Look Before You Leap: Confirming Edge Signs in Random Walk with Restart for Personalized Node Ranking in Signed Networks

Wonchang Lee†
Hanyang University
Seoul, Korea
wonchang24@hanyang.ac.kr

Yeon-Chang Lee∗
Hanyang University
Seoul, Korea
lyc0324@hanyang.ac.kr

Dongwon Lee
The Pennsylvania State University
University Park, PA, USA
dongwon@psu.edu

Sang-Wook Kim†
Hanyang University
Seoul, Korea
wook@hanyang.ac.kr

ABSTRACT
In this paper, we address the personalized node ranking (PNR) problem for signed networks, which aims to rank nodes in an order most relevant to a given seed node in a signed network. The recently-proposed PNR methods introduce the concept of the signed random surfer, denoted as SRSurfer, that performs the score propagation between nodes using the balance theory. However, in real settings of signed networks, edge relationships often do not strictly follow the rules of the balance theory. Therefore, SRSurfer-based PNR methods frequently perform incorrect score propagation to nodes, thereby degrading the accuracy of PNR. To address this limitation, we propose a novel random-walk based PNR approach with sign verification, named as OBOE (Look Before you Leap). Specifically, OBOE carefully verifies the score propagation of SRSurfer by using the topological features of nodes. Then, OBOE corrects all incorrect score propagation cases by exploiting the statistics of a given network. The experiments on 3 real-world signed networks show that OBOE consistently and significantly outperforms 5 competing methods with improvement up to 13%, 95%, and 249% in top-k PNR, bottom-k PNR, and troll identification tasks, respectively. All OBOE codes and datasets are available at: http://github.com/wonchang24/OBOE.

CCS CONCEPTS
• Information systems → Retrieval models and ranking.

KEYWORDS
personalized node ranking; signed networks; balance theory

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM Reference Format:

1 INTRODUCTION
Given a seed node in a network, the personalized node ranking (PNR) problem is to rank the remaining nodes in an order most relevant to the seed node by considering both the structure of the network and the connectivity with the seed node. Unlike the traditional node ranking problem [16, 23, 29], the PNR problem ranks nodes from the view point of a given seed node. In this sense, solutions [9, 15, 35, 40] to the PNR problem can be utilized in a variety of applications that need personalization such as friend recommendation and targeted marketing. Popular PNR methods include random-walk with restart (RWR) [34] and personalized SALSA (PSALSA) [1].

Recently, there has been a surge of interest on signed networks with both positive and negative edges between nodes [25, 38]. For example, sites such as Slashdot (a technology news site) or Epinions (a now-defunct consumer review site) allow users to decide whether users trust (i.e., positive edges) or distrust (i.e., negative edges) each other. In such a setting, the edge signs provide rich semantics between nodes [6, 10, 18, 20]. However, the aforementioned PNR methods were designed for (unsigned) networks with only positive edges between nodes. For this reason, many prior works have extended such PNR methods for signed networks—e.g., FriendTNS [33], OPT+GAUC [31], SRWR [13, 14], and SSRW [27].

In particular, two recently-proposed and also best-performing PNR methods, i.e., SRWR [13, 14] and SSRW [27], extend the existing RWR models by introducing the concept of the signed random surfer, denoted by SRSurfer in this paper. Basically, SRSurfer traverses the edges starting from a seed node $n_x$ in a signed network while propagating the positive/negative scores of $n_x$ to the visited nodes $n_y$ using the balance theory [2], a well-known theory in psychology. This theory states that social relationships follow four rules: (R1) a friend of my friend is my friend; (R2) a friend of my enemy is my enemy; (R3) an enemy of my friend is my enemy; (R4) an enemy of my enemy is my friend.

Specifically, SRSurfer predicts an edge sign between two nodes $n_x$, $n_y$ by analyzing the combination of edge signs along the random-walk path from $n_x$ to $n_y$ based on the balance theory. Then, if the predicted sign is positive (resp. negative), SRSurfer propagates the
positive (resp. negative) score of \( n_x \) to \( n_y \). As shown in Figure 1, when SRSurfer walks on two negative edges from \( n_x \) to \( n_y \), it propagates a positive score of \( n_x \) to \( n_y \) based on R4 of the balance theory. Here, we note that this type of score propagation is performed under the assumption that the decades-old balance theory always holds in real settings. However, in real settings of signed networks, edge relationships often do not strictly follow the rules of the balance theory [7]; this observation will be elaborated in Section 3.

In this sense, the edge signs predicted by SRSurfer may be inaccurate. For instance, in the above example, \( n_x \) and \( n_y \) may not be a friend because the enemies of my enemy are not always my friends in the real world. Furthermore, as the length of the path between two nodes increases (i.e., the path between \( n_x \) and \( n_z \) in Figure 1), the number of edge signs used for prediction also increases, rendering the accurate sign predictions more challenging. As a demonstration, Figure 2 shows that the prediction accuracy of SRSurfer using the Wikipedia dataset decreases rapidly as the path length increases. This result clearly indicates that SRSurfer fails to predict edge relationships accurately. In other words, SRSurfer-based PNR methods frequently perform incorrect score propagation to nodes, thereby degrading the accuracy of node rankings.

