q}_s.$$

In the above equation, $\sum_{j=0}^{\infty}((1-c)\tilde{\mathbf{B}}^\top)^j$ is a geometric series of the matrix $(1-c)\tilde{\mathbf{B}}^\top$, and the series converges to $(\mathbf{I} - (1-c)\tilde{\mathbf{B}}^\top)^{-1}$ since the spectral radius of $(1-c)\tilde{\mathbf{B}}^\top$ is less than one. Note that the inverse matrix is a nonnegative matrix whose entries are positive or zero because the matrix is the sum of nonnegative matrices (i.e., $\sum_{j=0}^{\infty}((1-c)\tilde{\mathbf{B}}^\top)^j$). Hence, each entry of $\mathbf{h}$ is nonnegative (i.e., $\mathbf{h}_u \geq 0$ for any node u). □

Properties of SRWR We discuss the properties of our ranking model SRWR to answer the following questions: (1) Is the signed random surfer able to visit all nodes in a network which is strongly connected? and (2) Does SRWR work on unsigned networks as well? The first question is answered in Property 1, and the second one is answered in Property 2.

Property 1 *Suppose a signed network is strongly connected. Then, all entries of* $\mathbf{r}^+ + \mathbf{r}^-$ *are positive (i.e.,* $\mathbf{r}^+ + \mathbf{r}^- > 0$*).*

Proof Let $\mathbf{r}^+ + \mathbf{r}^-$ be $\mathbf{p}$. By summing the recursive equations on $\mathbf{r}^+$ and $\mathbf{r}^-$ in Eq. (3), $\mathbf{p}$ is represented as follows:

$$\mathbf{p} = (1-c)\left(\tilde{\mathbf{A}}_+^\top \mathbf{p} + \tilde{\mathbf{A}}_-^\top \mathbf{p}\right) + c\mathbf{q} \Leftrightarrow \mathbf{p} = (1-c)|\tilde{\mathbf{A}}|^\top \mathbf{p} + c\mathbf{q} \Leftrightarrow \mathbf{p} = \mathbf{G}\mathbf{p}$$

where $|\tilde{\mathbf{A}}| = \tilde{\mathbf{A}}_+ + \tilde{\mathbf{A}}_-$ by Definition 3, $\mathbf{G} = (1-c)|\tilde{\mathbf{A}}|^\top + c\mathbf{q}\mathbf{1}^\top$, and $\mathbf{1}^\top\mathbf{p} = \sum_i \mathbf{p}_i = 1$ by Property 3. Note that the graph represented by $\mathbf{G}$ is also strongly connected since the graph of $|\tilde{\mathbf{A}}|$ has the same topology with the original graph which is strongly connected. Moreover, the graph represented by $\mathbf{G}$ has a self-loop at the seed node s due to $c\mathbf{q}\mathbf{1}^\top$. Thus, $\mathbf{G}$ is irreducible and aperiodic. Hence, all entries of $\mathbf{p} = \mathbf{r}^+ + \mathbf{r}^-$ are positive according to Perron–Frobenius theorem [24]. □

Note that $\mathbf{r}_u^+$ (or $\mathbf{r}_u^-$) indicates that the stationary probability of the positive (or negative) surfer visits node u after performing SRWR starting from a seed node. According to Property 1, $\mathbf{r}_u^+ + \mathbf{r}_u^-$ for an arbitrary node u is always positive if a given signed network is strongly connected. That is, the signed random surfer is able to visit node u with probability $\mathbf{r}_u^+ + \mathbf{r}_u^-$ which is always greater than zero.

Next, we prove that our model SRWR is a generalized version of RWR working on both unsigned and signed networks in the following property.

Property 2 *The result of SRWR on networks containing only positive edges is the same as that of RWR.*

Proof $\tilde{\mathbf{A}}_+ = \tilde{\mathbf{A}}$ and $\tilde{\mathbf{A}}_- = \mathbf{0}_{n\times n}$ because the adjacency matrix $\mathbf{A}$ only contains positive edges. Also, $\mathbf{r}^- = \mathbf{0}_{n\times 1}$ at the beginning time of Algorithm 2. Equation (3) is represented as follows:

$$\mathbf{r}^+ = (1-c)\left(\tilde{\mathbf{A}}^\top \mathbf{r}^+ + \beta\mathbf{0}_{n\times n} \times \mathbf{0}_{n\times 1} + (1-\gamma)\tilde{\mathbf{A}}^\top \mathbf{0}_{n\times 1}\right) + c\mathbf{q}$$
$$\mathbf{r}^- = (1-c)\left(\mathbf{0}_{n\times n} \times \mathbf{r}^+ + \gamma\tilde{\mathbf{A}}^\top \mathbf{0}_{n\times 1} + (1-\beta)\mathbf{0}_{n\times n} \times \mathbf{0}_{n\times 1}\right)$$

Therefore, $\mathbf{r}^- = \mathbf{0}_{n\times 1}$ and $\mathbf{r}^+ = (1-c)\tilde{\mathbf{A}}^\top \mathbf{r}^+ + c\mathbf{q}$. The equation of $\mathbf{r}^+$ is exactly the same as that of RWR. □

4.3 SRWR-PRE: preprocessing algorithm for signed random walk with restart

We propose SRWR- PRE, a preprocessing algorithm to quickly compute SRWR scores. The iterative approach SRWR- ITER in Algorithm 2 requires multiple matrix-vector multiplications to compute SRWR scores whenever seed node s changes; thus, the iterative method is not fast enough when we require SRWR scores for any pair of nodes in large-scale signed networks. Our goal is to directly compute SRWR scores from precomputed intermediate data without iterations. We exploit the following ideas for our preprocessing method:

- The positive and negative SRWR score vectors $\mathbf{r}^{+}$ and $\mathbf{r}^{-}$ are obtained by solving linear systems (Sect. 4.3.1).
- The adjacency matrix of real-world graphs is permuted so that it contains a large but easy-to-invert block diagonal matrix as shown in Fig. 6 (Sect. 4.3.2).
- The block elimination approach efficiently solves a linear system on a matrix if it has an easy-to-invert submatrix (Sect. 4.3.3).

Our preprocessing method comprises two phases: preprocessing phase (Algorithm 3) and query phase (Algorithm 4). The preprocessing phase preprocesses a given signed adjacency matrix into several submatrices required in the query phase to compute SRWR scores w.r.t. seed node s. Note that the preprocessing phase is performed once, and the query phase is run for each seed node. The starting vector $\mathbf{q}$ in Eq. (3) is called an SRWR query, and $\mathbf{r}^{+}$ and $\mathbf{r}^{-}$ are the results corresponding to the query $\mathbf{q}$. The query vector $\mathbf{q}$ is determined by the seed node s, and $\mathbf{r}^{+}$ and $\mathbf{r}^{-}$ are distinct for each SRWR query. To exploit sparsity of graphs, we save all matrices in a sparse matrix format such as compressed column storage [7] which stores only nonzero entries and their locations.

4.3.1 Formulation of signed random walk with restart as linear systems

We first represent linear systems related to $\mathbf{r}^{+}$ and $\mathbf{r}^{-}$. Let $\mathbf{p}$ be the sum of $\mathbf{r}^{+}$ and $\mathbf{r}^{-}$ (i.e., $\mathbf{p} = \mathbf{r}^{+} + \mathbf{r}^{-}$). Then, $\mathbf{p}$ is the solution of the following linear system:

$$|\mathbf{H}|\mathbf{p} = c\mathbf{q} \Leftrightarrow \mathbf{p} = c|\mathbf{H}|^{-1}\mathbf{q} \tag{5}$$

where $|\mathbf{H}| = \mathbf{I} - (1-c)|\tilde{\mathbf{A}}|^{\top}$ and $|\tilde{\mathbf{A}}| = \tilde{\mathbf{A}}_{+} + \tilde{\mathbf{A}}_{-}$. The proof of Eq. (5) is presented in Lemma 1. The linear system for $\mathbf{r}^{-}$ is given by the following equation:

$$\mathbf{T}\mathbf{r}^{-} = (1-c)\tilde{\mathbf{A}}_{-}^{\top}\mathbf{p} \Leftrightarrow \mathbf{r}^{-} = (1-c)\left(\mathbf{T}^{-1}(\tilde{\mathbf{A}}_{-}^{\top}\mathbf{p})\right) \tag{6}$$

where $\mathbf{T} = \mathbf{I} - (1-c)(\gamma\tilde{\mathbf{A}}_{+}^{\top} - \beta\tilde{\mathbf{A}}_{-}^{\top})$, and γ and β are balance attenuation factors. Theorem 2 shows the proof of Eq. (6). Based on the aforementioned linear systems in Eqs. (5) and (6), $\mathbf{r}^{-}$ and $\mathbf{r}^{+}$ for a given seed node s are computed as follows:

1. Set a query vector $\mathbf{q}$ whose sth element is 1 and all other elements are 0.
2. Solve the linear system in Eq. (5) to obtain the solution $\mathbf{p}$.
3. Compute $\mathbf{r}^{-}$ by solving the linear system in Eq. (6).
4. Compute $\mathbf{r}^{+} = \mathbf{p} - \mathbf{r}^{-}$.

Lemma 1 *Suppose that* $\mathbf{p} = \mathbf{r}^{+} + \mathbf{r}^{-}$, $|\mathbf{H}| = \mathbf{I} - (1-c)|\tilde{\mathbf{A}}|^{\top}$ *and* $|\tilde{\mathbf{A}}| = \tilde{\mathbf{A}}_{+} + \tilde{\mathbf{A}}_{-}$. *Then,* $\mathbf{p}$ *is the solution of the following linear system:*

$$|\mathbf{H}|\mathbf{p} = c\mathbf{q} \Leftrightarrow \mathbf{p} = c|\mathbf{H}|^{-1}\mathbf{q}$$

Proof According to the result in Property 1, the recursive equation for $\mathbf{p}$ is represented as follows:

$$\mathbf{p} = (1-c)|\tilde{\mathbf{A}}|^{\top}\mathbf{p} + c\mathbf{q}$$

where $|\tilde{\mathbf{A}}| = \tilde{\mathbf{A}}_{+} + \tilde{\mathbf{A}}_{-}$ is the row-normalized matrix of $|\mathbf{A}|$. The linear system for $\mathbf{p}$ is represented by moving $(1-c)|\tilde{\mathbf{A}}|^{\top}\mathbf{p}$ to the left side as follows:

$$\left(\mathbf{I} - (1-c)|\tilde{\mathbf{A}}|^{\top}\right)\mathbf{p} = c\mathbf{q} \Leftrightarrow |\mathbf{H}|\mathbf{p} = c\mathbf{q}$$

where $|\mathbf{H}|$ is $\mathbf{I} - (1-c)|\tilde{\mathbf{A}}|^{\top}$. Note that $|\mathbf{H}|$ is invertible when $0 < c < 1$ because it is strictly diagonally dominant [11]. Hence, $\mathbf{p} = c|\mathbf{H}|^{-1}\mathbf{q}$. □

Theorem 2 *The SRWR score vectors* $\mathbf{r}^{+}$ *and* $\mathbf{r}^{-}$ *from Eq. (3) are represented as follows:*

$$\mathbf{r}^{+} = \mathbf{p} - \mathbf{r}^{-}$$
$$\mathbf{r}^{-} = (1-c)\left(\mathbf{T}^{-1}(\tilde{\mathbf{A}}_{-}^{\top}\mathbf{p})\right)$$

where $\mathbf{p} = c|\mathbf{H}|^{-1}\mathbf{q}$, $\mathbf{T} = \mathbf{I} - (1-c)(\gamma\tilde{\mathbf{A}}_{+}^{\top} - \beta\tilde{\mathbf{A}}_{-}^{\top})$, *and* γ *and* β *are balance attenuation factors which are between* 0 *and* 1 *(i.e.,* $0 < \gamma, \beta < 1$*).*

Proof Note that $\mathbf{r}^{-} = (1-c)(\tilde{\mathbf{A}}_{-}^{\top}\mathbf{r}^{+} + \gamma\tilde{\mathbf{A}}_{+}^{\top}\mathbf{r}^{-} + (1-\beta)\tilde{\mathbf{A}}_{-}^{\top}\mathbf{r}^{-})$ by Eq. (3), and $\mathbf{r}^{+} = \mathbf{p} - \mathbf{r}^{-}$ according to Lemma 1. The equation for $\mathbf{r}^{-}$ is represented by plugging $\mathbf{r}^{+} = \mathbf{p} - \mathbf{r}^{-}$ as follows:

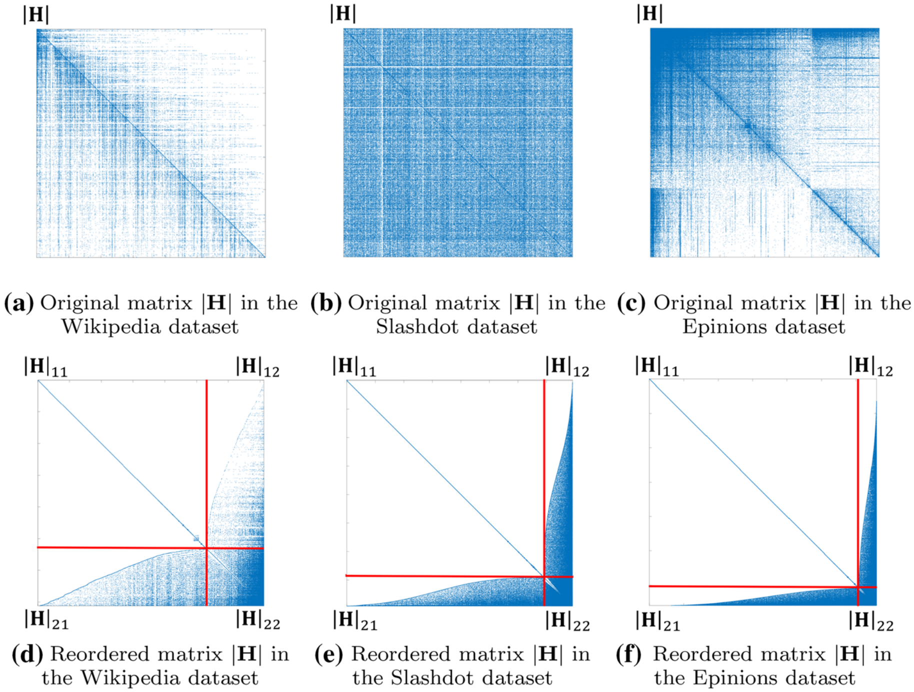
$$\mathbf{r}^{-} = (1-c)\left(\tilde{\mathbf{A}}_{-}^{\top}\mathbf{p} - \tilde{\mathbf{A}}_{-}^{\top}\mathbf{r}^{-} + \gamma\tilde{\mathbf{A}}_{+}^{\top}\mathbf{r}^{-} + (1-\beta)\tilde{\mathbf{A}}_{-}^{\top}\mathbf{r}^{-}\right) \Leftrightarrow$$
$$\mathbf{r}^{-} = (1-c)\left(\gamma\tilde{\mathbf{A}}_{+}^{\top} - \beta\tilde{\mathbf{A}}_{-}^{\top}\right)\mathbf{r}^{-} + (1-c)\tilde{\mathbf{A}}_{-}^{\top}\mathbf{p}$$

