LinkBoost: A Novel Cost-Sensitive Boosting Framework for Community-Level Network Link Prediction

Prakash Mandayam Comar, Pang-Ning Tan, Anil K Jain
Department of Computer Science and Engineering
Michigan State University
East Lansing, USA
Email: (mandayam,ptan,jain)@cse.msu.edu

***Abstract*—Link prediction is a challenging task due to the inherent skewness of network data. Typical link prediction methods can be categorized as either local or global. Local methods consider the link structure in the immediate neighborhood of a node pair to determine the presence or absence of a link, whereas global methods utilize information from the whole network. This paper presents a community (cluster) level link prediction method without the need to explicitly identify the communities in a network. Specifically, a variable-cost loss function is defined to address the data skewness problem. We provide theoretical proof that shows the equivalence between maximizing the well-known modularity measure used in community detection and minimizing a special case of the proposed loss function. As a result, any link prediction method designed to optimize the loss function would result in more links being predicted within a community than between communities. We design a boosting algorithm to minimize the loss function and present an approach to scale-up the algorithm by decomposing the network into smaller partitions and aggregating the weak learners constructed from each partition. Experimental results show that our proposed `LinkBoost` algorithm *consistently* performs as good as or better than many existing methods when evaluated on 4 real-world network datasets.**

I. Introduction

The ability to predict the formation of links in a network is an important task in network analysis. A reliable link prediction model is useful for uncovering missing links in a static network or for projecting the formation of new links in a dynamic network. The level of link prediction accuracy sought often varies depending on the context of its application. For example, despite its poor accuracy, the *FOF* (Friend of Friend) algorithm has been extensively used in predicting future links between users in large social networks. However, in critical applications such as bio-surveillance and terrorist network monitoring, predictions are cost sensitive in that there is a severe penalty factor associated with different incorrect predictions made by algorithm.

The low accuracies of link prediction algorithms can be attributed to the inherent *skewness* of network data. In this regards, there are two types of skewness to be considered. The obvious one is class skewness, which refers to the lopsided ratio of non-linked (the *negative* class) to linked (the *positive* class) node pairs in a network. Typically the ratio is of the order of $O(1/N)$. Such high skewness would result in a biased decision boundary and requires inclusion of skew correction approaches into the link prediction framework. Another type of skewness, which has received little attention in the link prediction literature, is in the degree distribution of the nodes. Since many networks exhibit a scale-free behavior, this results in the (few) high degree nodes exerting the most influence on the prediction for the positive class. As a result, most link prediction models tend to fail in their prediction for the majority of the low-degree nodes. To avoid this, we need to develop a loss function for link prediction that considers both type of skewness in the data.

In addition, existing link prediction methods are either global or local in nature. The former (e.g., *common neighbors*[13]) simply utilizes information from the immediate neighborhood of the nodes to make its prediction. Though such an approach tends to perform poorly especially on large networks, it is computationally efficient. The latter (e.g., based on supervised learning [9], [19]) often achieves better performance but at the expense of higher computation time. Moving away from these two extremes is the method of finding links at the community (cluster) level. The intuition here is that links are more likely to be formed between nodes in the same community rather than those in different communities. However, identifying the right set of communities is itself a challenging problem. One of the most well-known community finding algorithms is based on the network modularity measure [15]. However, to the best of our knowledge, none of the existing link prediction algorithms are designed to optimize the measure.

The main contributions of this paper are as follows:

- We propose a variable-cost loss function for supervised link prediction that considers both the imbalanced class distribution (of linked and non-linked node pairs) as well as skewness in the degree distribution. The variable-cost loss function addresses the bias in degree distribution by penalizing the misclassification of low-degree linked node pairs more than misclassification of high-degree linked node pairs.
- We show the intimate relationship between the pro-

posed loss function and the modularity measure used to identify communities in a network. As a consequence, a link prediction algorithm that optimizes the proposed loss function is inherently biased towards finding links within communities without explicitly identifying the communities.

- We design a boosting algorithm for link prediction called `LinkBoost` that optimizes the cost-sensitive loss function. We also provide weak learners that utilize the nodal attributes to estimate the link potentials between the node pairs.
- We present an approach for scaling up the `LinkBoost` framework by first decomposing the network into smaller, potentially overlapping partitions and then combining the predictions made by the weak learners constructed from the different partitions.

To the best of our knowledge, the degree dependent cost sensitive link prediction algorithm is the first of its kind. Through the design of our loss function, the paper also highlights the connection between community-based link prediction, modularity measure, and the boosting algorithm. Finally, experimental results show that our proposed LinkBoost algorithm *consistently* performs as good as or better than many existing methods when evaluated on 4 real-world network datasets.