To address this limitation, in this paper, we aim to carefully verify the score propagation of SRSurfer and then correct incorrect score predictions. Toward this end, we propose a novel random-walk-based PNR approach with sign verification, named as OBOE (Look Before You Leap). Existing SRSurfer-based PNR methods perform the score propagation based on the balance theory as follows: if \( n_y \) and \( n_z \) are connected by a positive edge (resp. a negative edge), the methods propagate \( n_y \)'s positive/negative scores to \( n_z \)'s positive/negative (resp. negative/positive) scores, respectively. On the other hand, OBOE carefully validates whether such score propagation is trustworthy or not before propagating \( n_y \)'s scores to \( n_z \). To this end, OBOE first predicts a sign between the seed node \( n_x \) and \( n_z \) based on their 23 topological features such as degree distribution and triad types [24, 26]. Then, if the sign prediction turns out to be trustworthy, OBOE degenerates to a regular SRSurfer model and propagates scores based on the balance theory. Otherwise, OBOE uses a novel score propagation strategy that exploits the statistics related to the edge signs of all triangles in a given signed network. By iteratively performing the aforementioned process (i.e., verification and score propagation), OBOE propagates the positive/negative scores of the seed node \( n_x \) to all other nodes. Finally, after convergence, OBOE ranks all nodes except for \( n_x \) based on their positive and negative scores.

In designing, implementing, and validating these ideas, in this paper, our contributions are summarized as follows:

- We demonstrate the limitation of SRSurfer-based PNR methods in propagating scores of nodes based on the balance theory.
- We propose a novel random-walk based PNR approach with a sign verification, named as OBOE.
  - We design a strategy to validate the score propagation based on the balance theory by exploiting topological features of nodes.
  - We design a strategy to leverage the statistics of a given network to replace the untrustworthy score propagation.
- We validate the effectiveness of OBOE via extensive experiments using three real-world datasets.
  - Specifically, OBOE dramatically improves the accuracy of top-k PNR/bottom-k PNR/troll identification tasks up to 13%/95% /249%, respectively, over the best performer among the five competing methods.

The rest of this paper is organized as follows: Section 2 reviews existing PNR methods. Section 3 demonstrates the limitation of SRSurfer-based PNR methods and Section 4 presents our proposed approach in detail. Section 5 validates the effectiveness of the proposed approach through extensive experiments. Finally, Section 6 summarizes and concludes the paper.

2 RELATED WORK

In this section, we briefly review two families of existing PNR methods: one for unsigned networks and the other for signed networks.

PNR for Unsigned Networks. First, RWR [34], PSALSA [1], and MPR [8] analyze the structure of a given network by designing their own random-walk models. They traverse the edges starting from a seed node in the network while propagating the scores of the seed node to other nodes. Finally, they rank nodes, except for the seed node, based on their scores. Next, PALE [28] and RDL [36] represent the nodes in a given network as low-dimensional vectors by employing low-rank models. Then, they rank nodes, except for a seed node, based on a distance function that exploits the vector of the seed node and that of each remaining node. However, none of these methods took edge signs into consideration in their designs.

PNR for Signed Networks. To address this limitation, various PNR methods for signed networks have been proposed. In the early days of research, FriendTNS [33] and Zhu et al. [42] designed heuristic measures that calculate relevance scores between nodes in consideration of edge signs. Then, both methods rank the nodes based on the relevance scores between the seed node and remaining nodes. Next, Opt-GAUC [31] proposed a matrix factorization model that learns existent positive/negative edges between nodes and further learns non-existent edges as no-relation edges.

Recently, several studies such as ModifiedRWR [30], SRWR [13, 14], and SSRW [27] have extended the existing RWR models. First, ModifiedRWR [30] computes positive/negative RWR scores of nodes by performing RWR on the positive/negative subgraph, respectively, and then ranks nodes by subtracting negative RWR scores from positive ones. Next, SRWR [13, 14] and SSRW [27] compute positive/negative scores of nodes based on SRSurfer and then rank nodes in the same way as in ModifiedRWR. Specifically, SRWR introduced the balance attenuation factors \( \beta \) into SRSurfer to consider the uncertainty for two rules (i.e., R2 and R4) of the balance theory. However, the factors (1) apply to all (trustworthy and

\[\text{Figure 1: A signed network:}\]
dotted lines indicate the edges walked by SRSurfer.

\[\text{Figure 2: Prediction accuracy of SRSurfer per path length.}\]
whose \( y \) where \( r \) (resp. a negative edge), it propagates via (Eq. 1) until \( r \) converges. In summary, both SRSurfer-based PNR methods frequently perform incorrect score propagation to nodes, thereby adversely affecting the accuracy of PNR.

Lastly, we can perform PNR based on the network embedding methods for signed networks, denoted as signed NE [5, 21, 37, 39, 41]. Specifically, the signed NE methods represent the nodes of a given signed network by vectors in a low-dimensional embedding space so that the vectors preserve structural and semantic properties in the network. That is, they attempt to represent the nodes with positive edges to be close and those with the negative edges to be distant in the embedding space. Literature [5, 21, 37, 39, 41] has shown that the low-dimensional vectors can be used as effective features of nodes in solving various downstream tasks including node ranking and recommendation [3, 4, 19]. For instance, in terms of PNR, we rank the nodes using the vectors of the seed node and those of the remaining nodes.

3 MOTIVATION

In this section, we present SRSurfer [13, 14] in detail and demonstrate its limitation via experiments over real-world datasets.