We move $(1-c)(\gamma\tilde{\mathbf{A}}_{+}^{\top} - \beta\tilde{\mathbf{A}}_{-}^{\top})\mathbf{r}^{-}$ to the left side; then, the above equation is represented as follows:

$$\left(\mathbf{I} - (1-c)(\gamma\tilde{\mathbf{A}}_{+}^{\top} - \beta\tilde{\mathbf{A}}_{-}^{\top})\right)\mathbf{r}^{-} = (1-c)\tilde{\mathbf{A}}_{-}^{\top}\mathbf{p} \Leftrightarrow \mathbf{T}\mathbf{r}^{-} = (1-c)\tilde{\mathbf{A}}_{-}^{\top}\mathbf{p}$$

where $\mathbf{T}$ is $\mathbf{I} - (1-c)(\gamma\tilde{\mathbf{A}}_{+}^{\top} - \beta\tilde{\mathbf{A}}_{-}^{\top})$. Note that the matrix $\mathbf{T}$ is strictly diagonally dominant when $0 < c < 1$ and $0 < \gamma, \beta < 1$; thus, $\mathbf{T}$ is invertible. Hence, $\mathbf{r}^{-} = (1-c)(\mathbf{T}^{-1}(\tilde{\mathbf{A}}_{-}^{\top}\mathbf{p}))$. $\mathbf{r}^{+}$ is obtained by computing $\mathbf{r}^{+} = \mathbf{p} - \mathbf{r}^{-}$. □

One naive approach (Inversion) for SRWR score vectors $\mathbf{r}^{+}$ and $\mathbf{r}^{-}$ based on the linear systems in Eqs. (5) and (6) is to precompute the inverse of the matrices $|\mathbf{H}|$ and $\mathbf{T}$. However, this approach is impractical for large-scale graphs since inverting a matrix requires $O(n^3)$ time and $O(n^2)$ space where n is the dimensions of the matrix. Another approach (LU) is to obtain the inverse of LU factors of $|\mathbf{H}|$ and $\mathbf{T}$ after reordering the matrices in the order of node degrees as suggested in [9] (i.e., $\mathbf{p} = c(\mathbf{U}_{\mathbf{p}}^{-1}(\mathbf{L}_{\mathbf{p}}^{-1}\mathbf{q}))$; $\mathbf{r}^{-} = (1-c)(\mathbf{U}_{\mathbf{r}^{-}}^{-1}(\mathbf{L}_{\mathbf{r}^{-}}^{-1}(\tilde{\mathbf{A}}_{-}^{\top}\mathbf{p})))$ where $|\mathbf{H}|^{-1} = \mathbf{U}_{\mathbf{p}}^{-1}\mathbf{L}_{\mathbf{p}}^{-1}$ and $\mathbf{T}^{-1} = \mathbf{U}_{\mathbf{r}^{-}}^{-1}\mathbf{L}_{\mathbf{r}^{-}}^{-1}$). Although LU is more efficient than Inversion in terms of time and space as shown in Fig. 13, LU still has a performance issue due to $O(n^3)$ time and $O(n^2)$ space complexities. On the other hand, our preprocessing method SRWR- Pre is faster and more memory efficient than Inversion and LU as we will see in Sect. 5.7.



(a) Original matrix |H| in the Wikipedia dataset
(b) Original matrix |H| in the Slashdot dataset
(c) Original matrix |H| in the Epinions dataset
(d) Reordered matrix |H| in the Wikipedia dataset
(e) Reordered matrix |H| in the Slashdot dataset
(f) Reordered matrix |H| in the Epinions dataset

Fig. 6 The result of node reordering on each signed network. **a**–**c** Are the original matrix $|\mathbf{H}|$ before node reordering in the Wikipedia, the Slashdot, and the Epinions datasets, respectively. **d**–**f** Present $|\mathbf{H}|$ reordered by the hub-and-spoke method. Note that $\mathbf{T}$ is also reordered equivalently to $|\mathbf{H}|$ since they have the same sparsity pattern. $|\mathbf{H}|_{11}$ and $\mathbf{T}_{11}$ are block diagonal

4.3.2 Node reordering based on hub-and-spoke structure

SRWR- Pre permutes the matrices $|\mathbf{H}|$ and $\mathbf{T}$ using a reordering technique based on hub-and-spoke structure. Previous works [34] have exploited the reordering technique to reduce computational cost of graph operations in real-world graphs. We also adopt the node reordering based on hub-and-spoke structure to efficiently solve the linear systems in Eqs. (5) and (6).

The hub-and-spoke structure indicates that most real-world graphs follow power-law degree distribution with few hubs (very high-degree nodes) and majority of spokes (low-degree nodes). The structure has been utilized to concentrate entries of an adjacency matrix by reordering nodes as shown in Fig. 6. Any reordering method based on the hub-and-spoke structure can be utilized for the purpose; in this paper, we use SlashBurn [19,28] as a hub-and-spoke reordering method because it shows the best performance in concentrating entries of an adjacency matrix (see the details in Appendix A.1).

We reorder nodes of the signed adjacency matrix $\mathbf{A}$ so that reordered matrix contains a large but easy-to-invert submatrix such as block diagonal matrix as shown in Fig. 6. We then compute $|\mathbf{H}| = \mathbf{I} - (1-c)(\tilde{\mathbf{A}}_{+}^{\top} + \tilde{\mathbf{A}}_{-}^{\top})$ and $\mathbf{T} = \mathbf{I} - (1-c)(\gamma\tilde{\mathbf{A}}_{+}^{\top} - \beta\tilde{\mathbf{A}}_{-}^{\top})$. Note that $|\mathbf{H}|$ and $\mathbf{T}$ have the same sparsity pattern as the reordered adjacency matrix $\mathbf{A}^{\top}$ except for the diagonal part. Hence, $|\mathbf{H}|$ and $\mathbf{T}$ are partitioned as follows:

$$|\mathbf{H}| = \begin{bmatrix} |\mathbf{H}|_{11} & |\mathbf{H}|_{12} \\ |\mathbf{H}|_{21} & |\mathbf{H}|_{22} \end{bmatrix}, \mathbf{T} = \begin{bmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} \\ \mathbf{T}_{21} & \mathbf{T}_{22} \end{bmatrix}. \tag{7}$$

Let n_1 and n_2 denote the number of spokes and hubs, respectively (see the details in Appendix A.1). Then $|\mathbf{H}|_{11}$ and $\mathbf{T}_{11}$ are $n_1 \times n_1$ matrices, $|\mathbf{H}|_{12}$ and $\mathbf{T}_{12}$ are $n_1 \times n_2$ matrices, $|\mathbf{H}|_{21}$ and $\mathbf{T}_{21}$ are $n_2 \times n_1$ matrices, and $|\mathbf{H}|_{22}$ and $\mathbf{T}_{22}$ are $n_2 \times n_2$ matrices. The linear systems for $|\mathbf{H}|$ and $\mathbf{T}$ in Eqs. (5) and (6) are represented as follows:

$$|\mathbf{H}|\mathbf{p} = c\mathbf{q} \Leftrightarrow \begin{bmatrix} |\mathbf{H}|_{11} & |\mathbf{H}|_{12} \\ |\mathbf{H}|_{21} & |\mathbf{H}|_{22} \end{bmatrix} \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \end{bmatrix} = c \begin{bmatrix} \mathbf{q}_1 \\ \mathbf{q}_2 \end{bmatrix} \tag{8}$$

$$\mathbf{T}\mathbf{r}^- = (1-c)\mathbf{t} \Leftrightarrow \begin{bmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} \\ \mathbf{T}_{21} & \mathbf{T}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{r}_1^- \\ \mathbf{r}_2^- \end{bmatrix} = (1-c) \begin{bmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \end{bmatrix} \tag{9}$$

where $\mathbf{t} = \tilde{\mathbf{A}}_-^\top \mathbf{p}$ is an $n \times 1$ vector.

4.3.3 Block elimination for solving linear systems

The solutions of the partitioned linear systems in Eqs. (8) and (9) are obtained by the following equations:

$$\mathbf{p} = \begin{bmatrix} \mathbf{p}_1 \\ \mathbf{p}_2 \end{bmatrix} = \begin{bmatrix} |\mathbf{H}|_{11}^{-1}(c\mathbf{q}_1 - |\mathbf{H}|_{12}\mathbf{p}_2) \\ c(\mathbf{S}_{|\mathbf{H}|}^{-1}(\mathbf{q}_2 - |\mathbf{H}|_{21}(|\mathbf{H}|_{11}^{-1}(\mathbf{q}_1)))) \end{bmatrix} \tag{10}$$

$$\mathbf{r}^- = \begin{bmatrix} \mathbf{r}_1^- \\ \mathbf{r}_2^- \end{bmatrix} = \begin{bmatrix} \mathbf{T}_{11}^{-1}((1-c)\mathbf{t}_1 - \mathbf{T}_{12}\mathbf{r}_2^-) \\ (1-c)(\mathbf{S}_{\mathbf{T}}^{-1}(\mathbf{t}_2 - \mathbf{T}_{21}(\mathbf{T}_{11}^{-1}(\mathbf{t}_1)))) \end{bmatrix} \tag{11}$$

where $\mathbf{S}_{|\mathbf{H}|} = |\mathbf{H}|_{22} - |\mathbf{H}|_{21}|\mathbf{H}|_{11}^{-1}|\mathbf{H}|_{12}$ is the Schur complement of $|\mathbf{H}|_{11}$ and $\mathbf{S}_{\mathbf{T}} = \mathbf{T}_{22} - \mathbf{T}_{21}\mathbf{T}_{11}^{-1}\mathbf{T}_{12}$ is the Schur complement of $\mathbf{T}_{11}$. Equations (10) and (11) are derived by applying block elimination described in Lemma 2 to the partitioned linear systems in Eqs. (8) and (9), respectively. Note that the submatrices $|\mathbf{H}|_{11}$ and $\mathbf{T}_{11}$ are invertible when $0 < c < 1$ and $0 < \gamma, \beta < 1$ since they are strictly diagonally dominant. If all matrices in Eqs. (10) and (11) are precomputed, then the SRWR score vectors $\mathbf{r}^+$ and $\mathbf{r}^-$ are efficiently and directly computed from the precomputed matrices.

Lemma 2 (Block Elimination [4]) *Suppose a linear system* $\mathbf{A}\mathbf{x} = \mathbf{b}$ *is partitioned as follows:*

$$\begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{bmatrix}$$

where $\mathbf{A}_{11}$ *and* $\mathbf{A}_{22}$ *are square matrices. If the submatrix* $\mathbf{A}_{11}$ *is invertible, then the solution* $\mathbf{x}$ *is represented as follows:*

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{11}^{-1}(\mathbf{b}_1 - \mathbf{A}_{12}\mathbf{x}_2) \\ \mathbf{S}^{-1}(\mathbf{b}_2 - \mathbf{A}_{21}(\mathbf{A}_{11}^{-1}(\mathbf{b}_1))) \end{bmatrix}$$

where $\mathbf{S} = \mathbf{A}_{22} - \mathbf{A}_{21}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}$ *is the Schur complement of* $\mathbf{A}_{11}$. □

Lemma 2 implies that a partitioned linear system is efficiently solved if it contains an easy-to-invert submatrix and the dimension of the Schur complement is small. Note that inverting $\mathbf{H}_{11}$ and $\mathbf{T}_{11}$ is trivial because they are block diagonal matrices as shown in Fig. 6. Also, the dimension of $\mathbf{S}_{|\mathbf{H}|}$ and $\mathbf{S}_{\mathbf{T}}$ is n_2 where n_2 is the number of hubs and most real-world graphs have a small number of hubs compared to the number of nodes (see Table 2).