The rest of the paper is organized as follows. In Section II we give an overview of related work. In Section III we discuss the nature of cost sensitive learning. In Section IV we present the proposed variable-cost loss function and discuss its importance to the link prediction problem. In Section V we discuss the relationship between the risk associated with proposed loss function and the well-known modularity measure. In Section VI we derive the boosting framework for the proposed loss function and describe the weak learners used for the framework. We also describe the heuristics for applying the proposed LinkBoost algorithm on large networks. Experimental evaluations are provided in Section VII, followed by concluding remarks.

II. Related Work

Link prediction algorithms can be categorized in many ways. First, the algorithms can be supervised or unsupervised. Second, they can be based on the observed link structure only or may incorporate nodal attributes. Third, the algorithms may utilize the link structure information from immediate neighborhoods (local methods), entire network (global methods), or at community levels.

The simplest unsupervised link prediction algorithm is based on computing the similarity scores between a pair of nodes using the nodal attributes or their local network topology. Examples of such local methods include common neighbors, Salton index [18], preferential attachment [2], [23], and Adamic-Adar index [1]. The performance of the different local measures were compared in [13], [24]. The results suggest that simple common neighbors approach performs better than other local measures. A theoretical justification for the better performance of *common neighbors* approach was presented in [16].

Unsupervised global methods for link prediction typically consider the weighted paths between node pairs. Examples include the Katz measure [12], random walk with restart [22], average commute time, and matrix forest index [4]. Measures based on paths, in general offer higher prediction accuracy compared to the local similarity measures. However, they require the entire network link structure and their computations are generally time consuming.

Link prediction using supervised learning has been investigated by many authors [9], [10], [21], [19]. Al Hasan et al. [9] derived several nodal and topological features for link prediction and applied a variety of classifiers such as support vector machines and decision tree to predict links in bibliographic databases. Kashima and Abe [10] proposed a parameterized probability model for the link structure and developed an expectation maximization algorithm to estimate the model parameters. Scripps et al. [19] employed a regularized matrix factorization approach for link prediction. Taskar et al. [21] used a relational Markov network to jointly model the nodal attributes and links. However, one of the main challenges in link prediction is the extremely large class skew, which leads to poor detection rate. Rattigan and Jensen [17] suggested an alternative problem known as anomalous link discovery to identify the most interesting links in the network. Recently, there have been attempts to develop a semi-supervised approach for link prediction [11] as well as combining link prediction with other tasks such as collective classification [3].

More recently, there have been attempts to develop link prediction algorithms using generative models that account for the clustering (community) structure in the network. Guimera et al. [8], used the likelihood based methods for estimating the reliability of a link between any node pair, given the observed link structure. The reliability score is then used to predict both missing and spurious links. Clauset et al. [5] have proposed maximum likelihood based methods that represent the clusters in the network as a hierarchy, which in turn are represented as a dendrogram. Each dendrogram has an associated likelihood value indicating the strength of community structure represented by the dendrogram. The missing links are predicted by first sampling large a number of dendrograms proportional to their likelihood and for each unconnected node pairs i and j, the expected connecting probability is computed by averaging the corresponding probability over all sampled dendrograms. Finally, the node pairs are sorted according the connecting probability and highest ranked ones are declared as potential links.

Both the reliability and hierarchical cluster model try to estimate the link potentials between the node pairs at cluster level. They in fact, average over all possible partitions of communities present in given network which makes it is very costly to implement even on small sized networks. In this paper, we suggest an alternative to these two algorithms which strive to identify more links with in a community. We do this by defining loss function whose associated risk when minimized, leans towards giving higher rating for the with in community node pairs. We also show the relationship between the proposed cost sensitive loss function and the well known modularity measure used for clustering networks.