SRSurfer predicts positive scores \( r^+_y \) and negative scores \( r^-_y \) of all nodes \( n_y \) from the perspective of a seed node \( n_x \). If \( x = y \), the initial values of \( r^+_y \) and \( r^-_y \) are set to 1 and 0, respectively. Otherwise, both are set to 0. Then, SRSurfer propagates \( r^+_y/r^-_y \) of \( n_x \) to all other nodes. Suppose that SRSurfer walks to a node \( n_z \) in the neighborhood of a node \( n_y \) with a probability of \((1 - c)\). In this case, SRSurfer propagates \( r^+_y/r^-_y \) of \( n_y \) to \( n_z \) based on the balance theory as follows: if \( n_y \) and \( n_z \) are connected by a positive edge (resp. a negative edge), it propagates \( r^+_y/r^-_y \) of \( n_y \) to \( r^+_z/r^-_z \) (resp. \( r^-_y/r^+_z \)) of \( n_z \). In addition, SRSurfer restarts at \( n_x \) with a probability \( c \) for personalization w.r.t \( n_x \). This probability \( c \) is empirically determined.

Given a seed node \( n_x \), the aforementioned score propagation can be formulated as follows [13, 14]:

\[
    r^+_x = (1 - c) (\tilde{A}^+)^T r^+ + (\tilde{A}^-)^T r^- + \text{eq.},
\]

\[
    r^-_x = (1 - c) (\tilde{A}^+)^T r^- + (\tilde{A}^-)^T r^+ .
\]

where \( r^+_x \) and \( r^-_x \) represent positive/negative score vectors w.r.t. \( n_x \) whose \( y \)-th elements are \( r^+_y \) and \( r^-_y \) of \( n_y \), respectively. \( \tilde{A}^+ \) and \( \tilde{A}^- \) indicate positive/negative semi-row normalized matrices that contain all positive/negative values in the adjacency matrix \( A \) of a given signed network, respectively. \( q \) is a unit vector whose \( x \)-th element (i.e., seed node \( n_x \)) is 1 and all other elements are 0. Finally, \( c \) is a restart probability. Then, SRSurfer iteratively updates \( r^+ \) and \( r^- \) via (Eq. 1) until \( r^+ \) and \( r^- \) converge. Finally, SRSurfer computes the ranking score vector \( r^d \) of \( n_x \) as follows:

\[
    r^d = r^+ - r^- .
\]

where \( r^d \) indicates a final ranking score for \( n_y \) w.r.t \( n_x \).

Note that the score propagation of SRSurfer totally depends on the predicted signs by the balance theory. As alluded in Section 1, however, edge relationships often do not strictly follow the rules of the balance theory. To show the evidence of this claim in real-world signed networks (i.e., Wikipedia, Slashdot, and Epinions), we first sample all the triangles \((n_x, n_y, n_z)\) where the edge directions between three nodes satisfy the transitivity. For instance, if \( n_x \) points to \( n_y \), \( n_y \) points to \( n_z \), and \( n_x \) points to \( n_z \), the triangle \((n_x, n_y, n_z)\) satisfies the transitivity. Then, we examine how much the triangles follow the rules of the balance theory. That is, given two signs (i.e., prior signs) in a triangle, we check whether the remaining sign (i.e., posterior sign) matches the rules of the balance theory (e.g., a balanced triangle in Figure 3-(a)) or not (e.g., an unbalanced triangle in Figure 3-(b)).

Table 1 shows the ratios of balanced and unbalanced triangles among all triangles in three signed networks, according to four types of prior signs − i.e., \((+, +)\), \((+, -)\), \((-,-)\), and \((-, +)\). We observe that \((+, +)\) type follows the rules of the balance theory considerably, whereas other types often do not follow the rules strictly. For instance, for \((+, -)\) type in Wikipedia or for \((-, +)\) type in Epinions networks, 72% and 59% of triangles do not follow the balance theory (i.e., unbalanced triangles), respectively. This result clearly demonstrates that the balance theory does not always hold in real settings. Therefore, score propagation models that strictly follows the balance theory (e.g., SRSurfer-based PNR methods) are likely to contain substantial amount of incorrect score propagation.

Next, further, we test how much one can improve the sign prediction accuracy by not strictly following the balance theory. To this end, we regard 80% of the edges in a signed network as a training set, and the remaining 20% as a test set. Then, we sample all unbalanced triangles \((n_x, n_y, n_z)\) from the training set. Next, we perform sign prediction on the test set using SRSurfer. Here, we made two variants, denoted as Original SRSurfer and Modified SRSurfer, of predicting the signs of posterior edges \((n_x, n_z)\) when SRSurfer walks on two prior edges \((n_x, n_y), (n_y, n_z)\) of a sampled unbalanced triangle \((n_x, n_y, n_z)\) in turn. Specifically, whenever two variants encounter two prior edges of the sampled unbalanced triangles during the random-walk process, the sign of each posterior edge is predicted differently as follows:

\[\text{Original SRSurfer:}\]

\[\text{Modified SRSurfer:}\]

\[\text{2}\]

Such predictions are difficult to implement in terms of matrix-vector multiplication as shown in (Eq. 1). Thus, we performed both original/modified SRSurfer in a random-walk manner.

![Figure 3: Examples of balanced/unbalanced triangles.](image-url)
The PNR problem for signed networks is formulated as follows: Let $G = (N, E^+, E^-)$ be a signed network, where $N = \{n_1, n_2, \cdots, n_m\}$ represents a set of $m$ nodes and $E^+$ and $E^-$ represent the sets of positive and negative edges, respectively. Note that $E^+ \cap E^- = \emptyset$, a node pair cannot have both positive and negative edges simultaneously. Given a seed node $n_x$, PNR methods for signed networks aim to output ranking of the nodes except for $n_x$ by analyzing both the structure of the network and their connectivity with $n_x$. Table 2 summarizes a list of notations used in this paper.

