CE7490 Project: Benchmarking Algorithms for Weight Prediction in Weighted Signed Networks

Ruihang Wang
School of Computer Science Engineering
Nanyang Technological University
`ruihang001@e.ntu.edu.sg`

Meng Shen
School of Computer Science Engineering
Nanyang Technological University
`meng005@e.ntu.edu.sg`

Yihang Li
School of Electrical and Electronic Engineering
Nanyang Technological University
`leah9704@gmail.com`

Abstract

The abstract paragraph should be indented ½ inch (3 picas) on both the left- and right-hand margins. Use 10 point type, with a vertical spacing (leading) of 11 points. The word **Abstract** must be centered, bold, and in point size 12. Two line spaces precede the abstract. The abstract must be limited to one paragraph.

1 Introduction

[Context] A number of weighted signed networks (WSN) exist in our world while many of them are incomplete. The values for weights prediction are ...

[Existing Solutions] State-of-the-art fairness and goodness. Baselines: Reciprocal, Triadic balance, Triadic Status, Pagerank ....

[Our project] To better understand theories and master practical skills, we investigate and implement a common set of algorithms and evaluate their performance on real-world datasets.

The rest of the paper is structured as follows: Section 2 presents related work in this topic. Section 3 describes the overview of our project. Section 4 formulates the problem of weight prediction in weighted signed networks. Section 5 conducted experiments on real-world datasets using different algorithms. Section 6 evaluates the performance of all tested methods. The conclusion is summarized in Section 7.

2 Related Work

Edge Sign Prediction in SSNs ...

Edge Weight Prediction in Social Networks ...

3 Project Overview

In this project, we extensively investigate and experiment methods for edge weight prediction in weighted signed networks. All algorithms are tested and evaluated on published real-world datasets. Moreover, we try our best to improve the performance on some traditional methods. The finished works are summarized as follows:

1. Literature review on OSN and select a topic about predicting weight of edges for weighted signed network.
2. Investigate the state-of-the-art algorithms *fairness-goodness* in [ ] and studied a common set of baselines for weight prediction.
3. Conducted experiments on each algorithm and reproduce the results mentioned in original paper using real-world dataset.
   Experimental 1 - Removing one edge prediction : ...
   Experimental 2 - Removing N %-out edge prediction : ...
4. Evaluate results of different methods.

4 Problem formulation

5 Experiments implementation

5.1 Reciprocal

As a baseline of weight prediction, reciprocal algorithm takes the weight of (v, u) as equal to the weight of (u, v). When the reciprocal edge doesn't exist, the weight of that edge is set to 0. Therefore, the weight of (u, v) is predicted by:

$$W(u,v) = \begin{cases} W(v,u), & \text{if } u \to v \text{ exist} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

5.2 Triadic balance

This definition is derived directly from balance theory according to . The algorithm takes the average product of edge weights for all incomplete triads that the edge (u, v) is a part of. Incomplete triads are triads that would form involving edge (u, v) after it is created. To be specific, let U_n and V_n denotes the set of neighbors of vertex u and v, repectively. To find all possible triads, vertexes with both connections of u and v are obtained by $N = U_n \cap V_n$. When $N = \emptyset$ the weight of (u, v) is set to 0. Then, the weight of (u, v) is predicted by:

$$W(u,v) = \begin{cases} \frac{\sum_{n \in N} W(u,n)+W(n,u)+W(v,n)+W(n,v)}{M}, & \text{if } N \neq \emptyset \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where M is number of vertexes in set N, $W(u, n), W(n, u), W(v, n), W(n, v)$ are weights of each edge connected to u or v of vertexes in set N, repectively.

5.3 Status Theory

Status theory is derived from . The prediction made by this measure is the difference between the status of vertex u and vertex v. The status $\sigma(i)$ of a vertex i is defined as $\sigma(i) = |W^+_{in}(i)| - |W^-_{in}(i)| + |W^-_{out}(i)| - |W^+_{out}(i)|$. Status increases when receiving positive incoming edges and generating negative outgoing edges to other vertices, while decreases when receiving negative edges and generating outgoing positive edges. Difference in status measures how much higher us status is compared to vs.