III. Preliminaries

We consider the link prediction task as a binary classification problem, in which a node pair is assigned to the positive class if there is a link between them, or to the negative class otherwise. Let $\mathcal{V} = \{1, 2, \cdots, n\}$ denote the set of nodes in the network and $\mathcal{E} = \mathcal{V} \times \mathcal{V}$ denote the set of all node-pairs. We represent the adjacency matrix of the network as A, where $A_{ij} = \{+1, -1\}$ indicating the presence or absence of links. Each node $i \in \mathcal{V}$ is associated with a set of d-dimensional nodal attributes $\mathbf{x}_i = \{x_{i1}, x_{i2}, ..., x_{id}\}$. Our objective is to learn a target function $f : \mathcal{V} \times \mathcal{V} \rightarrow \Re$ that maps each node pair to its link potential. The function is optimal if it minimizes the expected risk $R = E_{\mathcal{E},A}[L(f(e), a)]$ for any given node-pair $e \in \mathcal{E}$, where $L[f(e), a]$ is the loss function. The loss function usually takes the form of

$$L[f(\mathbf{e}_{ij}), A_{ij}] = \begin{cases} 0, & \text{if } \operatorname{sgn}(f(\mathbf{e}_{ij})) = A_{ij} \\ C_1, & \text{if } \operatorname{sgn}(f(\mathbf{e}_{ij})) \neq A_{ij} = 1 \\ C_2, & \text{if } \operatorname{sgn}(f(\mathbf{e}_{ij})) \neq A_{ij} = -1 \end{cases} \tag{1}$$

where $sgn(\cdot)$ is the sign function, whose value is equal to +1 if its argument is non-negative and -1 otherwise. When $C_1 = C_2 = 1$, this corresponds to the 0-1 loss function. Many supervised link prediction algorithms are designed to yield a classifier that minimizes the following 0-1 empirical loss function, which is given by

$$\hat{R}_{\text{0-1}} = \frac{1}{n^2} \sum_{i,j=1}^{n} \mathbf{I}\Big[A_{ij}\operatorname{sgn}(f(\mathbf{e}_{ij})) \leq 0\Big] \tag{2}$$

where $\mathbf{I}(\cdot)$ is an indicator function, which is equal to 1 if its argument is true and zero otherwise.

A major hinderance in this binary classification task is the class imbalance problem. In the social network data, negative examples (non-linked node pairs) tend to outnumber the positive examples by a significantly large proportion. The literature for classification on imbalanced data suggests two approaches to tackle this problem, namely, sampling and cost-sensitive learning. In the first approach, a balanced training set is obtained by undersampling the negative examples or oversampling the positive examples. This approach has several drawbacks. Firstly, undersampling the negative examples reduces the amount of data available for training an accurate model. Furthermore, one has to do the undersampling repeatedly to remove the sampling bias. On the other hand, oversampling the positive examples in the social network data increases the training set size significantly, which in turn, makes the training time considerably longer.

The cost sensitive learning approach is based on the premise that different classes of examples (positives or negatives) incur different penalties for misclassification. The loss function defined in (1) is cost sensitive if $C_1 \neq C_2$, where C_1 is the cost for misclassifying linked node pairs as non-linked node pairs and C_2 is the cost of misclassifying the non-linked node pairs as linked node pairs.

The loss function defined in (1) and the associated risk functions are not differentiable, hence does not offer mathematical dexterity in designing classifiers. The risk associlated with exponential loss can be used as an alternative:

$$\hat{R}_{\text{exp}} = \frac{1}{n^2} \sum_{ij} \exp\Big[-A_{ij} f(\mathbf{e}_{ij})\Big] \tag{3}$$

The exponential risk is a continuous and differentiable function and it bounds the risk for 0-1 loss from above. It can be shown that an equivalent expression bounding the cost sensitive loss function defined in (1) is

$$\begin{aligned} \hat{R}_{\text{cost-sens}} &= \frac{1}{n^2} \sum_{ij} \Big[\mathbf{I}(A_{ij} = 1) \exp(-C_1 f(\mathbf{e}_{ij})) \\ &\qquad + \mathbf{I}(A_{ij} = -1) \exp(C_2 f(\mathbf{e}_{ij}))\Big] \end{aligned} \tag{4}$$

By simply changing the class labels for presence and absence of links from $\{+1, -1\}$ to $\{C_1, -C_2\}$ the cost sensitive risk in Equation (4) can be transformed to the empirical risk of (3). Generally the cost parameters C_1 and C_2 are chosen in such a way that they correct for the classification bias that arises due to skewness in the class distribution. If n_+ and n_- represent the number of positive and negative examples in the data, then C_1 and C_2 are often chosen such that $\frac{C_1}{C_2} = \frac{n_-}{n_+}$. For large sparse networks, the fraction $\frac{n_-}{n_+} = O(n)$, thus if we fix $C_1 = 1$, then the value of $C_2 \sim n^{-1}$ which results in working with extreme penalties that are easily polluted by the limitations of the machine precision. To avoid this, we need to scale the cost of both positive and negative labels such that the desired penalty ratio is maintained. Another significance of the cost ratio $\frac{C_1}{C_2}$ is its role in determining the optimal cost sensitive decision surface. The optimal decision surface for the cost sensitive learning

$$f^* = \arg\min_f E_{\mathcal{E},A}[L(e, a)]$$

is given by the Bayes Decision Rule [14]

$$f^*(e) = \log \frac{\mathrm{P}_{A|\mathcal{E}}(a = 1|\mathbf{e})C_1}{\mathrm{P}_{A|\mathcal{E}}(a = -1|\mathbf{e})C_2} \tag{5}$$

Hence for any cost structure (C_1, C_2), cost sensitive optimality differs from cost insensitive optimality only through the threshold $T = \log \frac{C_1}{C_2}$.