We present the overall procedure of OBOE. Given a seed node $n_x$, OBOE sets the positive/negative scores $r^+_y/r^-_y$ of all nodes $n_y$ from the perspective of $n_x$. $r^+_y$ and $r^-_y$ represents the likelihood (measured by OBOE) to which the edge relationship between $n_x$ and $n_y$ has a positive sign (i.e., friend) and a negative sign (i.e., enemy), respectively. If $x = y$, the initial values of $r^+_y$ and $r^-_y$ are set to 1 and 0, respectively. Otherwise, both are set to 0. Then, OBOE propagates $r^+_y/r^-_y$ to $n_x$ to all other nodes. Toward this end, the random surfer of OBOE start from $n_x$ and then walk along the outgoing edges with a probability of $(1 - c)$, while going back to $n_x$ with a probability of $c$.

Now, we describe the process of the score propagation of OBOE with Figure 5. As shown in Figure 5-(a), suppose that a random surfer is currently at a node $n_y$ and $n_y$ has already received the scores from the previously-visited node $n_x$ by OBOE’s score propagation strategy (to be explained later). When the surfer walks to a node $n_z$ in the neighborhood of $n_y$ with $(1 - c)$, OBOE propagates both $r^+_y$ (1) in Figure 5-(a) and $r^-_y$ (2) in Figure 5-(a) of $n_y$ to $n_z$. We here explain the propagations of $r^+_y$ since both propagation of $r^+_y$ and $r^-_y$ are similar. Basically, if an edge sign, $\text{sign}(n_y, n_z)$ (3 in Figure 5-(a)), between $n_y$ and $n_z$ is positive (resp. negative), OBOE propagates $r^+_y$ to $r^+_z$ (resp. $r^-_z$) in the same way as SRSurfer. As demonstrated in Section 3, however, score propagation models based on SRSurfer are likely to contain a substantial amount of incorrect score propagation (Table 1).

To validate the score propagation from $n_y$ to $n_z$ based on the balance theory, OBOE predicts an edge sign, $\text{sign}_\text{feature}(n_x, n_z) \in \{+1, -1\}$ (4 in Figure 5-(b)), between $n_y$ and $n_z$ with additional information. Toward this end, OBOE exploits 23 topological features of $n_y$ and $n_z$. Then, OBOE examines whether both $\text{sign}_\text{feature}(n_x, n_z)$ and the sign of $n_z$’s score, to which $r^+_y$ is propagated by SRSurfer,
We describe the key ideas of OBOE \( r_f \). In particular, OBOE regards the score propagation as trustworthy and thus performs that propagation (the upper case in Figure 5-(d)). Otherwise, OBOE regards it as untrustworthy and uses a novel score propagation (the lower case in Figure 5-(d)). Specifically, OBOE propagates \( r_f \) to both scores \( r_f \) and \( r_f \), not a score determined by SRSurfer. The intuition behind this idea is that the edge relationship between \( n_x \) and \( n_y \) are uncertain. Therefore, OBOE should set the ratios at which \( r_f \) is propagated to \( r_f \) and \( r_f \). In this paper, we utilize the ratios of the prior and posterior signs of triangles in a given signed network, to be elaborated in Section 4.2. OBOE also performs the same process as above for the propagation of \( r_f \).

So far, we have presented a situation in which OBOE propagates \( r_f \) of a node \( n_x \) to \( r_f \) of another node \( n_y \). In OBOE at every iteration, each node propagates its own scores to its outgoing edges and receives the propagation from its incoming edges. OBOE iteratively performs this score propagation until the scores of all nodes converge. Finally, after convergence, OBOE ranks nodes except for \( n_x \) by subtracting their negative scores from their positive ones.

### 4.2 Sign Verification and Score Propagation

We describe the key ideas of OBOE in detail. Given a seed node \( n_x \), a currently-visiting node \( n_y \), and a next-visiting node \( n_z \), OBOE performs the following two steps: (STEP 1) sign verification and (STEP 2) score propagation. OBOE first verifies the score propagation from \( n_x \) to \( n_y \) by directly exploiting the relationship between \( n_x \) and \( n_y \) and then classify it into trustworthy or untrustworthy one. Next, OBOE performs the trustworthy one as it is, while it corrects the untrustworthy one by performing a safe score propagation exploiting the statistics of a given network.

**STEP 1: Sign Verification.** In this step, OBOE predicts a \( \text{sign feature}(n_x, n_z) \) between \( n_x \) and \( n_z \) based on their topological features and validates whether the score propagation from \( n_y \) to \( n_z \) is trustworthy or not by using \( \text{sign feature}(n_x, n_z) \).

For sign prediction, we employ a well-known feature-based method, fEXtra [24, 26]. For all node pairs \( (n_x, n_y) \) in a network, we first construct a vector \( f_{xy} = (f_1, \cdots, f_3) \) consisting of values for 23 topological features for each of the pairs. Specifically, we first exploit 7 features related to nodes’ degrees: (1) the number of outgoing positive edges of \( n_x \), (2) the number of outgoing negative edges of \( n_x \), (3) the number of incoming positive edges of \( n_y \), (4) the number of incoming negative edges of \( n_y \), (5) the total number of common neighbors of \( n_x \) and \( n_y \), (6) the total out-degree of \( n_x \), and (7) the total in-degree of \( n_y \). In addition, we exploit features related to 16 distinct types of triads containing \( n_x \), \( n_y \), and their common neighbors \( n_z \) in the network: the edge between \( n_x \) and \( n_z \) can be in either direction and of either sign, and the edge between \( n_y \) and \( n_z \) can also be in either direction and of either sign; this leads to \( 2 \times 2 \times 2 \times 2 \times 2 = 16 \) possibilities. Thus, we set the number of triads of each type to the value of the corresponding feature.