5.4 Bias and Deserve

This method is proposed by Mishra and Bhattacharya in . To compute "bias" and "deserve", we should first normalize the ratings (weights), and keep them in the range of [-1, 1] where 0 is a neutral opinion. Then, we say node u gives a trust-score of w_ij to node v for a given rating of (u, v). The two attributes of a node are defined by:

- *Bias*: This reflects the expected weight of an outgoing edge.

- *Deserve*: This reflects the expected weight of an incoming edge from an unbiased vertex.

Let $d^o(u)$ denotes the set of all outgoing edges from vertex u and likewise, $d^i(u)$ denotes the set of all incoming links to node u. Then, bias (BIAS) and deserve (DES) are iteratively computed as:

$$BIAS^{(t+1)}(u) = \frac{1}{2|d^o(u)|} \sum_{v \in d^o(u)} [W(u,v) - DES^t(v)] \tag{3}$$

$$DES^{(t+1)}(u) = \frac{1}{2|d^i(u)|} \sum_{v \in d^i(u)} [W(v,u)(1 - X^t(v,u))] \tag{4}$$

where $X^t(v,u) = max\{0, BIAS^t(v) \times W(v,u)\}$. The interative formulations of bias and deserve allow us to predict the weight of (u,v) based on the deserve value $DES(v)$ of vertex v. Thus, the weight is directly predicted by:

$$W(u,v) = DES(v) \tag{5}$$

5.5 Page Rank

We first compute Page Rank (ωPR) value of every vertex(k) in an unsigned graph according to **?**:

$$\omega PR(k) = \frac{1-\alpha}{|V|} + \alpha \sum_{z \in in(k)} \frac{\omega PR(z) * W(z,k)}{|out(z)|} \tag{6}$$

α is the probability called damping factor meaning a person will continue clicking on links(travelling in graph), and we set α to 0.85. $|V|$ is the number of all nodes. Eventually, $\omega PR(k)$ will converge to the probability that a person will go and stay at node k.

Then we compute edge weight of (u,v) using weighted average of PageRank values:

$$W(u,v) = \frac{\sum_{z \in out(u)} \omega PR(z) * W(u,z) + \sum_{z \in in(v)} \omega PR(z) * W(z,v)}{\sum_{z \in out(u)} \omega PR(z) + \sum_{z \in in(v)} \omega PR(z)} \tag{7}$$

5.6 Signed-Hits

The prediction is computed by using a modified version of HITS for signed network, called Signed-HITS**?**. Signed-HITS will compute the hub and authority scores of every node separately on positive graph(all edges' weights are positive) and negative graph, using the euqation:

$$\begin{aligned} h^+(u) &= \sum_{v \in out^+(v)} a^+(v); a^+(u) = \sum_{v \in ^+(u)} h^+(v) \\ h^-(u) &= \sum_{v \in out^-(v)} a^-(v); a^-(u) = \sum_{v \in ^-(u)} h^-(v) \end{aligned} \tag{8}$$

after convergence, we assign the authority score $a(u) = a^+(u) - a^-(u)$ and hub score $h(u) = h^+(u) - h^-(u)$ to each vertex u. Authority scores estimate the node value based on the incoming links, while hub scores estimate the node value based on outgoing links. However, the paper**?** didn't deliver the method to compute edge weight by using the authority score and hub score. Again, we use weighted average to compute the weight prediction:

$$W(u,v) = \frac{h(u) * \sum_{z \in out(u)} W(u,z) + a(v) * \sum_{z \in in(v)} W(z,v)}{h(u) * |V_{z \in out(u)}| + a(v) * |V_{z \in in(v)}|} \tag{9}$$