The preceding formulation assumes that C_1 and C_2 are constants. We argue that it may not desirable to treat the misclassification cost for all the linked (and non-linked) node pairs by the same yard stick. In the next section, we present a variable cost loss function, such that misclassification of low degree linked node pairs incurs more penalty than misclassification of high degree linked node pairs. We also show that such modification leads to a link prediction algorithm that leans towards predicting more links within the same community than otherwise.

IV. Variable Cost Loss Function for Link Prediction

This section describes our rationale for introducing a variable cost loss function for link prediction. It is generally observed that the degree distribution of real-world networks tends to follow a power law distribution, where there are few high degree nodes and a large number of low degree nodes. Consequently, a supervised learning algorithm for link prediction not only faces the bias from the large number of non-linked node pairs (negative class) but also from the small number of high degree nodes. Specifically, among the linked node pairs (positive class), the high degree nodes contribute more in determining the decision surface. Since we want to build models that can explain the observed links between any node pairs and not strongly influenced by the links formed for a few of the high degree nodes, we need to design a loss function that removes this bias within the positive class.

One way to do this would be to make the misclassification penalty dependent on the degree of the nodes. Let k_i be the degree of node i. Then the cost of misclassifying the linked node pair $\mathbf{e}_{ij}$ is given by

$$C_1(\mathbf{e}_{ij}) = 1 - \beta k_i k_j,$$

where β is user defined parameter, typically chosen to keep the cost function non-negative. Notice that C_1 monotonically decreases with increasing degrees of k_i or k_j, thus penalizing more for misclassification of links between low degree nodes compared to misclassification of links between the high degree nodes.

Analogously, the same reasoning can be made about the non-linked node pairs. The low degree node pairs exert a higher influence on the negative class than the high degree node pairs. To remove this bias among the negative examples, we define the cost for misclassifying non-linked node pairs as

$$C_2(\mathbf{e}_{ij}) = \gamma k_i k_j,$$

which increases when the node degrees are higher. We now need to account for the overall bias between the positive and negative examples, this is done by choosing the value of β and γ such that $C_2 < C_1$. Putting it all together, we obtain the following loss function

$$L(f(\mathbf{e}_{ij}), A_{ij}) = \begin{cases} 1 - \beta k_i k_j, & \text{if } sgn(f(\mathbf{e}_{ij})) \neq A_{ij} = \ 1 \\ \gamma k_i k_j, & \text{if } sgn(f(\mathbf{e}_{ij})) \neq A_{ij} = -1 \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

The distinguishing aspect of the above loss function is that it assigns variable misclassification cost for different node pairs. When $\beta = \frac{1}{\sum_i k_i}$ the term $\beta k_i k_j$ represents the expected number of links between the node pair i and j [15]. We will show in the next section that for this specific value of β, lowering the risk associated with the variable cost loss function is same as maximizing the modularity measure. This results in the learning algorithm being biased more towards learning links between the node pairs in same community than learning the links that lie between the communities.

V. Modularity

A well accepted conjecture in the network mining literature is that link densities are expected to be higher within a community than between communities. This suggests the possibility of an intimate connection between link prediction and community finding tasks. A popular method to identify communities in a network is using the well known modularity measure [15]. Here the possible existence of a community in a given network is revealed by comparing the actual link density in the subgraph induced by the community and the density one would *expect* to have if the nodes of the subgraph were linked irrespective of the community structure. The modularity measure can be mathematically quantified as follows.

$$\mathcal{Q} = \sum_{ij} \Big[\mathbf{I}(A_{ij} > 0) - P_{ij} \Big] \mathbf{I}(c_i = c_j), \quad (7)$$

where P_{ij} is the expected number of links between the nodes i and j under a null model (or reference network). The variables c_i and c_j represent the community membership of nodes i and j respectively. Modularity-based community finding algorithms are designed to assign the nodes to different communities such that the overall modularity measure, $\mathcal{Q}$, is maximized. The null model used often corresponds to that of a random graph with the same degree distribution as the given network. This leads to [6]

$$\mathcal{Q} = \frac{1}{n^2} \sum_{ij} \left[\mathbf{I}(A_{ij} > 0) - \frac{k_i k_j}{2m} \right] \mathbf{I}(c_i = c_j) \quad (8)$$

where $m = \sum_i k_i/2$ is the number of links in the network.

