Link Prediction in Social Networks with Edge Aging

Ricky Laishram, Kishan Mehrotra, Chilukuri K. Mohan
Department of Electrical Engineering and Computer Science
Syracuse University, NY 13244-4100
Email: {rlaishra, mehrotra, mohan}@syr.edu

***Abstract*—In social networks that change with time, an important problem is the prediction of new links that may be formed in the future. Existing works on link prediction have focused only on networks where links are permanent, an assumption that is not valid in many real world social networks. In many real world networks, in addition to new links being created, existing links also get removed. In this paper, we extend existing link prediction methods and apply a supervised learning algorithm to networks with non-permanent links. The results we obtain on Twitter @-mention networks show that our method performs very well in such networks.**

***Index Terms*—Social Network, Link Prediction, Machine Learning, Artificial Intelligence**

1. Introduction

Social networks are extremely dynamic in nature. Two basic changes can happen in a social network - addition or removal of edges, and addition or removal of nodes. The features that characterize the network change as a consequence of these changes. In addressing the link prediction problem, we model the evolution of links between the nodes using features intrinsic to the network [1].

The study of link prediction in social networks is an important research area with applications in various fields. For example, social networking websites (such as Twitter, Facebook and LinkedIn) can give more relevant friend suggestions. In another important application, security agencies can use knowledge of an existing network to predict links and monitor future interactions between persons of interest [2].

Many researchers have worked on this problem using different approaches such as Markov chains [3], [4] and statistical relational learning [5]. Liben-Nowell et al. [1] approached the problem by relating link prediction and various similarity measures. Subsequent works [6], [7] have produced better results by using these similarity measures and applying machine learning techniques.

Most of the previous works focused on networks whose edges do not get deleted. However, in many real world social networks, edges are not permanent [8], [9]. Studies by Dunbar et al. [10] have found that, on average, an individual has a conflict with a directly connected neighbor in the network network every 7.2 months, resulting in the link being broken. So, to apply link prediction methods to such networks, we need to take into account these links that are deleted.

Existing works [11], [12] on social networks with non-permanent edges have considered the problem of creation and removal of nodes in such social networks. In our work, the focus is on link prediction and we assume that the set of nodes remains unchanged [1].

The contribution of this paper is the development of a link prediction method for social networks where edges are not permanent; that is, in our network model edges can either be created or deleted. This model attempts a more realistic representation of a real world social network [8], [9], [10]. In our model, each edge has a weight associated with it. This weight decays over time and if it falls below a certain threshold (θ), the edge is deleted. Interactions between nodes results in the creation of new edges, or strengthens existing edges. The feature scores (such as similarity measures, path distance measures etc.) used for link prediction are modified to take into account the edge weight. Finally, these feature scores are used to train a supervised learning algorithm and to predict the creation of edges.

Experiments using Twitter @-reply network [13] have shown that our method performs significantly better than existing methods. Our method produces results which are consistently of high quality, with AUC in the range of 0.79 to 0.87. In comparison, the performance of other existing methods varies widely, with AUC ranging from 0.53 to 0.83 depending on the network.

In Section 2, we discuss the network and model the edge decay procedure. In Section 3, we extend the network feature scores to take the edge weights into consideration. The link prediction method is described in Section 4; in this section we also discuss how the training and testing data are generated, and how class imbalance is handled . We present the experimental setup, results and analysis in Section 5 and 6. Finally, Section 7 presents concluding remarks.

2. Network Model

In this section, we describe the network model used in our link prediction method. First we describe the edge aging model that results in edge deletion.

In a social network, if an edge between nodes (u, v) existed at time t_i but no longer exists at some time t_j, where

$t_i < t_j$, the edge (u, v) is said to have been deleted. If the nodes u and v interact after the edge was deleted, it will be created again. The term "interact" is used in this paper to mean an action between two nodes that results in the formation of an edge between them or increases the edge weight if an edge already exist. For example, in a collaboration network, an interaction can be the action of two researchers collaborating on a research project.

Research by Burt [14] has shown that there is a pattern to the rate at which edges decay in a social network - new edges decay faster and older edges decay slower. If δ ($\delta < 1$) is the decay rate (or decay factor) at time $t = 0$, the decay rate at time $t = T$ is given by, $\delta_T \propto \delta^T$. This can also be written as $\delta_T \propto \delta \cdot \delta_{T-1}$.