Algorithm 3: Preprocessing phase of SRWR- Pre

```
Input: signed adjacency matrix: $\mathbf{A}$, restart probability: $c$, balance attenuation factors:
    $\beta$ and $\gamma$
Output: preprocessed matrices from $|\mathbf{H}|$ and $\mathbf{T}$, negative semi-row-normalized matrix $\tilde{\mathbf{A}}_-$
 1: reorder $\mathbf{A}$ using the hub-and-spoke reordering method [19,28]
 2: compute $\tilde{\mathbf{A}}_+$ and $\tilde{\mathbf{A}}_-$ from $\mathbf{A}$ using Algorithm 1
 3: compute $|\mathbf{H}|$ and $\mathbf{T}$, i.e., $|\mathbf{H}| = \mathbf{I} - (1-c)|\tilde{\mathbf{A}}|^\top$ and $\mathbf{T} = \mathbf{I} - (1-c)(\gamma\tilde{\mathbf{A}}_+^\top - \beta\tilde{\mathbf{A}}_-^\top)$
 4: partition $|\mathbf{H}|$ into $|\mathbf{H}|_{11}$, $|\mathbf{H}|_{12}$, $|\mathbf{H}|_{21}$, $|\mathbf{H}|_{22}$, and compute $|\mathbf{H}|_{11}^{-1}$
 5: partition $\mathbf{T}$ into $\mathbf{T}_{11}$, $\mathbf{T}_{12}$, $\mathbf{T}_{21}$, $\mathbf{T}_{22}$, and compute $\mathbf{T}_{11}^{-1}$
 6: compute the Schur complement of $|\mathbf{H}|_{11}$, i.e., $\mathbf{S}_{|\mathbf{H}|} = |\mathbf{H}|_{22} - |\mathbf{H}|_{21}|\mathbf{H}|_{11}^{-1}|\mathbf{H}|_{12}$
 7: compute the Schur complement of $\mathbf{T}_{11}$, i.e., $\mathbf{S}_\mathbf{T} = \mathbf{T}_{22} - \mathbf{T}_{21}\mathbf{T}_{11}^{-1}\mathbf{T}_{12}$
 8: compute the inverse of LU factors of $\mathbf{S}_{|\mathbf{H}|}$, i.e., $\mathbf{S}_{|\mathbf{H}|}^{-1} = \mathbf{U}_{|\mathbf{H}|}^{-1}\mathbf{L}_{|\mathbf{H}|}^{-1}$
 9: compute the inverse of LU factors of $\mathbf{S}_\mathbf{T}$, i.e., $\mathbf{S}_\mathbf{T}^{-1} = \mathbf{U}_\mathbf{T}^{-1}\mathbf{L}_\mathbf{T}^{-1}$
10: return preprocessed matrices from $|\mathbf{H}|$: $\mathbf{L}_{|\mathbf{H}|}^{-1}$, $\mathbf{U}_{|\mathbf{H}|}^{-1}$, $|\mathbf{H}|_{11}^{-1}$, $|\mathbf{H}|_{12}$, and $|\mathbf{H}|_{21}$
           preprocessed matrices from $\mathbf{T}$: $\mathbf{L}_\mathbf{T}^{-1}$, $\mathbf{U}_\mathbf{T}^{-1}$, $\mathbf{T}_{11}^{-1}$, $\mathbf{T}_{12}$, and $\mathbf{T}_{21}$
           negative semi-row-normalized matrix $\tilde{\mathbf{A}}_-$
```

Algorithm 4: Query phase of SRWR- Pre

```
Input: seed node: $s$, preprocessed matrices from Algorithm 3
Output: trustworthiness SRWR score vector: $\mathbf{r}$
 1: create $\mathbf{q}$ whose $s$th entry is 1 and the others are 0, and partition $\mathbf{q}$ into $\mathbf{q}_1$ and $\mathbf{q}_2$
 2: compute $\mathbf{p}_2 = c(\mathbf{U}_{|\mathbf{H}|}^{-1}(\mathbf{L}_{|\mathbf{H}|}^{-1}(\mathbf{q}_2 - |\mathbf{H}|_{21}(|\mathbf{H}|_{11}^{-1}\mathbf{q}_1))))$
 3: compute $\mathbf{p}_1 = |\mathbf{H}|_{11}^{-1}(c\mathbf{q}_1 - |\mathbf{H}|_{12}\mathbf{p}_2)$
 4: create $\mathbf{p}$ by concatenating $\mathbf{p}_1$ and $\mathbf{p}_2$
 5: compute $\mathbf{t} = \tilde{\mathbf{A}}_-^\top\mathbf{p}$, and partition it into $\mathbf{t}_1$ and $\mathbf{t}_2$
 6: compute $\mathbf{r}_2^- = (1-c)(\mathbf{U}_\mathbf{T}^{-1}(\mathbf{L}_\mathbf{T}^{-1}(\mathbf{t}_2 - \mathbf{T}_{21}(\mathbf{T}_{11}^{-1}\mathbf{t}_1))))$
 7: compute $\mathbf{r}_1^- = \mathbf{T}_{11}^{-1}((1-c)\mathbf{t}_1 - \mathbf{T}_{12}\mathbf{r}_2^-))$
 8: create $\mathbf{r}^-$ by concatenating $\mathbf{r}_1^-$ and $\mathbf{r}_2^-$
 9: compute $\mathbf{r}^+ = \mathbf{p} - \mathbf{r}^-$
10: compute $\mathbf{r} = \mathbf{r}^+ - \mathbf{r}^-$
11: return $\mathbf{r}$
```

Preprocessing phase (Algorithm 3) Our preprocessing phase precomputes the matrices exploited for computing SRWR scores in the query phase. Our algorithm first reorders nodes of a given signed adjacency matrix $\mathbf{A}$ using the hub-and-spoke reordering method, and performs semi-normalization on $\mathbf{A}$ to obtain $\tilde{\mathbf{A}}_+$ and $\tilde{\mathbf{A}}_-$ using Algorithm 1 (lines 1–refalg:method:pre:normalize). Then our algorithm computes $|\mathbf{H}|$ and $\mathbf{T}$ and partitions the matrices as shown in Fig. 6 (lines 3–5). Our algorithm calculates the inverses of $|\mathbf{H}|_{11}$ and $\mathbf{T}_{11}$, and computes the Schur complements of $|\mathbf{H}|_{11}$ and $\mathbf{T}_{11}$ (lines 4–7). When we compute $\mathbf{S}_{|\mathbf{H}|}^{-1}$ and $\mathbf{S}_\mathbf{T}^{-1}$, we invert the LU factors of $\mathbf{S}_{|\mathbf{H}|}$ and $\mathbf{S}_\mathbf{T}$ (lines 8 and 9) because this approach is faster and more memory efficient than directly inverting $\mathbf{S}_{|\mathbf{H}|}$ and $\mathbf{S}_\mathbf{T}$ as in [34].

Query phase (Algorithm 4) Our query phase computes SRWR score vectors $\mathbf{r}^+$ and $\mathbf{r}^-$ for a given seed node s using precomputed matrices from Algorithm 3. Our algorithm first creates a starting vector $\mathbf{q}$ whose entry at the index of the seed node s is 1 and otherwise 0, and partitions $\mathbf{q}$ into $\mathbf{q}_1$ and $\mathbf{q}_2$ (line 1). We then compute $\mathbf{p}_2$ and $\mathbf{p}_1$ based on Eq. (10) and concatenate the vectors to obtain $\mathbf{p}$ (lines 2∼4). Our algorithm calculates $\mathbf{t} = \tilde{\mathbf{A}}_-^\top\mathbf{p}$ and partitions $\mathbf{t}$ into $\mathbf{t}_1$ and $\mathbf{t}_2$ (line 5). We compute $\mathbf{r}_2^-$ and $\mathbf{r}_1^-$ based on Eq. (11), and concatenate

Table 2 Dataset statistics

Dataset	n	m	m_+	m_-	n_2
Wikipedia[1]	7118	103, 617	81, 285	22, 332	1800
Slashdot[2]	79, 120	515, 561	392, 316	123, 245	10, 160
Epinions[3]	131, 828	841, 372	717, 667	123, 705	10, 164

n is the number of nodes, and m is the total number of edges. m_+ is the number of positive edges, m_- is the number of negative edges, and n_2 is the number of hubs

[1] http://snap.stanford.edu/data/wiki-Vote.html

[2] http://dai-labor.de/IRML/datasets

[3] http://www.trustlet.org/wiki/Extended_Epinions_dataset

the vectors to obtain $\mathbf{r}^-$ (lines 6–8). After computing $\mathbf{r}^+ = \mathbf{p} - \mathbf{r}^-$ to obtain $\mathbf{r}^+$ (line 9), we obtain $\mathbf{r} = \mathbf{r}^+ - \mathbf{r}^-$ (line 10).

The space and time complexities of Algorithms 3 and 4 are analyzed in Lemmas 6–8 of Appendix A.3.

5 Experiments

We evaluate the effectiveness of SRWR compared to existing ranking methods. Since there is no ground-truth of personalized rankings for each node in real-world graphs, we exploit an indirect way by examining the performance of applications such as link prediction, troll identification, and sign prediction tasks. We also investigate the performance of our approaches in terms of time and space. Based on these settings, we aim to answer the following questions from the experiments:

- **Q1. Link prediction (Sect.** 5.2) How effective is our proposed SRWR model for the link prediction task in signed networks?
- **Q2. User preference preservation (Sect.** 5.3) How well does our model SRWR preserve users' known preferences in personalized rankings in signed networks?
- **Q3. Troll detection (Sect.** 5.4) How well do personalized rankings of SRWR capture trolls who are abnormal users compared to those of other models?
- **Q4. Sign prediction (Sect.** 5.5) How helpful are trustworthiness scores of SRWR for predicting missing signs of edges in signed networks?
- **Q5. Effects of balance attenuation factors (Sect.** 5.6) How effective are the balance attenuation factors of SRWR for applications in signed networks?
- **Q6. Efficiency (Sect.** 5.7) How fast and memory efficient is our preprocessing method SRWR- PRE compared to other baselines?

5.1 Experimental settings

Machines The experiments on the effectiveness of SRWR in Sects. 5.2, 5.4, 5.5 and 5.6 are conducted on a PC with Intel(R) Core(TM) i5-4590 CPU @ 3.30GHz and 8GB memory. The experiments on the computational performance of SRWR- PRE in Sect. 5.7 are performed on a workstation with a single Intel(R) Xeon(R) CPU E5-2630 v4 @ 2.20GHz and 256GB memory.

Datasets The signed networks for our experiments are summarized in Table 2. We use all datasets in the link prediction task, the sign prediction task and the experiments for evaluating the computational performance of the proposed methods (Sects. 5.2, 5.5, and 5.7). We use the Slashdot dataset in the troll identification task (Sect. 5.4) since there is a troll list only in the dataset.

Methods To answer **Q1–4**, we compare our proposed model with Random Walk with Restart (RWR) [13], Modified Random Walk with Restart (M-RWR) [33], Modified Personalized SALSA (M-PSALSA) [30], Personalized Signed spectral Rank (PSR) [23], Personalized Negative Rank (PNR) [23], Troll-Trust Model (TR-TR) [41], TRUST [12], LOGIT [26], and GAUC-OPT [35]. Note that RWR is computed on the absolute adjacency matrix of a signed network. For **Q5**, we compare our model SRWR to H- SRWR which is a version of SRWR without the balance attenuation parameters. For **Q6**, we compare our preprocessing method SRWR- PRE to other baseline methods Inversion and LU mentioned in Sect. 4.3.1 including our iterative method SRWR- ITER.

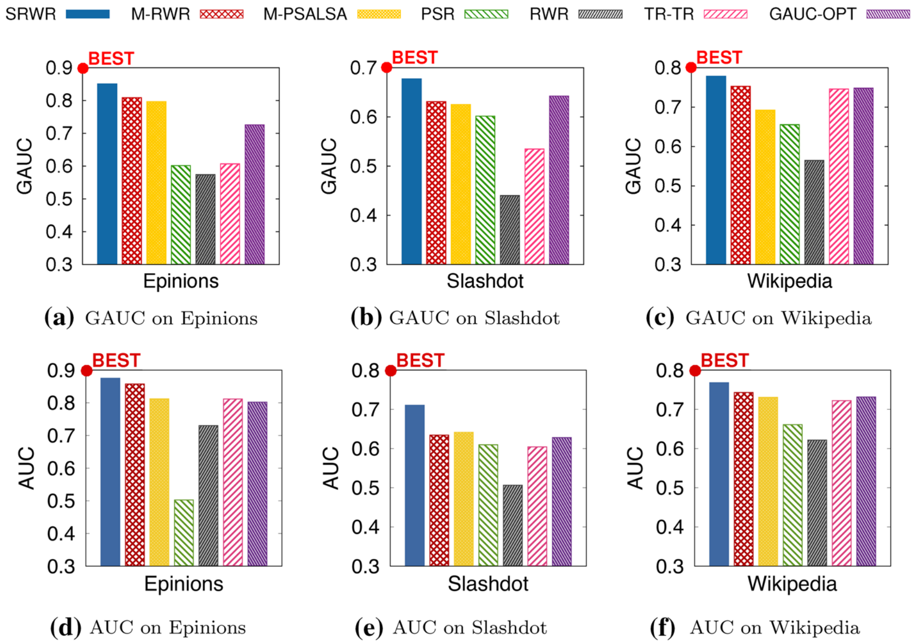
Parameters There are three hyper-parameters in our ranking model, i.e., restart probability c and balance attenuation parameters β and γ. We set c to 0.15 for all random walk-based approaches including our model for simplicity. To choose β and γ, we perform a grid search over a range $0 \leq \beta, \gamma \leq 1$ by 0.1 (i.e., search (β, γ) in $\mathbf{P} = \{(0.1x, 0.1y) | 0 \leq x, y \leq 10 \text{ and } x, y \in \mathbb{Z}\}$). To select proper parameters, we randomly split a dataset into training, validation, and test sets; and then, we compute personalized rankings based on the training set, and choose the best parameter combination (β, γ) on the validation set with a target metric corresponding to each task. We report results on the test set with the validated parameters. The detailed settings on how to split the dataset and which metric is used are described in each subsection of the corresponding task. The validated parameters of SRWR are summarized as follows:

- Link prediction task (Sect. 5.2): In the Epinions and the Slashdot datasets, $\beta = 0.5$ and $\gamma = 0.8$. In the Wikipedia dataset, $\beta = 0.5$ and $\gamma = 0.5$.
- Troll identification task (Sect. 5.4): In this task, the Slashdot dataset is used as mentioned above, and in the dataset, $\beta = 0.1$ and $\gamma = 1.0$.
- Sign prediction task (Sect. 5.5): In the Epinions and the Slashdot datasets, $\beta = 0.5$ and $\gamma = 0.8$. In the Wikipedia dataset, $\beta = 0.2$ and $\gamma = 0.6$.

5.2 Link prediction task

We evaluate the performance of personalized ranking models on link prediction in signed networks. The link prediction task is defined as follows: given a signed network and a seed node s, predict nodes which will be positively or negatively linked by the seed node in the future. An ideal personalized ranking for this task should place nodes that the seed node s potentially trusts (i.e., positive links) at the top, those that s potentially distrusts (i.e., negative links) at bottom, and other unknown ones in the middle. GAUC (Generalized AUC), proposed by [35], has been used to evaluate the quality of personalized rankings for link prediction in signed networks, and it measures such ideal ranking as 1.0. We also evaluate the ranking quality in terms of AUC indicating how many positive nodes are ranked higher than negative ones (see the details in Appendix A.5.1).