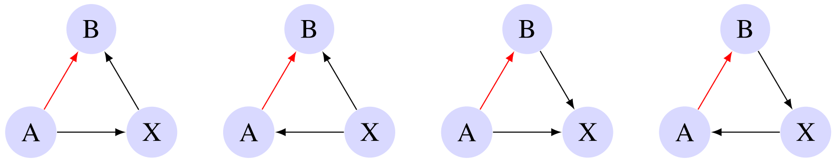
In this equation, we use $h(u)$ to be the weight of all out-going edges and $a(v)$ to be the weight of all in-coming edges. The reason is that $h(u)$ represents the node value based on out-going edges. So the predicted weight of out-edges from u should be higher if $h(u)$ is high. Likewise, the predicted weight of in-edges into v should be higher if $a(v)$ is high.

«««< HEAD

=======

5.7 Triadic Status

Imaging there are two nodes named A and B, and they are unlinked. What we want to do is to predict the weight of the edge from A to B. Suppose that there is another node X, and (A, B, X) compose a triangle once A and B are linked. Overall, there are four different triangle-links, as the edge between X and A can go in either direction and similarly for the edge between X and B, if the sign of weight is not considered.



(a) t1 (b) t2 (c) t3 (d) t4

Figure 1: Triadic Status

And we predict the weight of the edge (A, B) using three rules:

1. In Graph t1, the predicted weight of the transitive edge is the sum of the two non-transitive edge weights.
2. In Graph t2 and t3, the predicted weight of a non-transitive edge is the difference of the transitive and the other non-transitive edge weights.
3. In Graph t4, since it is a cyclic triad, the weight of the missing edge is the negative of the sum of other two edge weights.

In the end, we take the average of predicted weights over all triadic-links that these two nodes are a part of.

5.8 Linear Regression

In this experiment, we use a linear regression model to predict the weight of edge. Our features are generated from some algorithms mentioned above. First, the test edge e_t whose weight is to be predicted is removed from the network. Second, we remove the second edge e_s and predict the weight of e_s from the rest of the edges. And the predicted value of e_s will be the feature and the true value of e_s will be the ground truth. The second step is done multiple times to generate enough samples for training. After training, we use this model to predict the value of the removed edge e_t.
»»»> 2b8828d1b661eb188a51d817b3337c41c6e5fac7