The following theorem shows the equivalence between maximizing (8) and minimizing the risk associated with a special case of the loss function given in (6). Consider a community-based link prediction model that predicts the existence of a link between a node-pair based on whether the nodes are in the same community, i.e.,

$$sgn(f_{\text{comm}}(\mathbf{e}_{ij})) = \begin{cases} +1, & \text{if } \mathbf{I}(c_i = c_j); \\ -1, & \text{otherwise.} \end{cases} \tag{9}$$

Theorem 1. *For the variable cost loss function given in* (6)*, minimizing the risk associated with the community-based link prediction model $f_{comm}(\boldsymbol{e}_{ij})$ with $\beta = \gamma = \frac{1}{\sum_i k_i}$ is equivalent to maximizing the modularity function in* (8)

Proof: The empirical risk associated with the variable cost loss function given in (6) for the community-based link prediction model is

$$\begin{aligned}\hat{R}_{mod} &= \frac{1}{n^2}\Bigg[\sum_{ij:A_{ij}=1} \mathbf{I}(sgn(f_{\text{comm}}(\mathbf{e}_{ij})) = -1)(1-\beta k_i k_j) \\ &\quad + \sum_{ij:A_{ij}=-1} \mathbf{I}(sgn(f_{\text{comm}}(\mathbf{e}_{ij})) = 1)(\gamma k_i k_j)\Bigg] \\ &= \frac{1}{n^2}\Bigg[\sum_{ij:A_{ij}=1}(1-\delta_{ij})(1-\frac{1}{2m}k_i k_j) \\ &\quad + \sum_{ij:A_{ij}=-1}(\delta_{ij}\frac{1}{2m}k_i k_j)\Bigg] \end{aligned} \tag{10}$$

where we have replaced $\mathbf{I}(sgn(f_{\text{comm}}(\mathbf{e}_{ij})) = 1) = \delta_{ij}$ and $\beta = \gamma = 1/2m$. Now minimizing the empirical risk with respect to δ is equivalent to maximizing the following

$$\begin{aligned} &\min_f \hat{R}_{mod} \\ =& \max_\delta \frac{1}{n^2}\Bigg[\sum_{ij:A_{ij}=1}\delta_{ij}(1-\frac{k_i k_j}{2m}) - \sum_{ij:A_{ij}=-1}\delta_{ij}\frac{1}{2m}k_i k_j\Bigg] \\ =& \max_\delta \frac{1}{n^2}\sum_{ij}\delta_{ij}\left[\mathbf{I}(A_{ij}>0) - \frac{k_i k_j}{2m}\right] \end{aligned} \tag{11}$$

Since $\delta_{ij} = \mathbf{I}(sgn(f_{\text{comm}}(\mathbf{e}_{ij})) = 1) = \mathbf{I}(c_i = c_j)$, this completes the proof. ■

The preceding theorem suggests that maximizing the modularity measure is equivalent to minimizing a special case of the loss function using the clustering solution, $f_{\text{comm}}(\mathbf{e}_{ij})$ as the link prediction model. The clustering solution uses only the network topology to explain the link potential between node pairs. In contrast, our proposed variable cost loss function provides a framework that allows us to estimate the link potential using other information including the nodal attributes. We design the $f_{\text{comm}}(\mathbf{e}_{ij})$ as function of nodal attributes x_i and x_j. Our experimental results have demonstrated the effectiveness of using an exponential loss compared to modularity function (24) for link prediction.