To perform this evaluation, we randomly select 1000 seed nodes, and choose 20% edges of positive and negative edges from each seed node to form a validation set. Then, we randomly select another 1000 seed nodes and choose 20% edges of positive and negative edges from each seed node as a test set. We remove those selected edges and utilize the



(a) GAUC on Epinions (b) GAUC on Slashdot (c) GAUC on Wikipedia

(d) AUC on Epinions (e) AUC on Slashdot (f) AUC on Wikipedia

Fig. 7 The link prediction performance of ranking models in terms of GAUC and AUC. GAUC indicates how well a model ranks nodes to be positively connected by a seed node at the top and those to be negatively linked at the bottom. AUC indicates how many positive nodes are ranked higher than negative ones (see the details in Appendix A.5.1). Our proposed model SRWR shows the best link prediction performance in terms of GAUC and AUC for all the datasets

remaining edges as a training set to compute personalized rankings. For given a parameter combination, we measure GAUC on the personalized ranking w.r.t. each seed node in the validation set, and record the average GAUC over all the seed nodes. Then, we pick the best parameter combination that provides the highest average GAUC in the validation set. With the validated parameters, we report the average GAUC over all seed nodes in the test set. We perform the same procedure for AUC. For nodes directly connected with a seed node s in the training set, we exclude those nodes from a personalized ranking list w.r.t. s since we need to recommend links which are unknown to s.

Results We compare SRWR to other random walk-based models M-RWR, M-PSALSA, PSR, RWR, and TR-TR on the link prediction task in signed networks. We also compare our method to GAUC-OPT which is a matrix factorization-based link prediction method approximately maximizing GAUC [35]. As demonstrated in Fig. 7, SRWR presents the best link prediction performance in terms of GAUC and AUC among the evaluated models over all the datasets. Compared to RWR which does not consider negative signs at all, our approach SRWR shows the significant improvement in the link prediction accuracy. Particularly, GAUC of all other methods considering signed edges is higher than that of RWR as shown in Fig. 7. This indicates that it is important to consider the sign of edges when we compute personalized rankings for link prediction in signed networks. Furthermore, SRWR outperforms other random walk-based models including GAUC-OPT which is specially designed for this task, implying our signed surfer based on balance theory effectively estimates personalized rankings for link prediction in signed networks.

Table 3 The user preference preservation quality of ranking models in terms of GAUC (Appendix A.5.1)

Datasets (GAUC)	SRWR (prop.)	M-RWR	M-PSALSA	PSR	RWR	TR-TR	GAUC-OPT
Epinions	**1.000**	0.999	0.902	0.730	0.708	0.650	0.824
Slashdot	**1.000**	0.982	0.800	0.728	0.705	0.625	0.708
Wikipedia	**0.999**	0.944	0.934	0.707	0.702	0.778	0.742

Note that 1.0 GAUC indicates that a method perfectly preserves a user's known preferences in its personalized ranking. Our proposed model SRWR shows the best performance in user preference preservation among all tested methods

5.3 User preference preservation task

Since a personalized ranking includes known and unknown users for a seed user (or node), how the ranking is consistent with the seed user's known preferences is also considered as one criterion for evaluating the quality of personalized rankings. In signed social networks, we consider that the known preferences of a seed user s are well preserved in a personalized ranking if positive users for s (i.e., they are positively connected by s) are at the top and negative ones are at the bottom in the ranking. Hence, an ideal ranking for s (excluding s from the ranking) should produce 1.0 GAUC with known positive and negative links from s in terms of user preference preservation. To evaluate the preference preservation performance of each method, we report the average GAUC over all test seed nodes without removing the selected test edges from a training set.

As shown in Table 3, our ranking model SRWR demonstrates the best GAUC in user preference preservation among all tested methods, indicating that SRWR almost perfectly preserves users' known preferences within their personalized rankings. The main reason for the result is that our signed surfer occasionally restarts at a seed node s with a positive sign; thus, the positive surfer frequently visits the positive neighbors of s, and the negative surfer frequently visits the negative neighbors of s. Hence, the trustworthiness scores on the positive neighbors are high, and those on the negative neighbors are low compared to those on nodes that are not connected by s.

One might think a simple approach that arbitrarily places the positive neighbors at the top, the negative ones at the bottom, and the other unknown nodes at the middle in a ranking list. The simple approach will produce 1.0 GAUC for user preference preservation; however, this cannot work on link prediction since we need to predict target nodes among unknown nodes (i.e., they are not connected to a seed node). On the contrary, our model SRWR is effective for not only user preference preservation but also signed link prediction as shown in Table 3 and Fig. 7.

5.4 Troll identification task

In this section, we investigate the quality of a personalized ranking generated by SRWR in identifying trolls who behave abnormally or cause normal users to be irritated. The task is defined as follows: given a signed network and a normal user, identify trolls using a personalized ranking w.r.t. the user. In signed networks, we consider that a good personalized ranking of the normal user needs to capture trolls at the bottom of the ranking since most normal users are likely to dislike those trolls. Thus, we measure how well a personalized ranking of each method captures trolls at the bottom of the ranking to examine the quality of personalized node rankings.

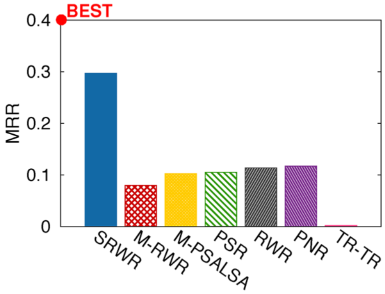
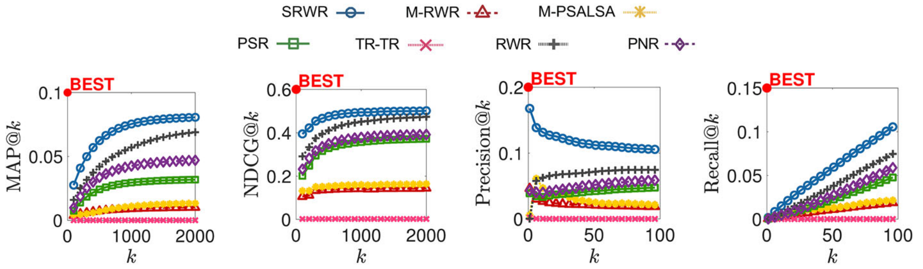


Fig. 8 MRR of SRWR. The measure indicates how trolls are ranked low in a personalized ranking. The SRWR is the highest MRR among all tested models



(a) MAP@k **(b)** NDCG@k **(c)** Precision@k **(d)** Recall@k

Fig. 9 The performance of ranking models for the troll identification task through various measurements: MAP@k (**a**), NDCG@k (**b**), Precision@k (**c**), and Recall@k (**d**). SRWR shows the best performance for all the measurements compared to other competitors

As in the previous work [23], we also use the enemies of a user called *No-More-Trolls* in the Slashdot dataset as trolls. The user is an administrative account created for the purpose of collecting a troll list (i.e., the administrator is negatively connected to each troll). There are 96 trolls in the list. We exclude the edges adjacent to *No-More-Trolls* from the Slashdot dataset and use the remaining edges to estimate a personalized ranking as a training set. We use the bottom-k of the ranking to search for those trolls. We randomly select 1000 seed nodes as a validation set to search for hyper-parameters required by each method. We pick the best parameter combination that provides the highest Mean Reciprocal Rank (MRR) in the validation set. Then, for each user, we search for trolls within the bottom-k ranking, and evaluate how those trolls are ranked low in the ranking, which is measured by MRR. We also measure Mean Average Precision (MAP@k), Normalized Discount Cumulated Gain (NDCG@k), Precision@k, and Recall@k to check the performance of each method in terms of various metrics (see the details in Appendix A.5.2). Since there are no user-graded scores for the troll list, we set those scores to 1 for NDCG.

Results Our proposed model SRWR significantly outperforms other ranking models for the troll identification task as shown in Figs. 8 and 9 . According to Fig. 8, the rank of a bottom ranked troll from our model is lower than that of other ranking models because MRR of our model is the highest compared to other competitors. More trolls are captured within the bottom-k ranking produced by our proposed model according to MAP@k shown in Fig. 9a. Note that Fig. 9c, d indicates that SRWR achieves higher Precision@k and Recall@k for capturing trolls than other methods. SRWR provides 4× better performance than PNR, the second best one, in terms of Precision@k when $k = 1$. Many trolls tend to be ranked low in our personalized ranking because SRWR achieves better MAP@k and NDCG@k than other ranking models as presented in Fig. 9a, b.

Table 4 Troll prediction results of ranking models w.r.t. a normal user *"yagu"*

	SRWR (proposed)		**M-RWR**		**M-PSALSA**		**PSR**		**TR-TR**	
Rank	**Trust Ranking**	**Distrust Ranking**	**Trust Ranking**	**Distrust Ranking**	**Trust Ranking**	**Distrust Ranking**	**Trust Ranking**	**Distrust Ranking**	**Trust Ranking**	**Distrust Ranking**
1	**yagu***	**Klerck**†	**yagu***	dubba-d	Work+Ac	HanzoSa	**yagu***	SmurfBu	**yagu***	Jack+B.
2	Photon+	**Adolf+H**†	Bruce+P	derago	Unknown	**Jerk+Ci**†	Uruk	Dr.Seus	dexterp	inTheLo
3	Uruk	GISGEOL	CmdrTac	msfodde	afidel	NineNin	Photon+	Doctor_	Jamie+Z	Mactrop
4	stukton	Nimrang	CleverN	cramus	heirony	Rogerbo	clump	artoo	ryanr	DiceMe
5	TTMuskr	Kafka_C	Uruk	lakerdo	bokmann	**SexyKel**†	TTMuskr	Juggle	KshGodd	Einstei
6	clump	Thinkit	Photon+	p414din	ezeri	ScottKi	stukton	FreakyG	TheIndi	FinchWo
7	Bruce+P	**CmderTa**†	stukton	an+unor	As+Seen	qurob	RxScram	RunFatB	daoine	Penus+T
8	RxScram	SteakNS	clump	exfuga	KillerD	bendodg	charlie	jmpoast	Berylli	r_glen
9	CmdrTac	JonKatz	TTMuskr	kryptok	potaz	ArnoldY	ssbg	El_Muer	danhara	Roland+
10	aphor	Henry+V	RxScram	toomz	byolinu	jcr	Idarubi	Ghost+H	Degrees	sting3r
11	CleverN	Miguel+	John+Ca	Shazzma	Stanist	davesch	spotted	The+Hob	charlie	Tuvai
12	throx	ringbar	aphor	Jetboy0	TripMas	List+of	Golias	bananac	sagei	1234567
13	chrisd	fimbulv	chrisd	KrisCow	andy+la	**yagu***	Slider4	Sodade	tadghin	1337_h4
14	CowboyN	by+Fort	TripMas	%2BMaje	bani	linzeal	Twid	peattle	capocci	1g%24ma
15	Blakey+	**I+Am+Th**†	kfg	frankyf	SETIGuy	jeffy12	Toast	GISGEOL	einstei	%2B%2Bg
16	Hemos	VAXGeek	Hemos	Lukano	generic	foobar1	Unknown	Nimrang	jebell	3p1ph4n
17	cgenman	NineNin	davesch	J'raxis	Bi()haz	dealsit	nanojat	**Adolf+H**†	pythorl	4d49434
18	TripMas	dfenstr	CowboyN	funklor	_xeno_	UbuntuD	lucasth	Kafka_C	sphere	4e61747
19	kfg	Quantum	NewYork	staynz7	HeyLaug	greenrd	pitboss	SteakNS	winneto	ABeowul
20	topham	SmurfBu	dada21	ikkibr	Eli+Got	Concern	harlows	Thinkit	Allen+V	Absolut

* This indicates the querying normal user.
† This indicates a troll.

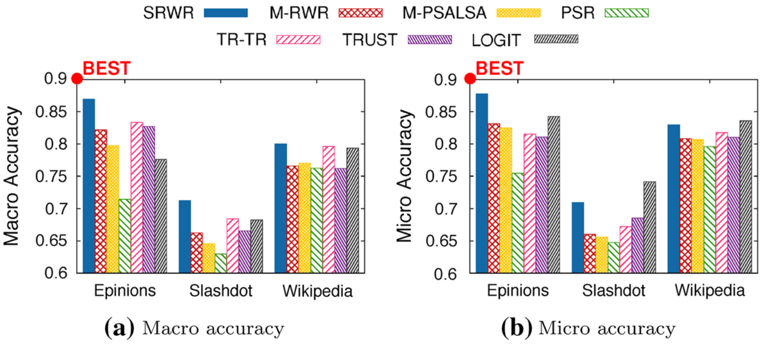
For each model, we show top-20 (trusted) and bottom-20 (distrusted) nodes based on the personalized ranking for "yagu". The users in the bottom-20 ranking are sorted in the ascending order of the ranking scores. Red-colored users (†) are trolls, a blue-colored user (⋆) is a query user, and the black-colored (non-marked) are normal users. Note that SRWR shows the best result: in SRWR, the query user is ranked 1st, and many trolls are ranked at the bottom in the personalized ranking. M-PSALSA provides inferior results since they rank the query user high at the bottom of the ranking, although the query user is the most trusted user for this task. M-RWR, PSR and TR-TR are not satisfactory either: they do not capture many trolls at the bottom of their rankings

Case study We investigate the top-20 and the bottom-20 of the personalized ranking for a user called *"yagu"* in Table 4. We list the users in the bottom-20 ranking in the ascending order of the ranking scores in Table 4. According to the result, more trolls are ranked low in the personalized ranking from SRWR, indicating that our model is more sensitive in capturing trolls than other models. Also, the query user is ranked low at the bottom of the ranking from M-PSALSA, while the user is ranked high in the ranking from our model. The query user should trust himself; thus, the user should be ranked at the top in a personalized ranking. This implies our model is more desirable than other models for personalized rankings in signed networks.