We represent the edge weight of (u, v) at time t as $w(u, v, t)$, which represents the strength of the relationship between the nodes. So, if $w(u, v, t) > w(x, y, t)$, the node pair (u, v) has a stronger relationship than (x, y), and the relationship will survive longer than the one between x and y [14]. The weight of an edge decreases with time unless it is reinforced by an interaction. In the absence of reinforcement, the weight decay is described as

$$w(u, v, t) = \delta \times w(u, v, t - 1).$$

The edge is deleted when the weight of the relationship drops below a certain threshold, θ.

Given the network $G_t = \langle V, E_t \rangle$ at time t, we describe how to obtain $G_{t+1} = \langle V, E_{t+1} \rangle$. Between t and $(t + 1)$ some nodes interact with each other, represented by the set of edges $I_{t,t+1}$. The set E_{t+1} is obtained from E_t using the following three steps:

1) Generate
$$T_{t+1} = E_t \cup I_{t,t+1}$$
2) The weight of each link in T_{t+1} is calculated to incorporate the aging as well as reenforcement represented by $I_{t,t+1}$:
$$w(u, v, t + 1) = \begin{cases} \delta \cdot w(u, v, t) + 1, & \text{if } (u, v) \in I_{t,t+1} \\ \delta \cdot w(u, v, t), & \text{if } (u, v) \notin I_{t,t+1} \end{cases}$$
3) Edges with weight less than θ are removed from T_{t+1}, resulting in E_{t+1}.

This is a more realistic model of real world scenarios than the ones used in earlier link prediction works, described in Section 1. Many real world social networks have edges that are not permanent, and work by Burt [14] has shown that the edge decay process is a major cause of edge deletion. In our network model, edges are added as a result of interactions between nodes, and they can be deleted due to edge aging.

3. Network Feature Scores

Work by Liben-Nowell et al. [1] has demonstrated that feature scores based on node similarity and paths are good predictors of links in unweighted networks. Subsequent works [6], [7] have shown that using these network feature scores with supervised learning provide better predictions in unweighted networks with permanent links. These feature scores are: *Common Neighbor Score, Preferential Attachment Score, Adamic-Adar Score, Katz Score, Shortest Path Score* and *Rooted PageRank*. In this section we describe how these network feature scores are updated for directed networks, accounting for edge weights.

Notation:

- The set of neighbors of a node u is represented by $\Gamma(u)$.
- In a directed network, a node u has two types of neighbor sets - neighbors with links that are directed away from u, represented by $\Gamma_o(u)$, and neighbors with links that are directed towards u, represented by $\Gamma_i(u)$.
- We represent the definitions for the incoming and outgoing links using one equation with subscript α that takes two values – o and i.
- The set of paths of length l from node u to v is denoted as $\rho(u, v|l)$.

In the following subsections we describe extensions of various feature scores for weighted and directed networks.

3.1. Weighted Common Neighbor Score

In an undirected unweighted network, the Common Neighbor Score [15] of (u, v) is given by $CN(u, v) = |\Gamma(u) \cap \Gamma(v)|$. To account for edge weights, this score is modified to:

$$CN^w(u, v) = \sum_{z \in (\Gamma(u) \cap \Gamma(v))} \frac{w(u, z) + w(v, z)}{2}$$

In the case of directed networks, we have two common neighbor scores based on Γ_o and Γ_i. These Common Neighbor Scores are given by:

$$CN_\alpha(u, v) = |\Gamma_\alpha(u) \cap \Gamma_\alpha(v)|$$

for $\alpha = i, o$.

For directed weighted networks, for $\alpha = i, o$, these scores are:

$$CN^w_\alpha(u, v) = \sum_{z \in (\Gamma_\alpha(u) \cap \Gamma_\alpha(v))} \frac{w(z, u) + w(z, v)}{2} \quad (1)$$

3.2. Weighted Preferential Attachment Score

In an undirected network, the Preferential Attachment Score [16] is:

$$PA(u, v) = |\Gamma(u)| \cdot |\Gamma(v)|$$

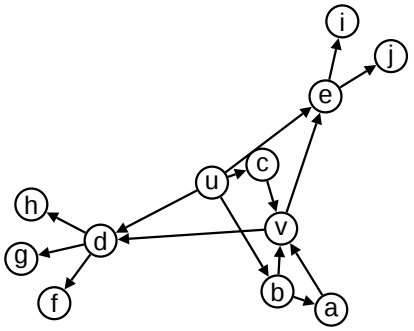


Figure 1. An example graph

In a directed network, the two Preferential Attachment Scores are:

$$PA_\alpha = |\Gamma_\alpha(u)| \cdot |\Gamma_\alpha(v)|$$

We can incorporate the edge weights into the Preferential Attachment Scores using the average weights of the edges to the neighboring nodes. For an undirected network,

$$PA^w(u,v) = \frac{\sum_{z\in\Gamma(u)} w(u,z)}{|\Gamma(u)|} \cdot \frac{\sum_{z\in\Gamma(v)} w(v,z)}{|\Gamma(v)|}$$

and for a directed network,

$$PA^w_\alpha(u,v) = \frac{\sum_{z\in\Gamma_\alpha(u)} w(z,u)}{|\Gamma_\alpha(u)|} \cdot \frac{\sum_{z\in\Gamma_\alpha(v)} w(z,v)}{|\Gamma_\alpha(v)|}$$

3.3. Weighted Adamic-Adar Score

For an undirected network, the Adamic-Adar Score [17] is:

$$AA(u,v) = \sum_{z\in(\Gamma(u)\cap\Gamma(v))} \frac{1}{\log|\Gamma(z)|}$$

For a directed network, it is:

$$AA_\alpha(u,v) = \sum_{z\in(\Gamma_\alpha(u)\cap\Gamma_\alpha(v))} \frac{1}{\log|\Gamma_\alpha(z)|}$$

To account for edge weights in the Adamic-Adar Score the neighbor counts are replaced by the sum of the edge weights. To illustrate this modification, consider the simple directed network in Fig. 1 in which we attempt to evaluate $AA^w_o(u,v)$. Since, $\Gamma_o(u) \cap \Gamma_o(v) = \{d, e\}$ we need evaluation of the contributions of nodes d and e towards $AA^w_o(u,v)$,

$$\gamma(d) = \frac{1}{\log\left(w(d,f) + w(d,g) + w(d,h)\right)}$$

and

$$\gamma(e) = \frac{1}{\log\left(w(e,i) + w(e,j)\right)}$$

respectively. Towards the final value of $AA^w_o(u,v)$ these contributions of $\gamma(d)$ and $\gamma(e)$ must take into account the average weights of the links to d and e respectively. Thus,

$$AA^w_o(u,v) = \frac{w(u,d) + w(v,d)}{\gamma(d)} + \frac{w(u,e) + w(v,e)}{\gamma(e)} \quad (2)$$

Equation 2 is generalized as follows:

$$AA^w_o(u,v) = \sum_{z\in(\Gamma_o(u)\cap\Gamma_o(v))} \frac{w(u,z) + w(v,z)}{\log\left(\sum_{x\in\Gamma_o(z)} w(z,x)\right)}$$

Similarly, we can can derive the equation for $AA^w_i(u,v)$. In the case of a weighted undirected network, we obtain:

$$AA^w(u,v) = \sum_{z\in(\Gamma(u)\cap\Gamma(v))} \frac{w(u,z) + w(v,z)}{\log\left(\sum_{x\in\Gamma(z)} w(z,x)\right)}$$

3.4. Weighted Katz Score

The Katz Score [18] is based on the paths between two nodes. In an unweighted network, it is given by:

$$KS(u,v) = \sum_{l=1}^{\infty} \left(\beta^l \cdot |\rho(u,v|l)|\right)$$

where β is the damping factor ($0 \le \beta \le 1$).

To take the edge weights into consideration, consider the network in Fig 1 as an example again. The set of paths from u to v are:

$$\rho(u,v|1) = \emptyset$$

$$\rho(u,v|2) = \{\{(u,b),(b,v)\},\{(u,c),(c,v)\}\}$$

$$\rho(u,v|3) = \{\{(u,b),(b,a),(a,v)\}\}$$

To take the edge weights into consideration, we will use the sum of average edge weights of each path instead of simply taking the path length. Then the contribution of each path set towards $KS^w(u,v)$ is:

$$\eta(1) = \beta^1 \cdot 0$$

$$\eta(2) = \beta^2 \cdot \left(\frac{w(u,b) + w(b,v)}{2} + \frac{w(u,c) + w(c,v)}{2}\right)$$

$$\eta(3) = \beta^3 \cdot \left(\frac{w(u,b) + w(b,a) + w(a,v)}{3}\right)$$

The Katz Score from node u to v is then given by:

$$KS^w(u,v) = \eta(1) + \eta(2) + \eta(3) \quad (3)$$

Equation 3 can be generalized to give us the Weighted Katz Score:

$$KS^w(u,v) = \sum_{l=1}^{\infty} \beta^l \cdot \sum_{p\in\rho(u,v|l)} \frac{\sum_{(x,y)\in p} w(x,y)}{l}$$