VI. Boosting Approach for Link Prediction

This section presents our method for optimizing the variable-cost loss function given in Section IV. The risk associated with the loss function given in (6) is non-differentiable, so we employ the following variable-cost empirical risk function:

$$\begin{aligned}\hat{R}_{mod} &= \frac{1}{n^2}\sum_{ij}\Big[\mathbf{I}(A_{ij}=1)\exp[-(1-\beta k_i k_j)f(\mathbf{e}_{ij})] \\ &\quad + \mathbf{I}(A_{ij}=-1)\exp[\gamma k_i k_j f(\mathbf{e}_{ij})]\Big]\end{aligned} \tag{12}$$

If we set $\beta = \gamma$ then the above loss function reduces to

$$\hat{R}_{mod} = \frac{1}{n^2}\sum_{ij}\exp\left[(\mathbf{I}(A_{ij}>0) - \beta k_i k_j)f(\mathbf{e}_{ij})\right] \tag{13}$$

This form of loss function is well studied in the machine learning community using additive modeling or *boosting* techniques [7]. Specifically, an additive model takes the form of

$$f_\alpha(\mathbf{x}) = sgn\left[\sum_t \alpha_t f_t(\mathbf{x})\right].$$

For boosting, each f_i corresponds to a weak learner and the goal is to identify a sequence of constants $\alpha_1, ..\alpha_k$ such that a linear combination of the weak learners performs better than any of the individual learners.

A. Estimating α_t

Our aim is to design a boosting algorithm that minimizes the variable-cost empirical risk function $\hat{R}_{mod}$. To do this, we need to induce a sequence of weak learners that help in reducing the risk as optimally as possible. Let $F = \sum_{i=1}^{t-1}\alpha_i f^i$ be the previous solution of the boosting algorithm at step $(t-1)$ and f^t is the currently induced weak learner. We need to identify an appropriate α_t that would lead to an improvement in $\hat{R}_{mod}$. The optimization problem at step t is given by

$$\min_{\alpha_t, f^t}\sum_{ij}\exp\left[-\left(\mathbf{I}(A_{ij}>0) - \frac{k_i k_j}{2m}\right)\left(F_{ij} + \alpha_t f^t(\mathbf{e}_{ij})\right)\right] \tag{14}$$

To highlight the effect of current weak learner we need to isolate the effect of past weak learners from the equation. Let

$$\begin{aligned} D_{ij} &= \exp\left[-\left(\mathbf{I}(A_{ij}>0) - \frac{k_i k_j}{2m}\right)F_{ij}\right] \\ M_{ij} &= \left(\mathbf{I}(A_{ij}>0) - \frac{k_i k_j}{2m}\right)f^t(\mathbf{e}_{ij}) \\ s_{ij} &= \text{sgn}(M_{ij}) \\ W^+ &= \sum_{ij\in M_{ij}>0} D_{ij} \mid M_{ij} \mid \\ W^- &= \sum_{ij\in M_{ij}<0} D_{ij} \mid M_{ij} \mid \end{aligned} \tag{15}$$

It can be shown that the objective function given in (14) is bounded as follows:

$$
\begin{aligned}
& \sum_{ij} \exp\left[-\left(\mathbf{I}(A_{ij} > 0) - \frac{k_i k_j}{2m}\right)\left(F_{ij} + \alpha_t f^t(\mathbf{e}_{ij})\right)\right] \\
= & \sum_{ij} D_{ij} \exp(-\alpha_t s_{ij}|M_{ij}|) \\
\leq & \sum_{ij} D_{ij}|M_{ij}|(\exp(-\alpha_t s_{ij}) - 1) \\
\leq & \ (W^+ \exp(-\alpha_t) + W^- \exp(\alpha_t) - W^+ - W^-) \qquad (16)
\end{aligned}
$$

where the inequality follows from applying Jensen's inequality and the assumption that $|M_{ij}| \leq 1$. For a given f^t taking its partial derivative with respect to α_t gives

$$\alpha_t = \frac{1}{2}\log\frac{W^+}{W^-} \qquad (17)$$

The formula for α_t is similar in spirit to regular AdaBoost in which $\alpha_t = \frac{1}{2}\log\frac{1-e}{e}$, where e is the error rate for the weak classifier. In our case, the W^+ and W_- represent the weighted sum of the correctly classified and incorrectly classified node pairs.

B. Weak Learners

This section describes the construction of the weak learners used in our boosting framework. Similar to traditional boosting, we could apply any simple classifier as long as it takes into consideration the weight matrix D associated with the node-pairs. The weak learner considered in this study is computed based on the nodal attributes and can be computed in closed form.