Next, we train a logistic regression classifier model based on the \( f_{xy} \) for node pairs \( (n_x, n_y) \) with real edge signs in the network. Using the learned classifier, we finally predict the \( \text{sign feature}(n_x, n_z) \) for all node pairs \( (n_x, n_z) \) without real edge signs in the network. During the prediction process, fEXtra also computes the confidence score \( C(\text{sign feature}(n_x, n_z)) \in [0, 1] \) for the predicted sign

\( \text{sign feature}(n_x, n_z) \). For instance, if \( \text{sign feature}(n_x, n_z) = +1 \) and \( C(\text{sign feature}(n_x, n_z)) = 0.95 \), it indicates fEXtra predicted the sign between \( n_x \) and \( n_z \) as positive and is quite confident for that prediction. Note that, before OBOE performs the random-walk process, it predicts \( \text{sign feature}(n_x, n_z) \) for all node pairs \( (n_x, n_z) \) without real edge signs in advance as a preprocessing task.

Now, OBOE verifies the score propagation from \( n_y \) to \( n_z \) using the \( \text{sign feature}(n_x, n_z) \) and the \( C(\text{sign feature}(n_x, n_z)) \). Note that we exploit the relationship between the seed node \( n_x \) (not \( n_y \)) and \( n_z \) as OBOE is propagating the scores of \( n_y \) received from \( n_x \) to \( n_z \).

Here, there are four possible score propagations (SP) from \( n_y \) to \( n_z \) as follows (Figure 6):

1. If \( \text{sign}(n_x, n_z) = +1 \), \( r_f \) is propagated to \( r_f \) (SP 1 in Figure 6-(a)) and \( r_f \) is propagated to \( r_f \) (SP 2 in Figure 6-(b)).
2. If \( \text{sign}(n_x, n_z) = -1 \), \( r_f \) is propagated to \( r_f \) (SP 3 in Figure 6-(c)) and \( r_f \) is propagated to \( r_f \) (SP 4 in Figure 6-(d)).

To validate the above score propagations, OBOE checks the following two conditions: (1) does the \( \text{sign feature}(n_x, n_z) \) and the sign of \( n_z \)’s score, to which \( n_y \)’s score is propagated, consistent? and (2) does the \( C(\text{sign feature}(n_x, n_z)) \) exceed a predefined threshold? For (1), OBOE considers the score propagation in which the two signs are consistent as trustworthy. For instance, if \( \text{sign feature}(n_x, n_z) = +1 \) and \( n_y \)’s score is propagated to \( r_f \) (resp. \( r_f \) as trustworthy) respectively.

For (2), OBOE finally uses only the score propagation considered trustworthy by the confident \( \text{sign feature}(n_x, n_z) \). In the above example, if \( C(\text{sign feature}(n_x, n_z)) = +1 \) is lower than the predefined threshold, OBOE does not regard the SP 1/4 as trustworthy. In this paper, we introduce a threshold \( \beta_+ \) for \( C(\text{sign feature}(n_x, n_z)) = +1 \) and another threshold \( \beta_- \) for \( C(\text{sign feature}(n_x, n_z)) = -1 \) separately. We will analyze the sensitivity of OBOE to \( \beta_+ \) and \( \beta_- \) in Section 5.2.

Here, there remains a question on whether the \( \text{sign feature}(n_x, n_z) \) is more reliable than the sign predicted by SRSurfer. To address this concern, we confirmed that fEXtra consistently and significantly improves the accuracy of sign prediction based on SRSurfer in our preliminary experiments. Thanks to (STEP 1) of OBOE, we can detect untrustworthy score propagation. In Section 5.2, we confirm that such untrustworthy score propagation occurs considerably but is detected via our OBOE.

**STEP 2: Score Propagation.** In this step, OBOE performs different score propagation from \( n_y \) to \( n_z \), according to trustworthy or untrustworthy one. OBOE first normalizes the \( r_f \) of \( n_y \) to the number of \( n_y \)’s outgoing edges \( |O_y| \) as follows:

\[
r_f = \frac{r_f}{|O_y|}, \quad r_f = \frac{r_f}{|O_y|}
\]

\( ^{1} \)We can handle the case of \( \text{sign feature}(n_x, n_z) = -1 \) as well in the same way.
Again, note that $r_w^+$ and $r_w^−$ represents the degree to which the edge relationship between $n_x$ and $n_y$ has a positive sign and a negative sign, respectively. Then, OBOE propagates $n_y$’s $r_w^+/r_w^−$ to $n_x$ by considering the following eight cases: (1) CASE 1/5 regard SP 1 as trustworthy/untrustworthy, respectively; (2) CASE 2/6 regard SP 2 as trustworthy/untrustworthy, respectively; (3) CASE 3/7 regard SP 3 as trustworthy/untrustworthy, respectively; (4) CASE 4/8 regard SP 4 as trustworthy/untrustworthy, respectively.

For CASE 1/2/3/4 (i.e., trustworthy score propagation), OBOE performs each score propagation as it is. On the other hand, for CASE 5/6/7/8 (i.e., untrustworthy score propagation), we assume that the edge relationship between $n_x$ and $n_y$ are uncertain. Therefore, OBOE uses a different score propagation strategy, instead of each (untrustworthy) score propagation. For CASE 5/7 (i.e., for propagating $r_w^+$ to $r_z$), OBOE propagates $r_w^+$ to both $r_z^+$ and $r_z^−$ while, for CASE 6/8 (i.e., for propagating $r_w^−$ to $r_z$), OBOE propagates $r_w^−$ to both $r_z^+$ and $r_z^−$.