3.5. Weighted Shortest Path Score

In an undirected and unweighted network, the Shortest Path score of (u, v) is simply the reciprocal of the length of the shortest path from u to v. To illustrate extension of the Shortest Path Score to weighted and directed network, we consider the network in Fig. 1. In this network, the length of the shortest path from u to v is 2 and the associated set of paths of length $2 = l_{min}$ is,

$$\rho(u, v|2) = \{\{(u, b), (b, v)\}, \{(u, c), (c, v)\}\}$$

Associated with these two paths the set of average link weights is:

$$\overline{\rho}(u, v|2) = \left\{ \frac{w(u, b) + w(b, v)}{2}, \frac{w(u, c) + w(c, v)}{2} \right\}$$

The Shortest Path Score of (u, v) is then given by the average of the elements of the set $\overline{\rho}(u, v|2)$,

$$SP^w(u, v) = \frac{(w(u, b) + w(b, v) + w(u, c) + w(c, v))}{4} \quad (4)$$

Equation 4 can be generalized as follows:

$$SP^w(u, v) = \frac{\sum_{p \in \rho(u,v|l_{min})} \sum_{(x,y) \in p} w(x, y)}{l_{min} \cdot |\rho(u, v|l_{min})|}$$

3.6. Weighted Rooted PageRank

Nowell et al. [1] proposed an adaptation of the PageRank measure [19] called *Rooted PageRank*. In the Rooted PageRank algorithm, we start a random walk starting from a "root" node. At each step, there is an α probability of the walk resetting and returning to the root node. The random walk assumes that all neighboring nodes are equally important, and are selected uniformly at random.

As described in Section 2, not all edges are equal in a network that follows the edge aging model. So, we need to modify the part of the Rooted PageRank algorithm that handles the selection of the next node. Edges that have higher weights should have higher probability of being "walked" compared to edges that have lower weight.

Assume that we are currently at node u, for any node $x \in \Gamma_o(u)$, the probability of the random walk selecting x in the next step is given by:

$$p(x) = \frac{w(u, x)}{\sum_{z \in \Gamma_o(u)} w(u, z)} \quad (5)$$

If we are dealing with an undirected network, we simply need to replace $\Gamma_o(u)$ by $\Gamma(u)$ in Equation 5.

4. Link Prediction Method

In this section, we describe our link prediction method in terms of a classification problem. In the next two subsections, we describe the training data and testing data, and how class imbalance is handled. Finally, we highlight some of the key differences between our method and existing methods.

Link prediction can be approached as a classification problem using supervised learning [6], [7]. To apply supervised learning to link prediction, the network feature scores are considered as the feature vectors, and a node pair (u, v) is considered to be in the positive class if there is an edge between them and in the negative class if there is no edge.

Let G_1, G_2 and G_3 be three snapshots of the network at time t_1, t_2 and t_3, respectively, such that $t_1 < t_2 < t_3$. We calculate the training data F_{train} from the snapshot G_1, and the class labels for the training feature vectors C_{train} from the snapshot G_2. Performance of the trained algorithm is measured using the testing feature vectors, F_{test}, calculated from G_2 to predict the links in G_3.

To clarify the selection of the training and testing set, if we represent a feature score i between nodes u and v in snapshot G_k by $s_i(u, v, G_k)$, and the set of all nodes in the network by V, then

$$F_{train} = \{(s_1(u, v, G_1), s_2(u, v, G_1), \ldots, s_n(u, v, G_1)) \mid \forall(u, v) : (u \in V \wedge v \in V \wedge u \neq v)\}$$

$$F_{test} = \{(s_1(u, v, G_2), s_2(u, v, G_2), \ldots, s_n(u, v, G_2)) \mid \forall(u, v) : (u \in V \wedge v \in V \wedge u \neq v)\}$$

$$C_{train} = \{class(u, v, G_2) \mid \forall(u, v) : (u \in V \wedge v \in V \wedge u \neq v)\}$$

where E_k is the set of edges associated with graph G_k and

$$class(u, v, G_k) = \begin{cases} 1, & \text{if } (u, v) \in E_k \\ 0, & \text{if } (u, v) \notin E_k \end{cases}$$

In a graph, the maximum number of potential edges increases quadratically with the number of nodes. However, the number of edges in most real world networks is much smaller. So, the size of the positive class will be much smaller than that of the negative class.

Prior work on classification of extremely imbalanced data [20] has shown that good results can be obtained by using an ensemble of Support Vector Machine (SVM) classifiers trained with the minority class (in our case the positive class) and different samples of the majority class.