Let $\mathbf{X}$ represent the $n \times d$ nodal attribute matrix. Given the current weight matrix D between the node pairs, the goal of weak learner is to estimate the $n \times n$ link potential matrix **L(X)** where $L_{ij} = f^t(\mathbf{e}_{ij})$ indicates the strength of link between the nodes i and j. Large positive values of L_{ij} indicate greater potential for link between the nodes and large negative values indicate greater repulsion for link formation between the nodes. We model **L(X)** as simple weighted correlation of the nodal features. Let **L(X)** = $\mathbf{XWX}^T$. Here the weight matrix $\mathbf{W}$ is a $d \times d$ matrix that needs to be estimated by solving the following objective function.

$$\mathcal{Q} = \max_W \sum_{ij}(D_{ij}B_{ij})[XWX^T]_{ij}) - \frac{\lambda}{2} \parallel W \parallel_2^2 \qquad (18)$$

where $B_{ij} = [\mathbf{I}(A_{ij} > 0) - \frac{k_i k_j}{2m}]$ is the coefficient term of the modularity measure or the cost associated with each node pair. Differentiating the objective function we get,

$$\frac{\partial \mathcal{L}}{W_{pq}} = -\sum_{ij} D_{ij}B_{ij}(X_{ip}X_{qj}^T) + \lambda W_{pq} = 0 \qquad (19)$$

We get,

$$
\begin{aligned}
W_{pq} &= \frac{1}{\lambda}\sum_{ij} X_{ip}D_{ij}B_{ij}X_{qj}^T \\
&= \frac{1}{\lambda}\sum_{ij} X_{pi}^T D_{ij}B_{ij}X_{jq} \qquad (20)
\end{aligned}
$$

Let $\bullet$ denote the element wise matrix multiplication, then W can be written as

$$\mathbf{W} = \frac{1}{\lambda}\mathbf{X}^T(B \bullet D)\mathbf{X} \qquad (21)$$

Thus the link potential function **L(X) = XWX**^{T} for the given weight matrix D is given by

$$\mathbf{L} = \frac{1}{\lambda}\mathbf{XX}^T(B \bullet D)\mathbf{XX}^T \qquad (22)$$

A crude interpretation of the above solution is that it aligns the correlation between the nodal attributes with the modularity matrix. λ is chosen as a normalization constant such that the estimated link potentials are mapped between $[-1, 1]$. A distinct aspect of above definition of weak learners is that it does not require explicit conversion of nodal features to edge features. Traditional classifiers like support vector machine or logistic regression requires one to construct feature for each node pair from the nodal features, which itself is a time consuming process.

C. Scalability

Link prediction algorithms such as *Preferential Attachment* and *Common Neighbors*, though often have poor performance, are still considered attractive for many practical applications as they are easy to implement and scalable to large sized networks. Scalability is one of the important aspects of the proposed link prediction algorithm. Even though the number of links in large sparse networks is small, the supervised link prediction algorithm must examine all possible node pairs thereby increasing the size of data to be dealt with. In addition to the number of node pairs, the number of features associated with each node may add severe constraints on the performance of the model with respect to speed, memory requirement, and accuracy.

In this section, we describe an approach to scale up our proposed algorithm by decomposing the network into smaller, potentially overlapping partitions and using the boosting approach to systematically combine the weak learners constructed from each partition. This divide-and-conquer strategy is well suited both for the link prediction problem and the boosting framework since link formation is typically a local phenomenon, in the sense that there are several small communities in the network and the links are formed more inside that community. Thus it is beneficial to construct a local (weak) learner from a small segment of the network at a time and aggregate them in a principled way to form the global model via the boosting formulation.

Algorithm 1 LinkBoost

```
Input: A: $n \times n$ adjacency matrix with $\{+1,-1\}$ entries
       X: $n \times d$ nodal attribute matrix
       $\eta$: threshold for feature partitioning
Output: $\mathbf{F}$: $n \times n$ link potential matrix
Initialize:
  $\mathbf{F} = [0]_{n\times n}$;   $D^{(0)} = [1]_{n\times n}$;
  $\mathbf{k}$: $n \times 1$ column vector of node degrees
for $t = 1$ to $T$ do
    $\mathcal{P} \leftarrow$ GetFeaturePartition($\eta$)
    for $X_p \in \mathcal{P}$ do
        $V \leftarrow$ GetSubGraphNodes($X_p$)
        W $\leftarrow$ GetBaseLearner($X_p, D_v, \mathbf{k}_v, A_v$)
        Compute $f_v = X_p W X_p^T$
        Compute $\alpha$ using (17)
        $\mathbf{F}_v \leftarrow \mathbf{F}_v + \alpha f_v$
        $D_v = D_v \exp(-\alpha A_v \bullet \mathbf{L}_V)$
    end for
end for
return $\mathbf{F}$
```

There are many strategies to create subgraph partitions from a large network. Our requirements are that (1) the partitions must be distinctive enough from each other to induce a diverse (uncorrelated) set of weak learners and (2) the partitioning approach must be efficient to implement especially for large-scale networks. We tried several partitioning strategies (e.g., applying random walk starting from randomly chosen seed nodes) but found that they often fail to satisfy one of the two requirements. This led us to consider the domain partitioning strategy, which is inexpensive to implement and often produces a diverse set of partitions.