6 Performance Evaluation

7 Conclusion

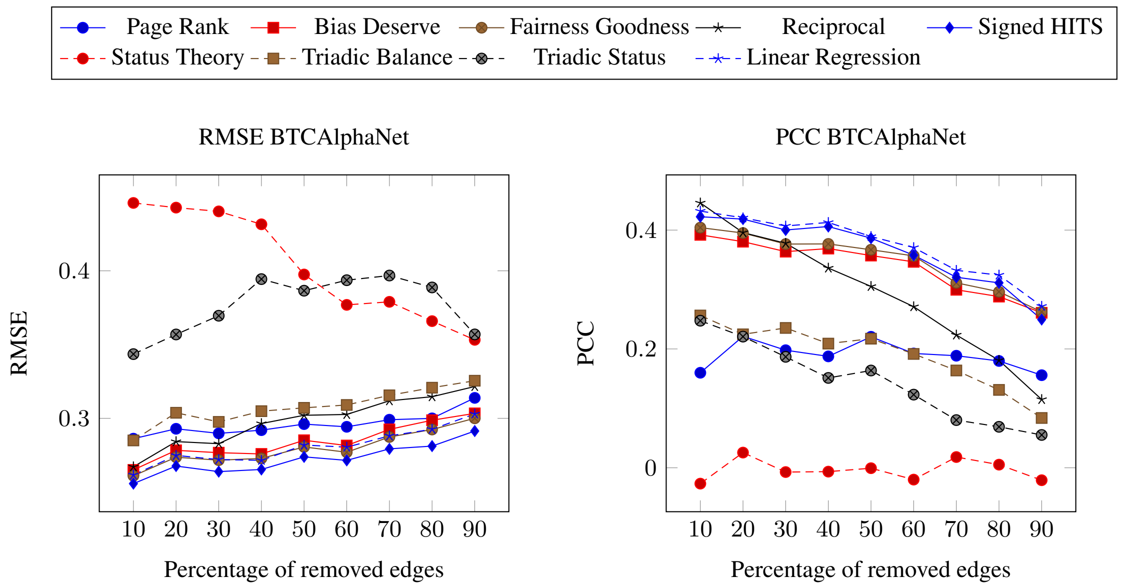


Figure 2: BTCAlphaNet

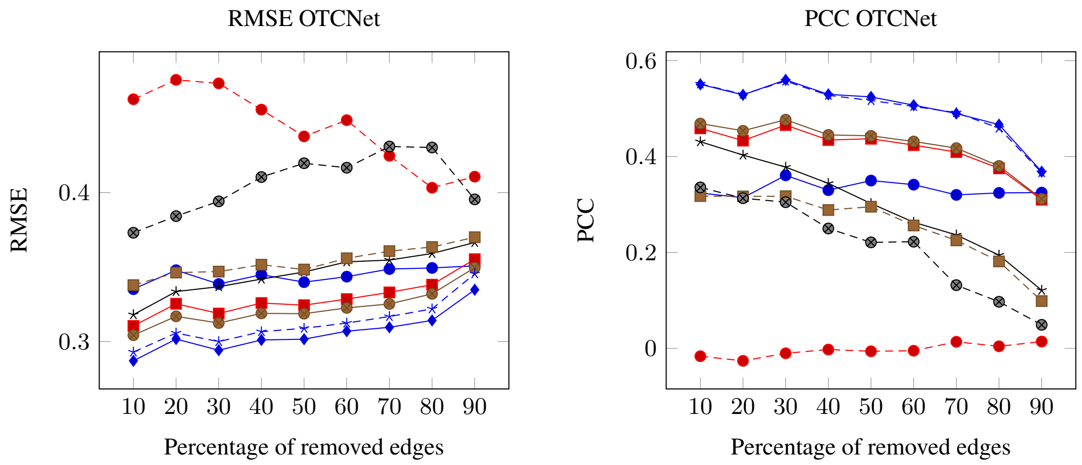


Figure 3: OTCNet

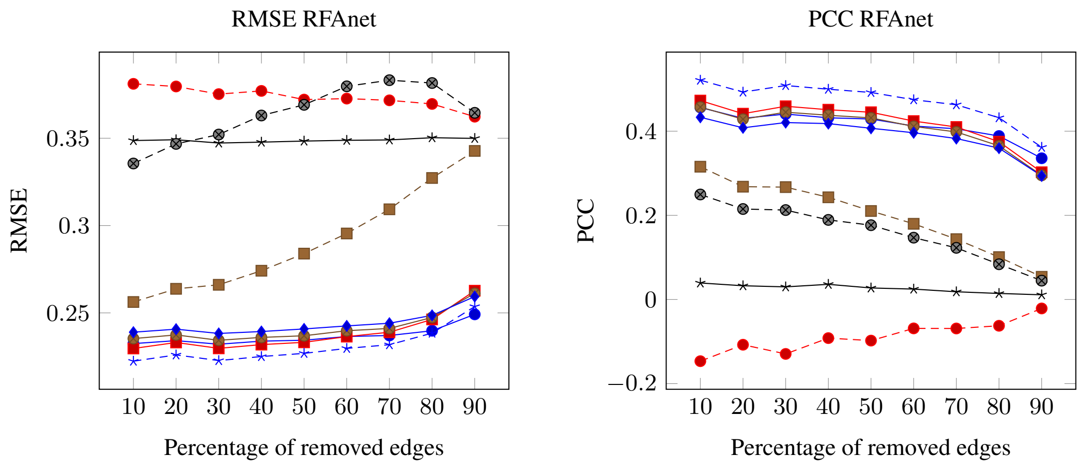


Figure 4: RFAnet