In the work reported here, we have used SVM to facilitate better comparison with the previous works, since most of them also use SVM. We note that machine learning algorithms other than SVM can also be used.

Suppose the training data is D_{train}. Consider its two components D_+ and D_- associated with the positive and negative class respectively, where $|D_-| >> |D_+|$. We implement the proposed ensemble approach as described below:

1) To use an ensemble of n SVM classifiers, n samples $[S_1, S_2, \ldots S_n]$ each of size $|D_+|$, are taken from D_- without replacement.
2) For $i = 1, 2, \ldots, n$, we train the classifier M_i using the sample S_i and D_+, and class membership represented by 1 for D_+ and 0 for S_i.
3) Let P_i consist of the predictions by classifier M_i, $i = 1, 2, \ldots, n$.
4) The predictions $P_1, P_2, \ldots P_n$ are combined using majority voting to generate the final prediction P.

$$P(u,v) = \begin{cases} 1 & \text{if } \left(\sum_{i=1}^{n} P_i(u,v)\right) > n/2 \\ 0 & \text{otherwise} \end{cases}$$

In our method, a new edge is predicted between two nodes u and v if there was no edge between them in G_2 and $P(u,v) = 1$, that is $(class(u,v,G_2) = 0) \wedge (P(u,v) = 1)$.

There are three main differences between our method and existing methods that use supervised learning:

1) The use of three snapshots in our method is different from the earlier works which use two snapshots. In our method, G_3 is completely unknown to the classification algorithm, and it is used only as ground truth for evaluation. In previous methods that use only two snapshots, the testing data is generated from the second snapshot, which is methodologically problematic since it can be argued that testing data was known during the training process.
2) Another difference between earlier methods and our method arises due to the fact that edges can be deleted in our network. In previous works, edges are permanent. So, the node pairs that already have an edge between them during the training period are not included in the testing data. As a result, the earlier methods have a testing data set that is smaller than the training data set. In our case, edges in the training period could be deleted in the testing period. So, we cannot remove them, and the testing data set and training data set have the same size in our method.
3) As described in Sections 2 and 3, the network model used in our method incorporates edge aging. So, the feature scores we use in our method depend on edge weights, whereas the feature scores in earlier methods do not take edge aging into consideration.

5. Experimental Setup

This section describes how the experimental simulations were carried out to test our method. The first subsection describes the dataset used for simulations. In the next subsection, we describe the sampling approach used..

5.1. Data set

We use the Twitter @-mention network to test our method. This network was obtained from the Twitter Streaming API [13] by observing it for 28 days. It contains approximately 10^5 nodes and 1.3×10^7 interactions between the nodes. Interactions are tweets from one twitter user to another using the @-username convention. In this network, each time step is one hour.

In this network, nodes represent the users, and there is an edge from node u to v if u sends a tweet mentioning v. Since the tweet is one way, the network is directed.

In the analysis presented below, edges are deleted using the edge aging model described in Section 2. The parameters for edge aging are: $\delta = 0.9$ and $\theta = 0.2$. These values were determined empirically.

To apply our link prediction method, the snapshots $G_1 = \langle V, E_1 \rangle$, $G_2 = \langle V, E_2 \rangle$ and $G_3 = \langle V, E_3 \rangle$ are taken at time $t_1 = 336$ hours, $t_2 = 504$ hours and $t_3 = 672$ hours, respectively.