For determining ratios of the propagation, as a heuristic policy, OBOE utilizes the ratios (i.e., Table 1 in Section 3) of the prior and posterior signs of the (real) triangles in a given signed network. Specifically, OBOE regards (1) the sign of $n_y$’s score (i.e., positive and negative one for CASE 5/7 and CASE 6/8, respectively) and (2) sign ($n_y$, $n_z$) (i.e., positive and negative one for CASE 5/6 and CASE 7/8, respectively) as two prior signs. Also, OBOE regards the signs of $n_z$’s score received from $n_y$ as the posterior signs. Given two prior signs in an input network, OBOE then checks the ratios at which the posterior signs are positive and negative. Finally, OBOE propagates $n_y$’s scores to both $r_z^+/r_z^−$ in accordance with the observed ratio of positive/negative signs, respectively.

Figure 7 depicts the cases in which $n_y$’s scores are propagated to $n_z$ in the Wikipedia dataset. Each case is divided into two sub-cases according to the posterior signs. Suppose that both prior signs are given as positive as shown in Figures 7-(a) and 7-(b). In this case, from Table 1, OBOE checks that the ratios of posterior signs to positive (i.e., +) and negative (i.e., −) are 92% and 8% in a Wikipedia dataset, respectively. Then, OBOE propagates 92% of $r_z^+$ to $r_z^/r_z^−$, respectively. Similar to the sign predictions in (STEP 1), in general, OBOE pre-computes the statistics for a given signed network as part of the pre-processing task.

Thanks to (STEP 2) of OBOE, we can now correct the incorrect score propagation by referring to the statistics of a given network.

![Figure 7: Score propagation strategy of OBOE (Wikipedia).](image)

By employing this heuristic policy, we can consider the structural property per given signed network. In Section 5.2, we show that our score propagation strategy with this policy to replace the untrustworthy score propagation is more effective than other strategies (e.g., uniform propagation).

### 4.3 Formulation for OBOE

We formulate the iteration of OBOE as a matrix-vector multiplication form. Toward this end, we extend (Eq. 1) of SRSurfer as follows:

$$
r^*_t = (1 - \epsilon) \left\{ \left( (\tilde{A}^{+})^\top (\tilde{A}^{+}) + (\tilde{A}^{-})^\top (\tilde{A}^{-}) \right) r^+ + \left( (\tilde{A}^{+})^\top (\tilde{A}^{-}) + (\tilde{A}^{-})^\top (\tilde{A}^{+}) \right) r^- \right\} + \epsilon q \cdot r^+.
$$

where $r^+/r^−$ represent positive/negative score vectors w.r.t. $n_x$, $q$ is an unit vector whose $x$-th element (i.e., seed node $n_x$) is 1 and all other elements are 0. Also, $\tilde{A}^{+}$ and $\tilde{A}^{-}$ indicate adjacency matrices for trustworthy and untrustworthy score propagations, respectively.

Note that the matrices of $\tilde{A}^{+}$ (i.e., $\tilde{A}^{+}$) have two signs, while the matrices of $\tilde{A}^{-}$ (i.e., $\tilde{A}^{-}$) have three signs. Specifically, in each of $\tilde{A}^{+}$, both signs represent two prior signs. For instance, $\tilde{A}^{+}$ represents a situation in which $r_w^+$ (i.e., the first prior sign) of each node $n_y$ is propagated to each node $n_z$ whose sign($n_y$, $n_z$) $= +1$ (i.e., the second prior sign) among the neighborhood of $n_y$ (i.e., CASE 3).

Next, in each of $\tilde{A}^{-}$, the first two signs are the same as those of $\tilde{A}^{+}$ and the last sign is a posterior sign. For instance, $\tilde{A}^{-}$ represents a situation where part of $r_w^+$ (i.e., the first prior sign) of each node $n_y$ is propagated to $r_z^+$ (i.e., the posterior sign) of each node $n_z$ whose sign($n_y$, $n_z$) $= -1$ (i.e., the second prior sign) among the neighborhood of $n_y$ (i.e., CASE 7-1). To build 12 adjacency matrices in (Eq. 4), we use the positive and negative semi-row normalized matrices $\tilde{A}^{+}, \tilde{A}^{-}$. Specifically, we first construct 6 (sub) adjacency matrices based on $\tilde{A}^{+}$: $\tilde{A}^{+}$ (CASE 1), $\tilde{A}^{+}$ (CASE 2), $\tilde{A}^{+}$ (CASE 5-1), $\tilde{A}^{+}$ (CASE 5-2), $\tilde{A}^{+}$ (CASE 6-1), and $\tilde{A}^{+}$ (CASE 6-2). In addition, we construct 6 remaining (sub) adjacency matrices based on $\tilde{A}^{-}$: $\tilde{A}^{-}$ (CASE 3), $\tilde{A}^{-}$ (CASE 4), $\tilde{A}^{-}$ (CASE 7-1), $\tilde{A}^{-}$ (CASE 7-2), $\tilde{A}^{-}$ (CASE 8-1), and $\tilde{A}^{-}$ (CASE 8-2).