5.5 Sign prediction task

We evaluate ranking scores produced by each ranking model rather than the order between nodes. Note that a ranking score between a seed node s and a target node t is based on the trustworthiness between those nodes. Hence, it is also important to examine how well those ranking scores reflect trust relationships between nodes. We measure the quality of those ranking scores exploiting the sign prediction task which is defined as follows: given a signed network and a seed node s where signs of edges connected from s are missed, predict those signs using the personalized ranking scores of each method with respect to the seed node s.

To construct a validation set, we randomly select 1000 seed nodes and choose 20% edges of positive and negative links from each seed node. We also randomly select another 1000 seed nodes, and choose 20% positive and negative edges from each seed node to form a test set. Then, we remove each selected edge ($s \rightarrow t$) and predict the edge's sign based on personalized ranking scores w.r.t. node s in the graph represented by the remaining edges.



(a) Macro accuracy **(b)** Micro accuracy

Fig. 10 The performance of SRWR on the sign prediction task in terms of macro- and micro-accuracies where the macro-accuracy indicates the average seed-wise accuracy, and the micro-accuracy indicates the ratio of the number of correct predictions to the total test edges. While the micro-accuracy of SRWR is the second best, the macro-accuracy of SRWR is the best compared to its competitors

Our ranking score vector is $\mathbf{r} = \mathbf{r}^{+} - \mathbf{r}^{-}$ whose values range from -1 to 1. If $\mathbf{r}_t$ is greater than or equal to 0, then we predict the sign of the edge $(s \rightarrow t)$ as positive. Otherwise, it is considered as negative. We pick the best parameter combination having the highest micro-accuracy (see the below) in the validation set. With the validated parameters, we measure the following prediction accuracies of a test set, *macro- and micro-accuracies* which are defined as follows:

$$\text{macro-accuracy} = \frac{1}{n_Q}\sum_{i=1}^{n_Q}\text{accuracy}(i)$$

$$\text{micro-accuracy} = \frac{\#\text{ of correct predictions}}{\#\text{ of total test edges}}$$

where n_Q is the number of test seed nodes, and accuracy(i) is the seed-wise accuracy of ith test seed node (i.e., the ratio of the number of correct predictions to the number of test edges on ith seed node).

Results We compare the performance of SRWR to that of other random walk-based ranking models M-RWR, M-PSALSA, TR-TR, and PSR on the sign prediction task. We also compare our model SRWR to TRUST [12] and LOGIT [26] which are specially designed for predicting signs between two arbitrary nodes in signed networks. As shown in Fig. 10a, SRWR shows the best macro-accuracy among all tested methods. Although SRWR obtains higher micro-accuracy than LOGIT in the Epinions dataset, the micro-accuracy of LOGIT is better than that of SRWR in other datasets as shown in Fig. 10b.

Another observation is that LOGIT has a large gap between macro- and micro-accuracies, while SRWR has a small gap as shown in Fig. 10. A large gap implies that on average, the deviation between micro-accuracy and seed-wise accuracy (i.e., accuracy(i)) is large, i.e., accuracy(i) for ith test seed node is likely to deviate substantially from the micro-accuracy. To analyze such deviation, we look into seed-wise accuracies in terms of node degrees. Since there are a few high-degree nodes and a lot of low-degree nodes in real-world graphs according to power-law degree distribution [3], we split test seed nodes into two groups as follows: high (top-20%) and low (bottom-80%) groups in the order of out-degrees of test seed nodes. Then, we measure the average of seed-wise accuracies for each group (i.e., the

Table 5 The difference between SRWR and LOGIT in terms of macro-accuracies of high- and low-degree groups

Datasets	Methods	Overall Group	High-Degree Group	Low-Degree Group	Standard Deviation
Epinions	SRWR	**0.8696**	**0.8876**	**0.8651**	**0.0159**
	LOGIT	0.7762	0.8760	0.7510	0.0883
Slashdot	SRWR	**0.7128**	0.7133	**0.7127**	**0.0004**
	LOGIT	0.6827	**0.7943**	0.6546	0.0987
Wikipedia	SRWR	**0.8004**	0.8556	**0.7865**	**0.0489**
	LOGIT	0.7937	**0.8671**	0.7752	0.0650

The overall group is the union of the high- and low-degree groups. We measure the average of seed-wise accuracies for each group (i.e., the macro-accuracy of the group) and the standard deviation between accuracies of those groups. LOGIT tends to predict a seed node's test edges in the high-degree group better than SRWR, while SRWR predicts better those in the low-degree group compared to LOGIT. Also, the result on the standard deviation indicates that the disparity of SRWR between accuracies of those groups is smaller than that of LOGIT

macro-accuracy of the group) and the standard deviation between the accuracies of those groups.

According to Table 5, LOGIT tends to better predict test edges of a seed node in the high-degree group than those in the low-degree one. In particular, on the Epinions and the Slashdot datasets, the macro-accuracy of LOGIT in the low-degree group is rather lower than that of LOGIT in the high-degree group. These results imply that LOGIT is biased toward predicting test edges from a high-degree seed node. Note that the number of test edges from a high-degree node is larger than that of test edges from a low-degree node since 20% test edges are randomly extracted from each selected test node. Thus, the total number of correct predictions from LOGIT is large (i.e., the micro-accuracy becomes high). However, the seed-wise accuracies of LOGIT are low in the low-degree group (i.e., the macro-accuracy becomes low) as shown in Table 5, thereby increasing the gap of LOGIT between micro- and macro-accuracies.

On the contrary, the gap of SRWR between micro- and macro-accuracies are relatively smaller than that of LOGIT as shown in Fig. 10, along with the small standard deviation of SRWR as shown in Table 5. Note that SRWR outperforms LOGIT in the low-degree group over all the datasets as shown in Table 5. That is why the macro-accuracy of SRWR is higher than that of LOGIT for the total test seed nodes as shown in Fig. 10. SRWR also shows a satisfactory performance in the high-degree group, especially on the Epinions dataset, although the performance of SRWR is not better than that of LOGIT on the Slashdot and the Wikipedia datasets as shown in Table 5. Thus, the standard deviation of SRWR between those groups is smaller than that of LOGIT. These experimental results indicate that SRWR is competitive enough to be comparable to other models such as LOGIT in the sign prediction task.

Note that LOGIT is a graph feature-based method which exploits local graph features, within 1 hop from seed and target nodes, such as node degrees, common neighbors, and local wedges for predicting the sign between the seed and target nodes. A high-degree node is likely to have plentiful features, since the high-degree node has many connections to other nodes. A low-degree node would not have such local features enough due to less connections; hence, LOGIT has a limitation on increasing the predictive performance for test edges from the low-degree node based only on local graph features. On the other hand, SRWR's inference

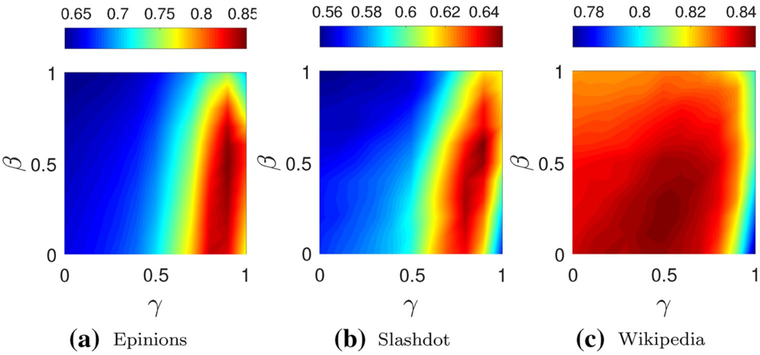


Fig. 11 Accuracy maps of SRWR according to β and γ where each color indicates a degree of accuracy. The Epinions and the Slashdot datasets present similar tendencies, while the Wikipedia dataset shows a different result from those of the two datasets

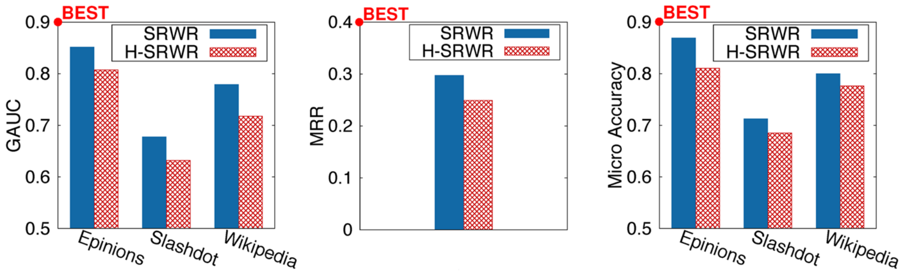
is based on the information more than 1 hop from the seed node because the signed random surfer visits the target node via various length of paths from the seed node to the target node. That is why SRWR works well on predicting test edges of low-degree nodes compared to LOGIT.

Balance attenuation factors We adjust the balance attenuation factors of SRWR, and evaluate the sign prediction task in terms of micro-accuracy to examine how well balance theory explains signed networks. In this experiment, we use the top-100 highest degree nodes as a test set for each network. The Epinions and the Slashdot datasets show similar results where larger values of β and γ achieve high accuracy as shown in Fig. 11a, b . Unlike these two datasets, the accuracy is high when β is small in the Wikipedia network as shown in Fig. 11c. This implies that "an enemy of my enemy is my friend" would not be correct in the network, which means balance theory does not apply well to the Wikipedia dataset. The reason is that the Wikipedia network represents votes between users to elect administrators; thus, the dataset is different from the Epinions and the Slashdot networks which are general social networks. Note that the validated balance attenuation factors for the sign prediction task in Sect. 5.1 are consistent with the tendency demonstrated in Fig. 11. Another observation is that the ideal balance theory does not apply to real-world signed networks because the accuracy is not the best over all datasets when $\beta = 1$ and $\gamma = 1$ (i.e., the ideal balance theory).

5.6 Effectiveness of balance attenuation factors

We examine the effects of the balance attenuation factors of SRWR on the performance of the link prediction, the troll identification, and the sign prediction tasks. In this experiment, we use H- SRWR ($\beta = 1$ and $\gamma = 1$) and SRWR with validated balance factors for each dataset as mentioned in Sect. 5.1. H- SRWR indicates that we compute SRWR scores using Eq. (2) which does not adopt balance attenuation factors. We measure GAUC for link prediction, MRR for troll prediction, and micro-accuracy for sign prediction to compare SRWR and H- SRWR.

Figure 12 indicates that introducing balance attenuation factors is helpful for improving the performance of each application in signed networks. As shown in Fig. 12a, SRWR obtains higher GAUC than H- SRWR in the link prediction task. Also, Fig. 12b presents that



(a) GAUC of the link prediction task

(b) MRR of the troll identification task

(c) Micro Accuracy of the sign prediction task

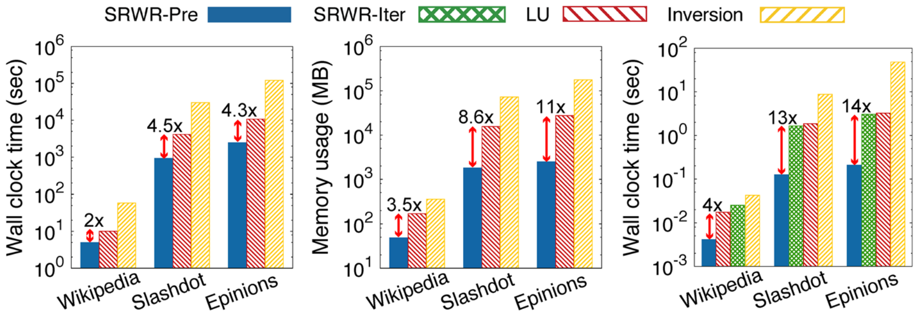
Fig. 12 Effect of the balance attenuation factors of SRWR. The performance of SRWR is better than that of H- SRWR (i.e., SRWR without using balance attenuation factors) in terms of the link prediction, the troll identification, and the sign prediction tasks

SRWR achieves better MRR than H- SRWR on the troll identification task. Moreover, the accuracy of SRWR is higher than that of H- SRWR over all datasets for the sign prediction task as presented in Fig. 12c. Although introducing balance attenuation factors increase the complexity of our model and demand an additional step for searching those factors, it makes our model flexible so that SRWR resolves the weakness inherent from the strong balance theory as discussed in Sect. 4.1.2 through adjusting those factors, and improves the performance of each application in signed social networks.

5.7 Performance of SRWR-Pre

We investigate the performance of our preprocessing method SRWR- Pre in terms of preprocessing time, memory space for precomputed data, and query time. We compare SRWR- Pre to other baseline preprocessing methods Inversion and LU as well as our iterative method SRWR- Iter. Preprocessing and query time are measured in wallclock time, and we measure the average query time for 1000 random seed nodes. We set $\beta = 0.5$, $\gamma = 0.5$, $c = 0.05$ for all tested methods. In SRWR- Pre, we set the hub selection ratio $t = 0.001$ for the hub-and-spoke reordering method to make the number of hubs n_2 small enough as in [34] We also measure how much memory space each preprocessing method needs for the precomputed matrices to compare memory efficiency. We omit bars for SRWR- Iter in Fig. 13a, b because SRWR- Iter does not involve a heavy preprocessing phase (i.e., the time cost for the normalization phase of SRWR- Iter in Algorithm 1 is trivial, and the memory usage of SRWR- Iter is equal to that of the input graph).