5.2. Data sampling

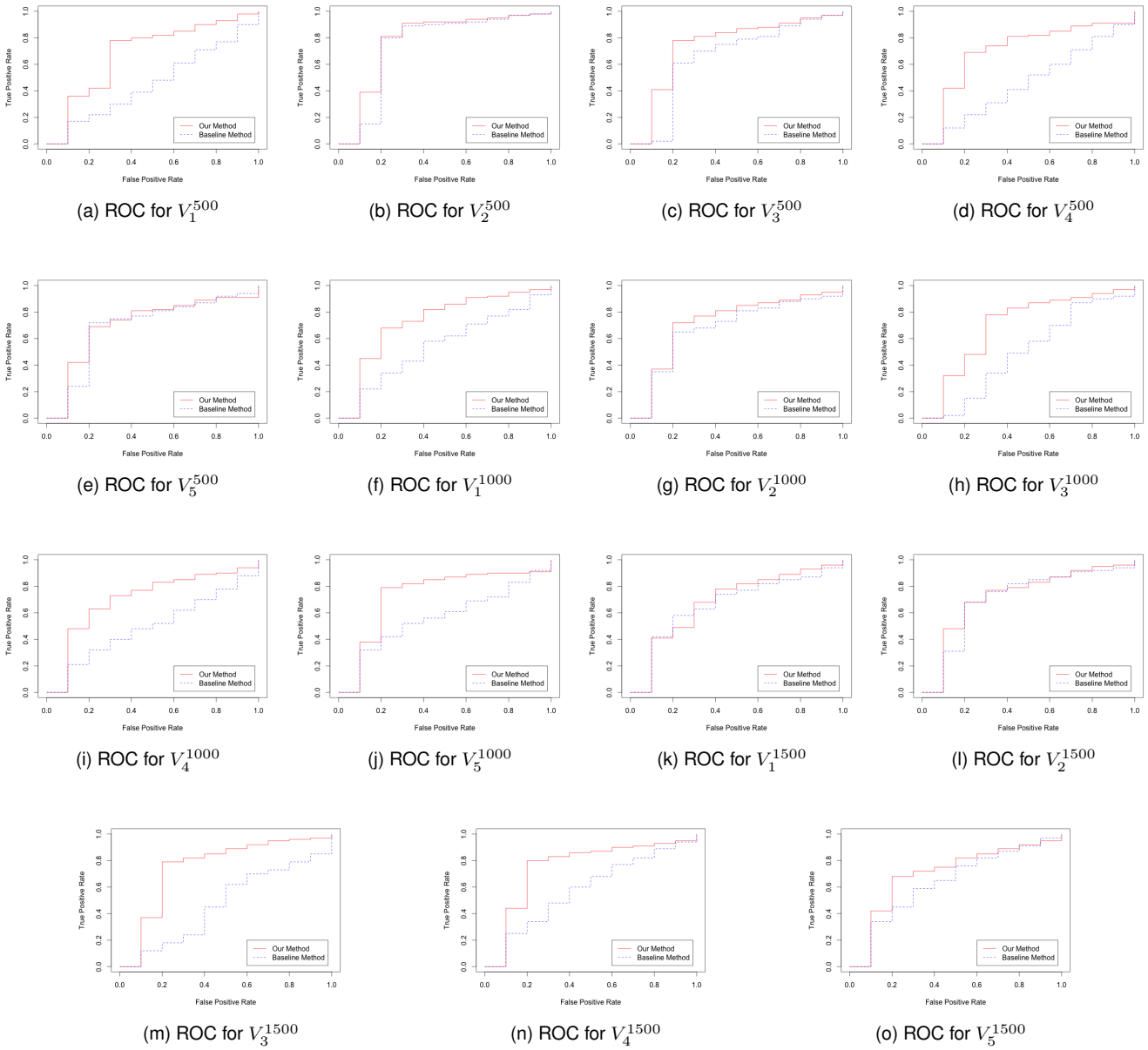
Since the network has approximately 10^5 nodes, the maximum number of potential edges is roughly 10^{10}. It is computationally very expensive to execute the link prediction algorithm over the entire network. So, we reduce the number of potential edges by selecting fifteen samples of node sets. Five of these samples $(V_1^{500}, V_2^{500}, V_3^{500}, V_4^{500}, V_5^{500})$ consist of 500 nodes each. Another five samples $(V_1^{1000}, V_2^{1000}, V_3^{1000}, V_4^{1000}, V_5^{1000})$ have 1000 nodes each, and the remaining five $(V_1^{1500}, V_2^{1500}, V_3^{1500}, V_4^{1500}, V_5^{1500})$ consist of 1500 nodes each.

Nodes whose graph distances are smaller are more likely to form an edge between them compared to those which are very far away [21], [7]. Hence nodes for each of the sample sets are selected using a random walk [22] starting from a random seed node.

6. Results and Analysis

We are not aware of any works on link prediction on networks where edges are non-permanent. Hence a modified version of the supervised link prediction methods by Al Hasan et al. [6], which assumes equal weights for all edges, is used as a baseline with which to compare the performance of our method. It should be noted here that Al Hasan's method uses some features that do not depend on the network structure (such as similarity in papers published). Such features are not considered here.

We denote the predictions made with the baseline method for sample i by Q_i, and made by our method by P_i.