Finally, Algorithm 1 sketches the iterative process of OBOE. Given a seed node $n_x$, OBOE constructs 12 (sub) adjacency matrices using $\tilde{A}^{+}$ and $\tilde{A}^{-}$ (line 1). Then, OBOE sets $q$ to $x$-th unit vector.
We show that the iteration in Algorithm 1 converges to the solution of a linear system as follows:

\[ \text{Algorithm 1} \]

\[ h^{(k)} = c(1 - (1 - c) \tilde{B})^{-1}q_k + c(1 - (1 - c) \tilde{B})^{-1}q_k \]

where \( \tilde{B} = [A_{+} + A_{+} + A_{+} + A_{+} - A_{-} + A_{-} - A_{-} + A_{-}] \),

\[ \text{Proof.} \quad \text{(Eq. 4) is represented as follows:} \]

\[ \begin{bmatrix} r' \\ r'' \end{bmatrix} = (1 - c) \tilde{B}^{k+1} - (1 - c) \tilde{B}^{k} + c \approx (1 - c) \tilde{B}^{k} + c \] \( \neq \text{h} = (1 - (c) \tilde{B})^{-1}q_k \).

Therefore, the iteration in Algorithm 1 is written as follows:

\[ h^{(k)} = (1 - (1 - c) \tilde{B})^{-1} + c(1 - (1 - c) \tilde{B})^{-1}q_k \]

where \( \tilde{B} = [A_{+} + A_{+} + A_{+} + A_{+} - A_{-} + A_{-} - A_{-} + A_{-}] \).

\[ \text{The spectral radius } \rho = \text{(1 - c) B} \text{ is less than 1 when 0 < c < 1 since } \text{B} \text{ is a column stochastic matrix and its largest eigenvalue is 1 [32].} \]

\[ \text{Therefore, converges as:} \]

\[ \lim_{k \to \infty} (1 - (1 - c) \tilde{B})^{k} = 0 \] and \( \lim_{k \to \infty} h^{(k)} = (1 - (1 - c) \tilde{B})^{-1}q_k \).

5 EVALUATION

In this section, we validate the effectiveness of our approach via extensive experiments. We designed our experiments, aiming at answering the following evaluation questions (EQs):

- EQ1: Do our score propagation strategy help PNR?
- EQ2: Do OBOE provide a more accurate top-k PNR than competing methods?
- EQ3: Does OBOE provide a more accurate bottom-k PNR than competing methods?
- EQ4: Does OBOE identify trolls more accurately than competing methods?
- EQ5: How sensitive is the accuracy of OBOE per different values of parameters?

5.1 Experimental Settings

**Datasets.** Following [13, 14], we used three real-world signed network datasets: Wikipedia, Slashdot, and Epinions. The datasets are all publicly available.

**Evaluation.** We designed our experiments, aiming at answering the following key evaluation questions (EQs):

- **EQ1:** Does our score propagation strategy help PNR?
- **EQ2:** Do OBOE provide a more accurate top-k PNR than competing methods?
- **EQ3:** Does OBOE provide a more accurate bottom-k PNR than competing methods?
- **EQ4:** Does OBOE identify trolls more accurately than competing methods?
- **EQ5:** How sensitive is the accuracy of OBOE per different values of parameters?
of each task. Toward this end, we use the following two metric popularly employed in other PNR research [13, 14, 27]: F1 score and normalized discounted cumulative gain (NDCG) [11, 12]. For troll identification task, following [13, 14], we use 96 enemies of a user, called No-More-Trolls, in Slashdot as trolls. Specifically, the user is an administrative account created for the purpose of collecting a troll list [17]. Note that such a list exists only in the Slashdot. We evaluate how much the bottom-\(k\) PNRs obtained by each method contain the trolls by using F1 score and NDCG.

5.2 Results

Due to space limitation, we omit some experimental results in this paper. The details for all experiments are available at http://sites.google.com/view/oobe-sigir21.

EQ1: Effectiveness of Our Score Propagation Strategy. To verify the effectiveness of our key ideas (i.e., sign verification and score propagation), we conduct experiments to answer the following two sub-questions:

- **EQ1-1 (Sign Verification):** How much of the balance theory based score propagation does OBOE detect as the untrustworthy score propagation?
- **EQ1-2 (Score Propagation):** Is it effective to exploit the statistics of a given network for correcting score propagation?

For EQ1-1, we examine the ratio of untrustworthy score propagation detected during the random-walk process of OBOE in each dataset. Figure 8 shows the results. The \(x\)-axis represents path lengths and the \(y\)-axis represents the ratio of untrustworthy score propagation per path length. Overall, we see that the ratio increases as the path length increases. We note, as shown in Figure 2, the prediction accuracy of SRSurfer decreases as the path length increases. Thus, we can say that OBOE detects untrustworthy score propagations more in the situation where the prediction accuracy of SRSurfer increases. Therefore, we can say that the prediction accuracy of SRSurfer decreases as the path length increases. We note, as shown in Figure 2, the prediction accuracy of SRSurfer decreases as the path length increases. Therefore, we can say that OBOE detects untrustworthy score propagations more in the situation where the prediction accuracy of SRSurfer increases. Thus, we can say that OBOE detects untrustworthy score propagations more in the situation where the prediction accuracy of SRSurfer increases.

Next, for EQ1-2, we validate the new score propagation of OBOE. Note that when OBOE encounters the untrustworthy score propagation, it propagates to both scores of two signs by leveraging the ratios of the prior and posterior signs of the triangles in a given signed network (STEP 2 in Section 4.2). To verify this strategy, we first made two variants of OBOE, denoted as OBOE(Balance) and OBOE(FExtra), which propagate only to a score of a single sign predicted by the balance theory and FExtra, respectively. Here, OBOE(Balance) coincides with SRWR that does not use the balance attenuation factors. Also, we made two variants of OBOE, denoted as OBOE(Uniform) and OBOE(Reverse), which propagate to both scores at different ratios from our strategy. Specifically, OBOE(Uniform) uniformly propagates to scores of both signs as the ratios of 50:50, while OBOE(Reverse) propagates to those by using the inverse of ratios used in our strategy. For comparison, we denote a variant using our strategy as OBOE(Ours).