Figure 13a, b shows that SRWR- Pre provides better performance than LU and Inversion in terms of preprocessing time and memory space for preprocessed data. SRWR- Pre is up to 4.5× faster than the second best preprocessing method LU in terms of preprocessing time. Also, SRWR- Pre requires up to 11× less memory space than LU. Particularly, our method SRWR- Pre uses 2.6GB memory for the precomputed data in the Epinions dataset, while LU and Inversion require 28GB and 180GB memory, respectively. These results imply that SRWR- Pre is fast and memory efficient compared to other preprocessing methods. SRWR- Pre also shows the fastest query speed among other competitors including our iterative method SRWR- Iter as presented in Fig. 13c. SRWR- Pre is up to 14× faster than SRWR- Iter, and up to 15× faster than the second best preprocessing method LU in the Epinions dataset. Note that SRWR- Pre computes SRWR scores for a given seed node in less than



(a) Preprocessing time **(b)** Memory space for preprocessed data **(c)** Query time

Fig. 13 Performance of SRWR- PRE: **a** and **b** show the comparison of the preprocessing time and the memory space for preprocessed data among preprocessing methods; **c** compares the query time among all tested methods. SRWR- PRE presents the best performance compared to other preprocessing methods in terms of preprocessing time and memory efficiency. SRWR- PRE also computes SRWR scores more quickly than SRWR- ITER and the baseline methods

Table 6 Total number of nonzeros (nnz_t) in precomputed matrices for each preprocessing method

Dataset	A: nnz_t in SRWR- PRE	B: nnz_t in LU	C: nnz_t in Inversion	Ratio B/A	Ratio C/A
Wikipedia	3,207,758	11,257,644	23,928,232	3.51	7.46
Slashdot	119,580,272	1,032,276,955	4,817,461,830	8.63	40.29
Epinions	165,006,379	1,825,755,902	11,755,245,476	11.06	71.24

Our method SRWR- PRE generates less nonzeros in precomputed matrices than other preprocessing methods

0.3 second over all signed networks. Inversion is the slowest among the tested methods over all datasets. The main reason is that Inversion produces a very large number of nonzeros in precomputed matrices (e.g., Inversion produces about 11 billion nonzeros in the Epinions dataset as presented in Table 6). These results indicate that SRWR- PRE is appropriate to serve given queries in real-time on the datasets with low memory usage compared to other methods.

Discussion In this work, we propose two methods for SRWR: SRWR- ITER and SRWR- PRE which are iterative and preprocessing methods computing SRWR scores, respectively. SRWR- ITER does not require heavy precomputed data to compute SRWR scores. However, SRWR- ITER shows slow query speed as presented in Fig. 13c because SRWR- ITER should perform matrix vector multiplications many times for a given seed node. On the other hand, SRWR- PRE is faster up to 14× than SRWR- ITER in term of query speed since SRWR- PRE directly computes SRWR scores from precomputed data. However, in SRWR- PRE, the values of the parameters c, β, and γ of SRWR are fixed through the preprocessing phase (Algorithm 3); thus, SRWR- PRE cannot change the parameters in the query phase (Algorithm 4). To obtain SRWR scores with the different values of the parameters, we need to perform the preprocessing phase with the parameters again. On the contrary, SRWR- ITER easily handles the change of the parameters in the query phase (Algorithm 2) without additional operations such as preprocessing. One appropriate usage for our methods is that a user uses SRWR- ITER to find proper parameters for a specific application; and then, the user exploits SRWR- PRE with the discovered parameters to accelerate the query speed in the application.

6 Conclusion

We propose Signed Random Walk with Restart, a novel model which provides personalized trust or distrust rankings in signed networks. In our model, we introduce a signed random surfer so that she considers negative edges by changing her sign for surfing on signed networks. Consequently, our model provides personalized trust or distrust rankings reflecting signed edges. Our model is a generalized version of Random Walk with Restart working on both signed and unsigned networks. We also devise SRWR- Iter and SRWR- Pre, iterative and preprocessing methods to compute SRWR scores, respectively. We experimentally show that SRWR achieves the best accuracy for link prediction, predicts trolls 4× more accurately, and shows a satisfactory performance for inferring missing signs of edges compared to other methods. SRWR- Pre preprocesses a signed network up to 4.5× faster, and requires 11× less memory space than other preprocessing methods; SRWR- Pre computes SRWR scores up to 14× faster than other methods. Future research directions include developing a learning algorithm which automatically learns the balance attenuation factors of our model from a given input graph.

Acknowledgements This work was supported by Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) [2013-0-00179, Development of Core Technology for Context-aware Deep-Symbolic Hybrid Learning and Construction of Language Resources]. The Institute of Engineering Research at Seoul National University provided research facilities for this work. The ICT at Seoul National University provides research facilities for this study.

A Appendix

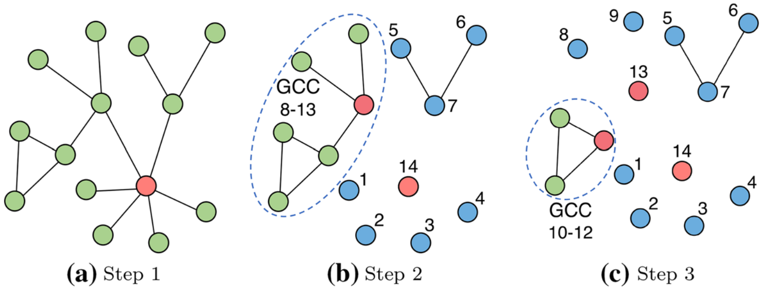
A.1 Details of the hub-and-spoke reordering method

SlashBurn [19,28] is a node reordering algorithm which concentrates nonzero entries of the adjacency matrix of a given graph based on the hub-and-spoke structure. Let n be the number of nodes in a graph, and t be the hub selection ratio whose range is between 0 and 1 where $\lceil tn \rceil$ indicates the number of nodes selected by SlashBurn as hubs. For each iteration, SlashBurn disconnects $\lceil tn \rceil$ high-degree nodes, called *hub nodes*, from the graph; then the graph is split into the giant connected component (GCC) and the disconnected components. The nodes in the disconnected components are called *spokes*, and each disconnected component forms a block in $|\mathbf{H}|_{11}$ (or $\mathbf{T}_{11}$) in Fig. 6. Then, SlashBurn reorders nodes such that the hub nodes get the highest ids, the spokes get the lowest ids, and the nodes in the GCC get the ids in the middle. SlashBurn repeats this procedure on the GCC recursively until the size of GCC becomes smaller than $\lceil tn \rceil$. After SlashBurn is done, the reordered adjacency matrix contains a large and sparse block diagonal matrix in the upper left area, as shown in Fig. 6. Figure 14 depicts the procedure of SlashBurn when $\lceil tn \rceil = 1$.

A.2 Properties and lemmas

A.2.1 Sum of positive and negative SRWR scores

Property 3 *Consider the recursive equation* $\mathbf{p} = (1-c)|\tilde{\mathbf{A}}|^\top \mathbf{p} + c\mathbf{q}$ *where* $\mathbf{p} = \mathbf{r}^+ + \mathbf{r}^-$ *and* $|\tilde{\mathbf{A}}|^\top$ *is a column stochastic matrix. Then* $\mathbf{1}^\top \mathbf{p} = \sum_i \mathbf{p}_i = 1$.



(a) Step 1 **(b)** Step 2 **(c)** Step 3

Fig. 14 Node reordering based on hub-and-spoke method when $\lceil tn \rceil = 1$ where $\lceil tn \rceil$ indicates the number of selected hubs at each step, and t is the hub selection ratio ($0 < t < 1$). Red nodes are hubs; blue nodes are spokes that belong to the disconnected components; green colored are nodes that belong to the giant connected component. At Step 1 in (**a**), the method disconnects a hub node, and assigns node ids as shown in (**b**). The hub node gets the highest id (14), the spoke nodes get the lowest ids (1–7), and the GCC gets the middle ids (8–13). The next iteration starts on the GCC in (**b**), and the node ids are assigned as in (**c**)

Proof By multiplying both sides by $\mathbf{1}^\top$, the equation is represented as follows:

$$\mathbf{p} = (1-c)|\tilde{\mathbf{A}}|^\top \mathbf{p} + c\mathbf{q} \Leftrightarrow \mathbf{1}^\top \mathbf{p} = (1-c)\mathbf{1}^\top |\tilde{\mathbf{A}}|^\top \mathbf{p} + c\mathbf{1}^\top \mathbf{q}$$

Note that $\mathbf{1}^\top |\tilde{\mathbf{A}}|^\top = (|\tilde{\mathbf{A}}|\mathbf{1})^\top$, and $|\tilde{\mathbf{A}}|$ is a row stochastic matrix; thus, $(|\tilde{\mathbf{A}}|\mathbf{1})^\top = \mathbf{1}^\top$. Hence, the above equation is represented as follows:

$$\mathbf{1}^\top \mathbf{p} = (1-c)\mathbf{1}^\top |\tilde{\mathbf{A}}|^\top \mathbf{p} + c\mathbf{1}^\top \mathbf{q} \Leftrightarrow \mathbf{1}^\top \mathbf{p} = (1-c)\mathbf{1}^\top \mathbf{p} + c \Leftrightarrow \mathbf{1}^\top \mathbf{p} = 1$$

□

A.2.2 Analysis on number of iterations of SRWR-Iter

Lemma 3 *Suppose* $\mathbf{h} = [\mathbf{r}^+; \mathbf{r}^-]^\top$, *and* $\mathbf{h}^{(k)}$ *is the result of kth iteration in* SRWR- Iter. *Let* $\delta^{(k)}$ *denote the error* $\|\mathbf{h}^{(k)} - \mathbf{h}^{(k-1)}\|_1$. *Then* $\delta^{(k)} \leq 2(1-c)^k$, *and the estimated number T of iterations for convergence is* $\log_{1-c} \frac{\epsilon}{2}$ *where* ϵ *is an error tolerance, and c is the restart probability.*

Proof According to Eq. (4), $\delta^{(k)}$ is represented as follows:

$$\begin{aligned}
\delta^{(k)} = \|\mathbf{h}^{(k)} - \mathbf{h}^{(k-1)}\|_1 &= (1-c)\|\tilde{\mathbf{B}}^\top(\mathbf{h}^{(k-1)} - \mathbf{h}^{(k-2)})\|_1 \\
&\leq (1-c)\|\tilde{\mathbf{B}}^\top\|_1 \|\mathbf{h}^{(k-1)} - \mathbf{h}^{(k-2)}\|_1 \\
&= (1-c)\|\mathbf{h}^{(k-1)} - \mathbf{h}^{(k-2)}\|_1 = (1-c)\delta^{(k-1)}
\end{aligned}$$

Note that $\|\tilde{\mathbf{B}}^\top\|_1 = 1$ since $\tilde{\mathbf{B}}^\top$ is column stochastic as described in Theorem 1. Hence, $\delta^{(k)} \leq (1-c)\delta^{(k-2)} \leq \cdots \leq (1-c)^k \delta^{(1)}$. Since $\delta^{(1)} = \|\mathbf{h}^{(1)} - \mathbf{h}^{(0)}\|_1 \leq \|\mathbf{h}^{(1)}\|_1 + \|\mathbf{h}^{(0)}\|_1 = 2$, $\delta^{(k)} \leq 2(1-c)^k$. Note that when $\delta^{(k)} \leq \epsilon$, the iteration of SRWR- Iter is terminated. Thus, for $k \leq \log_{1-c} \frac{\epsilon}{2}$, the iteration is terminated, and the number T of iterations for convergence is estimated at $\log_{1-c} \frac{\epsilon}{2}$. □

A.2.3 Time complexity of sparse matrix multiplication

Lemma 4 (Sparse Matrix Multiplication [32]) *Suppose that* $\mathbf{A}$ *and* $\mathbf{B}$ *are* $p \times q$ *and* $q \times r$ *sparse matrices, respectively, and* $\mathbf{A}$ *has* $\mathrm{nnz}(\mathbf{A})$ *nonzeros. Calculating* $\mathbf{C} = \mathbf{AB}$ *using sparse matrix multiplication requires* $O(\mathrm{nnz}(\mathbf{A})r)$.

A.3 Complexity analysis of proposed methods for SRWR

We analyze the complexity of our proposed methods SRWR- ITER and SRWR- PRE in terms of time and space. The space and time complexities of SRWR- ITER are presented in Lemma 5, and those of SRWR- PRE are in Lemmas 6, 7, and 8 , respectively.

A.3.1 Space and time complexities of SRWR-ITER

Lemma 5 (Space and Time Complexities of SRWR- ITER) *Let n and m denote the number of nodes and edges of a signed network, respectively. Then the space complexity of Algorithm 2 is* $O(n+m)$. *The time complexity of Algorithm 2 is* $O(T(n+m))$ *where the number* T *of iterations is* $\log_{1-c}\frac{\epsilon}{2}$, *c is the restart probability, and* ϵ *is an error tolerance.*

Proof The space complexity for $\tilde{\mathbf{A}}_{+}$ and $\tilde{\mathbf{A}}_{-}$ is $O(m)$ if we exploit a sparse matrix format such as compressed column storage to save the matrices. We need $O(n)$ for SRWR score vectors $\mathbf{r}^{+}$ and $\mathbf{r}^{-}$. Thus, the space complexity is $O(n+m)$. One iteration in Algorithm 2 takes $O(n+m)$ time due to sparse matrix vector multiplications and vector additions where the time complexity of a sparse matrix vector multiplication is linear to the number of nonzeros of a matrix [7]. Hence, the total time complexity is $O(T(n+m))$ where the number T of iterations is $\log_{1-c}\frac{\epsilon}{2}$ which is proved in Lemma 3. □

A.3.2 Space and time complexities of SRWR-PRE

Lemma 6 (Space Complexity of SRWR- PRE) *The space complexity of the preprocessed matrices from* SRWR- PRE *is* $O(n_2^2 + m)$ *where* n_2 *is the number of hubs and m is the number of edges in the graph.*

Proof The space complexity of each preprocessed matrix is summarized in Table 7. $\tilde{\mathbf{A}}_{-}$, $|\mathbf{H}|_{12}$, $|\mathbf{H}|_{21}$, $\mathbf{T}_{12}$, and $\mathbf{T}_{21}$ are sparse matrices, and constructed from the input graph; hence, the space complexity is bounded by the number of edges (i.e., $O(m)$). Note that $|\mathbf{H}|$ and $\mathbf{T}$ have the same sparsity pattern; hence, $|\mathbf{H}|_{11}$ and $\mathbf{T}_{11}$ identified by [19,28] have the same b blocks. The ith block in $|\mathbf{H}|_{11}^{-1}$ (or $\mathbf{T}_{11}^{-1}$) contains n_{1i}^2 nonzeros; therefore, $|\mathbf{H}|_{11}^{-1}$ and $\mathbf{T}_{11}^{-1}$ require $O(\sum_{i=1}^{b} n_{1i}^2)$ space, respectively. Since the dimension of $\mathbf{L}_{|\mathbf{H}|}^{-1}$, $\mathbf{U}_{|\mathbf{H}|}^{-1}$, $\mathbf{L}_{\mathbf{T}}^{-1}$, and $\mathbf{U}_{\mathbf{T}}^{-1}$ is n_2, they require $O(n_2^2)$ space. □

Note that the blocks in $|\mathbf{H}|_{11}$ (or $\mathbf{T}_{11}$) are discovered by the reordering method [19,28] as briefly described in Appendix A.1. In real-world graphs, $\sum_{i=1}^{b} n_{1i}^2$ can be bounded by $O(m)$ as shown in [34]. Hence, we assume that the space complexity of $|\mathbf{H}|_{11}^{-1}$ and $\mathbf{T}_{11}^{-1}$ is $O(m)$ for simplicity.