(a) ROC for V_1^{500} (b) ROC for V_2^{500} (c) ROC for V_3^{500} (d) ROC for V_4^{500}

(e) ROC for V_5^{500} (f) ROC for V_1^{1000} (g) ROC for V_2^{1000} (h) ROC for V_3^{1000}

(i) ROC for V_4^{1000} (j) ROC for V_5^{1000} (k) ROC for V_1^{1500} (l) ROC for V_2^{1500}

(m) ROC for V_3^{1500} (n) ROC for V_4^{1500} (o) ROC for V_5^{1500}

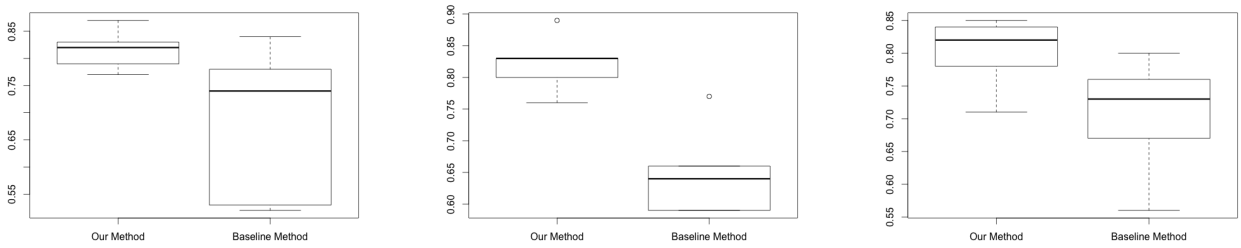
Figure 2. ROC curve comparison between our method and baseline method for the different samples. The red solid lines represent the ROC of our method and the blue dashed lines represent that of the baseline method.

In the datasets with imbalanced classes, the area under curve (AUC) of the receiver operating curve (ROC) is a better measure of performance than accuracy [23], [24]. So the prediction made using our method P_i is compared against the baseline method Q_i using the AUC. The AUCs of our method (P_i) and the baseline (Q_i) for all the fifteen samples are given in Table 1.

The ROCs for the fifteen samples are shown in Fig. 2. In the figures, the red solid lines represent the ROC curve obtained using our method, and the blue dashed lines represent that of the baseline method. It can be observed that our method outperforms the baseline in all the cases - and in some cases (Fig 2a, 2d, 2h, 2i and 2l) our method outperforms the baseline by a very large margin.

Fig 3 shows the box-plot of the AUC of our method against the baseline method for different sample sizes. The comparison for the samples with size 500, 1000 and 1500 are shown in Fig 3a, 3b and 3c respectively.

From the box-plot comparison (Fig 3), it can be observed that the median AUCs of our method are $0.82, 0.83$ and 0.82 for the samples with $500, 1000$ and 1500 nodes respectively; and for the baseline, they are $0.74, 0.64$ and 0.73 respectively. Thus our method has the better median AUC performance than the baseline.



(a) AUC comparison for samples with 500 nodes (b) AUC comparison for samples with 1000 nodes (c) AUC comparison for samples with 1500 nodes

Figure 3. Box-plot comparison between the AUC-ROC of our method against the baseline method for different sample sizes.

TABLE 1. AUC COMPARISON BETWEEN PREDICTION USING OUR METHOD P_i AND THE BASELINE METHOD Q_i

Sample	AUC of Q_i	AUC of P_i
V_1^{500}	0.53	0.79
V_2^{500}	0.83	0.87
V_3^{500}	0.74	0.82
V_4^{500}	0.52	0.77
V_5^{500}	0.78	0.83
V_1^{1000}	0.64	0.83
V_2^{1000}	0.77	0.89
V_3^{1000}	0.59	0.80
V_4^{1000}	0.59	0.76
V_5^{1000}	0.66	0.83
V_1^{1500}	0.76	0.78
V_2^{1500}	0.80	0.82
V_3^{1500}	0.56	0.85
V_4^{1500}	0.67	0.74
V_5^{1500}	0.73	0.71

It can also be observed that the Inter-Quartile Range (IQR) of our method is much smaller compared to the baseline. The IQRs of our method are $0.07, 0.08$ and 0.1 for the samples with $500, 1000$ and 1500 nodes respectively. For the baseline method, these values are $0.28, 0.13$ and 0.17 respectively. This shows that performance of our method is very consistent compared to the baseline regardless of the sample size.

7. Conclusion

Link prediction for networks where edges are non-permanent has not been investigated before. Many such networks exist in the real world [10], [25]. To account for this, we used a network model (Section 2) in which edges can be created or deleted. We updated the network feature scores (Section 3) to apply to our network model. This meant taking the edge weights into consideration while calculating the feature scores. Finally, we use these feature scores to train a classifier as described in Section 4 and predict presence or absence of links between nodes in the next network snapshot.