Table 4 shows the accuracies of the variants of OBOE for top-\(k\) and bottom-\(k\) PNR tasks on Slashdot. First, among four variants except for OBOE(Ours), we found that OBOE(Uniform) showed the best accuracy except for NDCG@10 in the top-\(k\) task. The results represent that, for untrustworthy score propagation, propagating to both scores helps to improve the accuracy of PNR tasks than propagating to a single score. However, we observe that OBOE(Ours) consistently outperforms all other variants in all tasks. Specifically, OBOE(Ours) improves F1@10 of OBOE(Balance), OBOE(FExtra), OBOE(Uniform), and OBOE(Reverse) up to 15.7%/93.8%, 165.4%/39.7%, 14.7%/3.2%, and 33.4%/113.4% for top-\(k\)/bottom-\(k\) PNR tasks, respectively. This indicates that, when propagating to both scores, it is most effective to consider the inherent property of a given network.

**Table 4: Accuracies of the variants of OBOE according to different score propagation strategies (Slashdot)**

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Top-(k) PNR</th>
<th>Bottom-(k) PNR</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>F1@10</td>
<td>NDCG@10</td>
</tr>
<tr>
<td>OBOE(Balance)</td>
<td>0.027</td>
<td>0.102</td>
</tr>
<tr>
<td>OBOE(FExtra)</td>
<td>0.012</td>
<td>0.016</td>
</tr>
<tr>
<td>OBOE(Uniform)</td>
<td>0.028</td>
<td>0.023</td>
</tr>
<tr>
<td>OBOE(Reverse)</td>
<td>0.024</td>
<td>0.021</td>
</tr>
<tr>
<td>OBOE(Ours)</td>
<td>0.032</td>
<td>0.024</td>
</tr>
</tbody>
</table>

**Figure 8: Ratios of untrustworthy score propagation.**
follow the rules strictly. In this sense, SRSurfer-based PNR methods frequently perform inaccurate predictions for negative relationships, while OBOE effectively corrects such predictions, thereby improving the accuracy of bottom-k PNR considerably. Lastly, note the OBOE is relatively less effective on Epinions than other datasets. As mentioned in EQ1, on Epinions, 90% of all triangles are of the “balanced” (+, +) type, thus leaving little room for OBOE to exploit new ideas (so only small gain (%) compared to the results of SRWR).

**EQ4. Accuracy Comparison in Troll Identification Task.** We verify that OBOE outperforms the competing methods in a troll identification task. Note that the bottom-k task aims to evaluate how much accurately each method finds the real negative edges of each seed node (i.e., personalized ranking), while the troll identification task aims to evaluate how much accurately each method finds the trolls regardless of a specific seed node (i.e., global ranking). Table 6 shows the results. We observe that OBOE consistently and remarkably outperforms all competing methods. More specifically, OBOE dramatically improves the F1@100 and NDCG@10 of SRWR, the best performer among the competing methods, up to 239.0% and 249.2%, respectively. The results indicate that correcting incorrect score propagation contributes to accurately measuring not only PNR of each node but also global ranking as well.

**EQ5. Parameter Analysis for OBOE.** We carefully analyze the sensitivity of accuracy according to different values for parameters, $\beta_+$ (i.e., $C(\text{sign feature} = +1)$) and $\beta_-$ (i.e., $C(\text{sign feature} = -1)$). Figure 9 shows the accuracy changes of OBOE for Slashdot with varying $\beta_+$ and $\beta_-$. In the bottom-k PNR task. The x-axis and y-axis represent the values of $\beta_+$ and those of $\beta_-$, respectively, and the z-axis represents the accuracies in terms of NDCG@10. In general, as $\beta_+$ and $\beta_-$ increase, the accuracies of OBOE increase as well.

Also, the result shows that the accuracy is the highest when $\beta_+$ and $\beta_-$ are 0.9 and 0.9, respectively. This indicates that we should carefully use the FExtra-based predictions for the sign verification task, only when the predictions by FExtra are sufficiently confident.

**6 CONCLUSIONS**

In this paper, we investigated the limitation of SRSurfer-based PNR methods based on the balance theory: (1) edge relationships often do not strictly follow the rules of the balance theory and (2) the balance theory results in incorrect sign predictions of SRSurfer. To address this limitation, we proposed a novel random-walk based PNR approach, named as OBOE. OBOE is composed of (1) a sign verification based on nodes’ topological features and a score propagation based on the statistics of a given signed network. In addition, we formulated the iteration of OBOE as a matrix-vector multiplication form. Furthermore, we analyzed that the iterative algorithm of OBOE converges to a unique solution. Through comprehensive experiments using three real-world datasets, we demonstrated that (1) our sign verification and score propagation strategies are effective and (2) OBOE consistently and significantly outperforms all competing methods in three types of tasks, i.e., top-k PNR, bottom-k PNR, and troll identification, with three real-world datasets.

**ACKNOWLEDGMENT**

The work of Sang-Wook Kim was supported by Samsung Research Funding & Incubation Center of Samsung Electronics under Project Number SRFC-IT1901-03. The work of Dongwoon Lee was in part supported by NSF awards #1742702, #1820609, #1909702, #1915801, and #1934782.
REFERENCES