Lemma 7 (Time Complexity of Preprocessing Phase in SRWR- PRE) *The preprocessing phase in Algorithm 3 takes* $O(T(m + n\log n) + n_2^3 + mn_2)$ *where* $T = \lceil \frac{n_2}{tn} \rceil$ *is the number of iterations, and t is the hub selection ratio in the hub-and-spoke reordering method [19,28].*

Table 7 Space complexity of each preprocessed matrix from Algorithm 3

Matrix	Space complexity
$\tilde{\mathbf{A}}_{-}$, $\lvert\mathbf{H}\rvert_{12}$, $\lvert\mathbf{H}\rvert_{21}$, $\mathbf{T}_{12}$, and $\mathbf{T}_{21}$	$O(m)$
$\lvert\mathbf{H}\rvert_{11}^{-1}$, and $\mathbf{T}_{11}^{-1}$	$O(\sum_{i=1}^{b} n_{1i}^2) = O(m)$
$\mathbf{L}_{\lvert\mathbf{H}\rvert}^{-1}$, $\mathbf{U}_{\lvert\mathbf{H}\rvert}^{-1}$, $\mathbf{L}_{\mathbf{T}}^{-1}$, and $\mathbf{U}_{\mathbf{T}}^{-1}$	$O(n_2^2)$

Note that m is the number of edges of the input graph; n_2 is the number of hubs, and n_{1i} is the number of nodes in ith block where b blocks in $\lvert\mathbf{H}\rvert_{11}$ (or $\mathbf{T}_{11}$) are identified by the hub-and-spoke reordering method

Proof We only consider the main factors of the time complexity of Algorithm 3 in this proof. The hub-and-spoke reordering method takes $O(T(m+n\log n))$ time (line 1) where T is $\lceil \frac{n_2}{tn} \rceil$ which is proved in [19,28]. Computing the Schur complement of $\lvert\mathbf{H}\rvert_{11}$ takes $O(n_2^2 + mn_2)$ because it takes $O(mn_2)$ to compute $\mathbf{P}_1 = \lvert\mathbf{H}\rvert_{11}^{-1}\lvert\mathbf{H}\rvert_{12}$ and $\mathbf{P}_2 = \lvert\mathbf{H}\rvert_{21}\mathbf{P}_1$ by Lemma 4, and $O(n_2^2)$ to compute $\lvert\mathbf{H}\rvert_{22} - \mathbf{P}_2$ (line 6). It takes $O(n_2^3)$ to compute the inverse of the LU factors (line 8). Note that computing $\lvert\mathbf{H}\rvert_{11}^{-1}$(line 4) requires $O(\sum_{i=1}^{b} n_{1i}^3)$ time where it takes n_{1i}^3 to obtain the inverse of ith block. In real-world networks, the size n_{1i} of each block is much smaller than the number n_2 of hubs; thus, we assume that $\sum_{i=1}^{b} n_{1i}^3 \ll n_2^3$ [34]. Hence, the time complexity of preprocessing $\lvert\mathbf{H}\rvert$ is $O(T(m+n\log n) + n_2^3 + mn_2)$. Note that the time complexity of preprocessing $\mathbf{T}$ is included into that of preprocessing $\lvert\mathbf{H}\rvert$ since $\mathbf{T}$ and $\lvert\mathbf{H}\rvert$ have the same sparsity pattern. □

Lemma 8 (Time Complexity of Query Phase in SRWR- Pre) *The query phase in Algorithm 4 takes $O(n_2^2 + n + m)$ time.*

Proof We only consider the main factors of the time complexity of Algorithm 4 in this proof. It takes $O(n_2^2 + m)$ to compute $\mathbf{p}_2$ since it takes $O(n_2 + m)$ to compute $\tilde{\mathbf{q}}_2 = \mathbf{q}_2 - \lvert\mathbf{H}\rvert_{21}(\lvert\mathbf{H}\rvert_{11}^{-1}\mathbf{q}_1)$, and $O(n_2^2)$ to compute $\mathbf{U}_{\lvert\mathbf{H}\rvert}^{-1}(\mathbf{L}_{\lvert\mathbf{H}\rvert}^{-1}\tilde{\mathbf{q}}_2)$ (line 2). It takes $O(n)$ time to concatenate the partitioned vectors (lines 4 and 8) and compute $\mathbf{r}^{+}$ and $\mathbf{r}$ (lines 9 and 10). Hence, the total time complexity of the query phase is $O(n_2^2 + n + m)$. □

A.4 Detailed limitations of existing random walk-based ranking models in signed networks

In this section, we describe the detailed limitation of existing random walk-based ranking models which are briefly described in Sect. 1.

- Random Walk with Restart (RWR): We perform RWR on a given signed network after taking absolute edge weights to obtain $\mathbf{r}$ as follows:

$$\mathbf{r} = (1-c)\lvert\tilde{\mathbf{A}}\rvert^{\top}\mathbf{r} + c\mathbf{q}$$

  where $\lvert\tilde{\mathbf{A}}\rvert$ is the row-normalized matrix of the absolute adjacency matrix in the signed network. RWR does not properly consider negative edges for $\mathbf{r}$.
- Modified Random Walk with Restart (M-RWR) [33]: M-RWR applies RWR separately on both a positive subgraph and a negative subgraph; thus, it obtains $\mathbf{r}^{+}$ on the positive subgraph and $\mathbf{r}^{-}$ on the negative subgraph, and then, computes $\mathbf{r} = \mathbf{r}^{+} - \mathbf{r}^{-}$. The detailed equations for M-RWR are as follows:

$$\mathbf{r}^{+} = (1-c)\tilde{\mathbf{B}}_{+}^{\top}\mathbf{r}^{+} + c\mathbf{q} \text{ and } \mathbf{r}^{-} = (1-c)\tilde{\mathbf{B}}_{-}^{\top}\mathbf{r}^{-} + c\mathbf{q}$$

where $\tilde{\mathbf{B}}_{+}$ is the row-normalized matrix of the adjacency matrix containing only positive edges, and $\tilde{\mathbf{B}}_{-}$ is that of the absolute adjacency matrix containing only negative edges. The main limitation of M-RWR is that it does not consider relationships between positive and negative edges due to the separation as shown in the above equations.

- Modified Personalized SALSA (M-PSALSA) [30]: Andrew et al. made a modification on SALSA[1] by introducing the random jump into it, called Personalized SALSA (PSALSA). As similar to M-RWR, we apply PSALSA separately on both positive and negative subgraphs and consider authorities on the positive subgraph as $\mathbf{r}^{+}$, and those scores on the negative subgraph as $\mathbf{r}^{-}$. M-PSALSA also has the same limitation with M-RWR.
- Personalized Signed Spectral Rank (PSR) [23]: Kunegis et al. proposed PSR which is a variant of PageRank by constructing the following matrix similar to Google matrix:

$$\mathbf{M}_{PSR} = (1-c)\mathbf{D}^{-1}\mathbf{A}^{\top} + c\mathbf{e}_s\mathbf{1}^{\top}$$

where $\mathbf{A}$ is the signed adjacency matrix, $\mathbf{D}$ is the diagonal out-degree matrix, and $\mathbf{e}_s$ is the sth unit vector. Then, PSR computes the left eigenvector of $\mathbf{M}_{PSR}$, which induces a relative trustworthy score vector $\mathbf{r}$ including positive and negative values. Although PSR is able to produce $\mathbf{r}$, the equation for PSR is heuristic because $\mathbf{M}_{PSR}$ is not a column stochastic matrix. Also, how the random surfer based on the equation interprets negative edges is veiled.

A.5 Detailed description of evaluation metrics

We describe the details of metrics used in the link prediction and the troll identification tasks. The metrics for the sign prediction task is described in Sect. 5.5.

A.5.1 Link prediction

- GAUC (Generalized AUC): Song et al. [35] proposed GAUC which measures the quality of link prediction in signed networks. An ideal personalized ranking w.r.t. a seed node s needs to rank nodes with positive links to s at the top, those with negative links at the bottom, and other unknown status nodes in the middle of the ranking. For a seed node s, suppose that $\mathbf{P}_s$ is the set of positive nodes potentially connected by s, $\mathbf{N}_s$ is that of negative nodes, and $\mathbf{O}_s$ is that of the other nodes. Then, GAUC of the personalized ranking w.r.t. s is defined as follows:

$$\begin{aligned}\mathrm{GAUC}_s = &\frac{\eta}{|\mathbf{P}_s|(|\mathbf{O}_s| + |\mathbf{N}_s|)}\left(\sum_{p\in\mathbf{P}_s}\sum_{i\in\mathbf{O}_s\cup\mathbf{N}_s}\mathbb{I}(\mathbf{r}_p > \mathbf{r}_i)\right)\\ &+ \frac{1-\eta}{|\mathbf{N}_s|(|\mathbf{O}_s| + |\mathbf{P}_s|)}\left(\sum_{i\in\mathbf{O}_s\cup\mathbf{P}_s}\sum_{n\in\mathbf{N}_s}\mathbb{I}(\mathbf{r}_i < \mathbf{r}_n)\right)\end{aligned}$$

where $\eta = \frac{|\mathbf{P}_s|}{|\mathbf{P}_s|+|\mathbf{N}_s|}$ is the relative ratio of the number of positive edges and that of

[1] SALSA [25] is a normalized version of HITS [22].

negative edges, and $\mathbb{I}(\cdot)$ is an indicator function that returns 1 if a given predicate is true, or 0 otherwise. GAUC will be 1.0 for the perfect ranking list and 0.5 for a random ranking list [35].

- AUC (Area Under the Curve): AUC of the personalized ranking scores $\mathbf{r}$ w.r.t. seed node s in signed networks is defined as follows [35]:

$$\mathrm{AUC}_s = \frac{1}{|\mathbf{P}_s||\mathbf{N}_s|} \sum_{p \in \mathbf{P}_s} \sum_{n \in \mathbf{N}_s} \mathbb{I}(\mathbf{r}_p > \mathbf{r}_n)$$

  where $\mathbf{P}_s$ is the set of positive nodes potentially connected by s, and $\mathbf{N}_s$ is the set of negative nodes. $\mathbb{I}(\cdot)$ is an indicator function that returns 1 if a given predicate is true, or 0 otherwise. With an ideal ranking list, AUC should be 1 representing each positive sample is ranked higher than all the negative samples. For a random ranking, AUC will be 0.5. However, AUC is not a satisfactory metric for the link prediction task in signed networks because AUC is designed for two classes (positive and negative), while the link prediction in signed networks should consider three classes (positive, unknown, and negative) as described in the above.

A.5.2 Troll identification

Suppose that we have a personalized ranking $\mathcal{R}$ in the ascending order of the trustworthiness scores w.r.t. a seed node (i.e., a node with a low score is ranked high) to have the same effect of searching trolls in the bottom of the original ranking in the descending order of those scores.

- MAP@k (Mean Average Precision): MAP@k is the mean of average precisions, AP@k, for multiple queries. Suppose that there are l trolls to be captured. Then, AP@k is defined as follows:

$$\mathrm{AP@}k = \frac{1}{\min(l, k)} \left(\sum_{t \in \mathbf{T}} \mathrm{Precision@}t \right)$$

  where Precision@t is the precision at the cutoff t. Note that $\mathbf{T} = \{t | \mathbb{I}(\mathcal{R}[t]) = 1 \text{ for } 1 \leq t \leq k\}$ where $\mathcal{R}[t]$ denotes the user ranked at position t in the ranking $\mathcal{R}$, and $\mathbb{I}(\mathcal{R}[t])$ is 1 if $\mathcal{R}[t]$ is a troll. For N queries, MAP@k is defined as follows:

$$\mathrm{MAP@}k = \frac{1}{N} \left(\sum_{i=1}^{N} \mathrm{AP@}k \right)$$

- NDCG@k (Normalized Discount Cumulative Gain): NDCG is the normalized value of Discount Cumulative Gain (DCG), which is defined as follows:

$$\mathrm{DCG@}k = rel_1 + \sum_{i=2}^{k} \frac{rel_i}{log_2(i)}, \quad \text{and} \quad \mathrm{NDCG@}k = \frac{\mathrm{DCG@}k}{\mathrm{IDCG@}k}$$

  where rel_i is the user-graded relevance score for the ith ranked item. Then, NDCG@k is obtained by normalizing using Ideal DCG(IDCG) which is the DCG for the ideal order of ranking.
- Precision@k and Recall@k: Precision@k (Recall@k) is the precision (recall) at the cutoff k in a ranking. Precision@k is the ratio of identified trolls in top-k ranking, and Recall@k is the ratio of identified trolls in the total trolls.

- MRR (Mean Reciprocal Rank): MRR@k is the mean of the reciprocal rank (RR) for each the top-k query response. RR is the multiplicative inverse of the rank of the first correct answer. Hence, for N multiple queries, MRR@k is defined as follows:

$$\text{MRR@}k = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i}$$

where $rank_i$ is the rank position of the first relevant item in the top-k ranking. If there is no relevant item in the ranking for the ith query, the inverse of the rank, ${rank_i}^{-1}$, becomes zero.