The experimental results in Section 6 show that the method we propose in this paper achieves very good performance regardless of the sample size. The AUC of our method is consistently around 0.80 for all fifteen samples we tested on. Compared to the baseline algorithm, for which the AUC varies from 0.53 to 0.83, our method performs very well.

It is also worth noting that the feature scores in our method depend on only the network topology. We do not use any domain knowledge or feature scores external to the network. Hence it can be applied easily to other networks that have non-permanent edges.

An area worth further investigation is to extend this method to use more than three network snapshots. If we increase the number of snapshots, the data could potentially provide us with a more detailed picture of how the network evolves. This might provide more insight that will be helpful in link prediction. Another new application area is link prediction in multi-layer social networks with two types of links: inter-layer links and intra-layer links.

References

[1] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *Journal of the American society for information science and technology*, vol. 58, no. 7, pp. 1019–1031, 2007.

[2] V. E. Krebs, "Mapping networks of terrorist cells," *Connections*, vol. 24, no. 3, pp. 43–52, 2002.

[3] R. R. Sarukkai, "Link prediction and path analysis using markov chains," *Computer Networks*, vol. 33, no. 1, pp. 377–386, 2000.

[4] B. Taskar, M.-F. Wong, P. Abbeel, and D. Koller, "Link prediction in relational data," in *Advances in neural information processing systems*, p. None, 2003.

[5] A. Popescul and L. H. Ungar, "Statistical relational learning for link prediction," in *IJCAI workshop on learning statistical models from relational data*, vol. 2003, Citeseer, 2003.

[6] M. Al Hasan, V. Chaoji, S. Salem, and M. Zaki, "Link prediction using supervised learning," in *SDM 06: Workshop on Link Analysis, Counter-terrorism and Security*, 2006.

[7] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla, "New perspectives and methods in link prediction," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 243–252, ACM, 2010.

[8] J. Preusse, J. Kunegis, M. Thimm, and S. Sizov, "Decline–models for decay of links in networks," *arXiv preprint arXiv:1403.4415*, 2014.

[9] S. Asur, B. A. Huberman, G. Szabo, and C. Wang, "Trends in social media: Persistence and decay," *Available at SSRN 1755748*, 2011.

[10] R. Dunbar and A. Machin, "Sex differences in relationship conflict and reconciliation," *Journal of Evolutionary Psychology*, vol. 12, no. 2-4, pp. 109–133, 2014.

[11] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins, "Microscopic evolution of social networks," in *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 462–470, ACM, 2008.

[12] S. Wu, A. Das Sarma, A. Fabrikant, S. Lattanzi, and A. Tomkins, "Arrival and departure dynamics in social networks," in *Proceedings of the sixth ACM international conference on Web search and data mining*, pp. 233–242, ACM, 2013.

[13] "Twitter streaming api."

[14] R. S. Burt, "Decay functions," *Social networks*, vol. 22, no. 1, pp. 1–28, 2000.

[15] M. E. Newman, "Clustering and preferential attachment in growing networks," *Physical Review E*, vol. 64, no. 2, p. 025102, 2001.

[16] A.-L. Barabâsi, H. Jeong, Z. Néda, E. Ravasz, A. Schubert, and T. Vicsek, "Evolution of the social network of scientific collaborations," *Physica A: Statistical mechanics and its applications*, vol. 311, no. 3, pp. 590–614, 2002.

[17] L. A. Adamic and E. Adar, "Friends and neighbors on the web," *Social networks*, vol. 25, no. 3, pp. 211–230, 2003.

[18] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.

[19] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web.," 1999.

[20] Y. Liu, A. An, and X. Huang, "Boosting prediction accuracy on imbalanced datasets with svm ensembles," in *Advances in Knowledge Discovery and Data Mining*, pp. 107–118, Springer, 2006.

[21] D. J. Watts and S. H. Strogatz, "Collective dynamics of small-world networks," *nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[22] J. Leskovec and C. Faloutsos, "Sampling from large graphs," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 631–636, ACM, 2006.

[23] N. V. Chawla, *Data Mining and Knowledge Discovery Handbook*, ch. Data Mining for Imbalanced Datasets: An Overview, pp. 853–867. Boston, MA: Springer US, 2005.

[24] T. Fawcett, "An introduction to roc analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[25] Y. Murase, H.-H. Jo, J. Török, J. Kertész, and K. Kaski, "Modeling the role of relationship fading and breakup in social network formation," *arXiv preprint arXiv:1505.00644*, 2015.