A.6 Discussion on relative trustworthiness scores of SRWR

In Sect. 4.1, we define the relative trustworthiness $\mathbf{r} = \mathbf{r}^{+} - \mathbf{r}^{-}$ where $\mathbf{r}^{+}$ is for positive SRWR scores, and $\mathbf{r}^{-}$ is for negative SRWR ones. We show that $\mathbf{r}^{+}$ and $\mathbf{r}^{-}$ are *measures*, and $\mathbf{r}$ is a *signed measure* using definitions from measure theory [38]. We first introduce the definition of *measure* as follows:

Definition 5 (*Measure* [38]) A measure μ on a (finite) set Ω with σ-algebra $\mathcal{A}$ is a function $\mu : \mathcal{A} \rightarrow \mathbb{R}_{\geq 0}$ such that

1. (Nonnegativity) $\mu(E) \geq 0 \; \forall E \in \mathcal{A}$,
2. (Null empty set) $\mu(\emptyset) = 0$,
3. (Countable additivity) $\mu(\bigcup_{i=1}^{\infty} E_i) = \sum_{i=1}^{\infty} E_i$ for any sequence of pairwise disjoint sets, $E_1, E_2, \ldots \in \mathcal{A}$

where σ-algebra $\mathcal{A}$ on Ω is a collection $\mathcal{A} \subseteq 2^{\Omega}$ s.t. it is nonempty, and closed under complements (i.e., $E \in \mathcal{A} \Rightarrow E^c \in \mathcal{A}$) and countable unions (i.e., $E_1, E_2, \cdots \in \mathcal{A} \Rightarrow \bigcup_{i=1}^{\infty} E_i \in \mathcal{A}$). The pair of $(\Omega, \mathcal{A})$ is called *measurable space*. □

In probability theory, σ-algebra $\mathcal{A}$ describes all possible events to be measured as probability. Note that $\mathbf{r}^{+}$ and $\mathbf{r}^{-}$ are joint probabilities of nodes and signs, i.e., $\mathbf{r}_u^{+} = P(N = u, S = +)$ and $\mathbf{r}_u^{-} = P(N = u, S = -)$ where N is a random variable of nodes, and S is a random variable of the surfer's sign. Note that N takes an item from σ-algebra $\mathcal{A}$. The following property shows that $\mathbf{r}^{+}$ and $\mathbf{r}^{-}$ are (nonnegative) measures.

Property 4 *Suppose Ω is the set $\mathbf{V}$ of nodes, and σ-algebra $\mathcal{A}$ on Ω is 2^{Ω}. Let $\mu^{+} = P(N, S = +)$ and $\mu^{-} = P(N, S = -)$. Then, both μ^{+} and μ^{-} are (nonnegative) measures according to Definition 5.*

Proof For any $E \in \mathcal{A}$, $\mu^{+}(E) \geq 0$ and $\mu^{+}(\emptyset) = 0$ are obviously true since $P(N, S = +)$ is a probability; hence, $P(E, S = +) \geq 0$ and $P(\emptyset, S = +) = 0$. Let $(E_n)_{n\in\mathbb{N}}$ be a sequence of pairwise disjoint sets where $E_n \in \mathcal{A}$. Since the sets in the sequence are mutually disjoint, the following holds:

$$P\left(\bigcup_{n\in\mathbb{N}} E_n, S = +\right) = \sum_{n\in\mathbb{N}} P(E_n, S = +)$$

Therefore, $\mu^{+} = P(N, S = +)$ is a measure by Definition 5. Similarly, $\mu^{-} = P(N, S = -)$ is also a measure. □

Next, we introduce the definition of *signed measure*, a generalized version of measure by allowing it to have negative values.

Definition 6 (*Signed Measure* [38]) Given a set Ω and σ-algebra $\mathcal{A}$, a signed measure on $(\Omega, \mathcal{A})$ is a function $\mu : \mathcal{A} \to \mathbb{R}$ such that

1. (Real value) $\mu(E)$ takes a real value in $\mathbb{R}$,
2. (Null empty set) $\mu(\emptyset) = 0$,
3. (Countable additivity) $\mu(\bigcup_{i=1}^{\infty} E_i) = \sum_{i=1}^{\infty} E_i$ for any sequence of pairwise disjoint sets, $E_1, E_2, \cdots \in \mathcal{A}$ □

Note that *Shannon entropy* and *electric charge* are representative examples of signed measure. Then, the following lemma indicates the difference between two nonnegative measures is a signed measure.

Lemma 9 (Difference Between Two Nonnegative Measures [38]) *Suppose we are given nonnegative measure μ^+ and μ^- on the same measurable space $(\Omega, \mathcal{A})$. Then, $\mu = \mu^+ - \mu^-$ is a signed measure.*

Proof Since μ^+ and μ^- are nonnegative, μ is located between $-\infty$ and ∞. Also, $\mu(\emptyset) = \mu^+(\emptyset) - \mu^-(\emptyset) = 0$. Moreover, μ is countable additive, i.e.,

$$\mu\left(\bigcup_{i=1}^{\infty} E_i\right) = \mu^+\left(\bigcup_{i=1}^{\infty} E_i\right) - \mu^-\left(\bigcup_{i=1}^{\infty} E_i\right) = \sum_{i=1}^{\infty}\left(\mu^+(E_i) - \mu^-(E_i)\right) = \sum_{i=1}^{\infty} \mu(E_i)$$

Hence, $\mu = \mu^+ - \mu^-$ is a signed measure according to Definition 6. □

Lemma 9 implies that the relative trustworthiness $\mathbf{r} = \mathbf{r}^+ - \mathbf{r}^-$ is a signed measure. The trustworthiness $\mathbf{r}_u$ measures a degree of trustworthiness between seed node s and node u: if $\mathbf{r}_u > 0$, seed node s is likely to trust node u as much as $\mathbf{r}_u$, while if $\mathbf{r}_u < 0$, s is likely to distrust u as much as $\mathbf{r}_u$.

References

1. Backstrom L, Leskovec J (2011) Supervised random walks: predicting and recommending links in social networks. In: Proceedings of the fourth ACM international conference on Web search and data mining. ACM, pp 635–644
2. Bahmani B, Chowdhury A, Goel A (2010) Fast incremental and personalized pagerank. Proc VLDB Endow 4(3):173–184
3. Barabási A-L, Albert R (1999) Emergence of scaling in random networks. Science 286(5439):509–512
4. Boyd S, Vandenberghe L (2004) Convex optimization. Cambridge University Press, Cambridge
5. Cartwright D, Harary F (1956) Structural balance: a generalization of heider's theory. Psychol Rev 63(5):277
6. Davis JA (1967) Clustering and structural balance in graphs. Hum Relat 20(2):181–187
7. Duff IS, Grimes RG, Lewis JG (1989) Sparse matrix test problems. ACM Trans Math Softw (TOMS) 15(1):1–14
8. Easley D, Kleinberg J (2010) Networks, crowds, and markets: reasoning about a highly connected world. Cambridge University Press, Cambridge
9. Fujiwara Y, Nakatsuji M, Onizuka M, Kitsuregawa M (2012) Fast and exact top-k search for random walk with restart. Proc VLDB Endow 5(5):442–453
10. Gleich DF, Seshadhri C (2012) Vertex neighborhoods, low conductance cuts, and good seeds for local community methods. In: Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining. ACM, pp 597–605
11. Golub GH, Van Loan CF (2012) Matrix computations, vol 3. JHU press
12. Guha R, Kumar R, Raghavan P, Tomkins A (2004) Propagation of trust and distrust. In: Proceedings of the 13th international conference on World Wide Web. ACM, pp 403–412
13. Haveliwala TH (2002) Topic-sensitive pagerank. In: Proceedings of the 11th international conference on World Wide Web. ACM, pp 517–526

14. Heider F (1946) Attitudes and cognitive organization. J Psychol 21(1):107–112
15. Jin W, Jung J, Kang U (2019) Supervised and extended restart in random walks for ranking and link prediction in networks. PLoS ONE 14(3):e0213857
16. Jung J, Jin W, Sael L, Kang U (2016) Personalized ranking in signed networks using signed random walk with restart. In: IEEE 16th international conference on data mining, ICDM 2016, December 12–15, 2016, Barcelona, Spain, pp 973–978. http://dx.doi.org/10.1109/ICDM.2016.0122
17. Jung J, Park N, Sael L, Kang U (2017) Bepi: Fast and memory-efficient method for billion-scale random walk with restart. In: Proceedings of the 2017 ACM international conference on management of data, SIGMOD conference 2017, Chicago, IL, USA, May 14–19, 2017, pp 789–804
18. Jung J, Shin K, Sael L, Kang U (2016) Random walk with restart on large graphs using block elimination. ACM Trans Database Syst 41(2):12. https://doi.org/10.1145/2901736
19. Kang U, Faloutsos C (2011) Beyond 'caveman communities': hubs and spokes for graph compression and mining, in 'ICDM'
20. Kang U, Tong H, Sun J (2012) Fast random walk graph kernel. In: Proceedings of the twelfth SIAM international conference on data mining, Anaheim, California, USA, April 26-28, 2012, pp 828–838
21. Kleinberg JM (1999) Authoritative sources in a hyperlinked environment. J ACM (JACM) 46(5):604–632
22. Kleinberg JM (1999) Hubs, authorities, and communities. ACM Comput Surveys (CSUR) f31(4es):5
23. Kunegis J, Lommatzsch A, Bauckhage C (2009) The slashdot zoo: mining a social network with negative edges. In: Proceedings of the 18th international conference on World wide web. ACM, pp 741–750
24. Langville AN, Meyer CD, Fernández P (2008) Googles pagerank and beyond: the science of search engine rankings. Math Intell 30(1):68–69
25. Lempel R, Moran S (2001) Salsa: the stochastic approach for link-structure analysis. ACM Trans Inf Syst (TOIS) 19(2):131–160
26. Leskovec J, Huttenlocher D, Kleinberg J (2010) Predicting positive and negative links in online social networks. In: Proceedings of the 19th international conference on World Wide Web. ACM, pp 641–650
27. Leskovec J, Huttenlocher D, Kleinberg J (2010) Signed networks in social media. In: Proceedings of the SIGCHI conference on human factors in computing systems. ACM, pp 1361–1370
28. Lim Y, Kang U, Faloutsos C (2014) Slashburn: graph compression and mining beyond caveman communities. IEEE Trans Knowl Data Eng 26(12):3077–3089
29. Mishra A, Bhattacharya A (2011) Finding the bias and prestige of nodes in networks based on trust scores. In: Proceedings of the 20th international conference on World Wide Web. ACM, pp 567–576
30. Ng AY, Zheng AX, Jordan MI (2001) Stable algorithms for link analysis. In: Proceedings of the 24th annual international ACM SIGIR conference on research and development in information retrieval. ACM, pp 258–266
31. Page L, Brin S, Motwani R, Winograd T (1999) The pagerank citation ranking: bringing order to the Web
32. Saad Y (2003) Iterative methods for sparse linear systems, vol 82. SIAM
33. Shahriari M, Jalili M (2014) Ranking nodes in signed social networks. Soc Netw Anal Min 4(1):1–12
34. Shin K, Jung J, Lee S, Kang U (2015) Bear: Block elimination approach for random walk with restart on large graphs. In: Proceedings of the 2015 ACM SIGMOD international conference on management of data. ACM, pp 1571–1585
35. Song D, Meyer DA (2015) Recommending positive links in signed social networks by optimizing a generalized auc. In: AAAI, pp 290–296
36. Strang G (2006) Linear algebra and its applications. Thomson, Brooks/Cole. https://books.google.ie/books?id=q9CaAAAACAAJ
37. Szell M, Lambiotte R, Thurner S (2010) Multirelational organization of large-scale social networks in an online world. Proc Nat Acad Sci 107(31):13636–13641
38. Taylor ME (2006) Measure theory and integration. American Mathematical Soc, Providence
39. Tong H, Faloutsos C, Gallagher B, Eliassi-Rad T (2007) Fast best-effort pattern matching in large attributed graphs. In: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp 737–746
40. Tong H, Faloutsos C, Pan J-Y (2008) Random walk with restart: fast solutions and applications. Knowl Inf Syst 14(3):327–346
41. Wu Z, Aggarwal CC, Sun J (2016) The troll-trust model for ranking in signed networks. In: Proceedings of the ninth ACM international conference on Web search and data mining. ACM, pp 447–456
42. Yang B, Cheung WK, Liu J (2007) Community mining from signed social networks. IEEE Trans Knowl Data Eng 19(10):1333–1348
43. Yoon M, Jin W, Kang U (2018) Fast and accurate random walk with restart on dynamic graphs with guarantees. In: Proceedings of the 2018 World Wide Web conference on World Wide Web, WWW 2018, Lyon, France, April 23–27, 2018, pp 409–418

44. Yoon M, Jung J, Kang U (2018) Tpa: Fast, scalable, and accurate method for approximate random walk with restart on billion scale graphs. In: 34th IEEE international conference on data engineering, ICDE 2018, Paris, France, April 16–19, 2018

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Jinhong Jung is a Ph.D. student in the Department of Computer Science and Engineering of Seoul National University. He received M.S. in the School of Computing at KAIST, after receiving B.S. in Computer Science and Engineering of Cheonbuk National University. His research interest includes large-scale graph mining and machine learning.

Woojeong Jin is a Ph.D. student in the Department of Computer Science at the University of Southern California. He received B.S. in Electrical and Computer Engineering at Seoul National University. His research interest lies in machine learning, graph mining, and natural language processing.

U Kang is an associate professor in the Department of Computer Science and Engineering of Seoul National University. He received Ph.D. in Computer Science at Carnegie Mellon University, after receiving B.S. in Computer Science and Engineering at Seoul National University. He won 2013 SIGKDD Doctoral Dissertation Award, 2013 New Faculty Award from Microsoft Research Asia, 2016 Korean Young Information Scientist Award, 2018 ICDM 10-year best paper award, and two best paper awards. He has published over 60 refereed articles in major data mining, database, and machine learning venues. He holds four US patents. His research interests include big data mining, deep learning, and machine learning.