The proposed scalable LinkBoost algorithm is summarized in Algorithm 1. The GetFeaturePartition(η) returns a feature partition where each partition set contains $\eta\%$ of the features. For each partition, we create a subgraph containing only those nodes that have at least one non-zero value with respect to the selected set of features. We then build a local model on the subgraph by invoking the GetBaseLearner subroutine. The subroutine takes the following parameters as input: (1) A_v, the adjacency matrix associated with the subgraph induced by the feature partition P, (2) X_p, subset of the nodal attributes in the subgraph, (3) D_v, weights on node pairs in the subgraph, and (4) $\mathbf{k}_v$, global degree of the nodes in the subgraph. The weight returned by the GetBaseLearner subroutine is used to update the estimated link potential matrix. This process is repeated T times on all subgraphs obtained by different feature partitions.

In addition to its efficient implementation and diversity of its induced weak learners, another advantage of the domain partitioning strategy is that the final hypothesis has a nonlinear decision surface. It can be easily seen that the weak learner XWX^T described in Section VI-B yields a linear decision surface separating the linked and non-linked node pairs. Since the boosting algorithm combines the weak learners also in a linear fashion, it will not be able to significantly alter the decision surface. However, by employing domain partitioning in the weak learner construction, we will work with a distinct subgraph at a time. The weight matrix W returned by GetBaseLearner function is applied only to the current subgraph V and not to the entire network. This results in inducing a non-linear decision surface (clipped line) in the feature space (a line with respect to node pairs in current partition and value zero for node pairs outside the partition). Finally the boosting algorithm combines the collection of clipped lines to produce a final classifier with non-linear decision surface.

VII. Experimental Evaluations

This section reports the results of experiments conducted on the proposed LinkBoost algorithm. Since link prediction is cost sensitive in nature, we compare the algorithm against other baseline methods using the receiver operating characteristic (ROC) curve. The curve is obtained by calculating the true positives and false positives by varying the threshold on the estimated link potentials between the node pairs. The link prediction model is built on the training set while the ROC curves are plotted for the node pairs in the test set.

A. Baseline Algorithms

We compared the performance of LinkBoost against the following link prediction algorithms discussed in the related work section.

Link-Based: We used three link based algorithm for link prediction. These are *Preferential Attachment*, *Katz* and *Modularity*. Preferential attachment estimates the link potential between a node pair as product of their degrees. The Katz measure is defined as

$$\texttt{score}(x,y) = \sum_{l=1}^{\infty} \beta^l \mid path_{ij}^{(l)} \mid \tag{23}$$

where $\mid path_{ij}^{(l)} \mid$ is set of all path of length l from node i to node j and $0 < \beta < 1$ is a user parameter. A special variant of Katz is the truncated Katz in which only finite number of terms in the summation are considered. The number of terms to consider is again a user given parameter. The Katz measure is sensitive to both these parameters. The modularity measure for link prediction is computed as follows. Let S_{ir} to be 1 if vertex i belongs to group r and zero otherwise. Then modularity maximization involves identifying a $n \times k$ matrix $\mathbf{S}$ with elements S_{ij} such that following equation is maximized.

$$\max_{\mathbf{S}} \text{tr } \mathbf{S}^T \mathbf{B} \mathbf{S} \tag{24}$$

The problem (24) is NP hard and is relaxed by letting $\mathbf{S}$ to be any real matrix such that $\mathbf{S}^T\mathbf{S} = \mathbf{I}$. We then define $f_{comm}(e_{ij}) = \sum_r S_{ir}S_{jr}$.

Attribute Based: Here we use the well known *Fixed Cost Adaboost* where the cost parameters are set to $C_1 = 1$ and $C_2 = 0.01$. It is not possible to make the cost more than 1 as the derivation of α assumes that $|M_{ij}| \leq 1$. Furthermore, only the cost ratio $\frac{C_1}{C_2}$ matters and not their absolute magnitudes. Similarly, the modularity matrix $\mathbf{B}$ is multiplied by constant factor so that the magnitude of entries are less than 1. For `LinkBoost`, we set $\eta == 0.05$. The base learners used for both `LinkBoost` and *Fixed Cost Adaboost* are the same (see Section VI-B).

B. Data Sets for Inferring Missing Links

Here we consider the problem of inferring missing links from an incomplete network. We use two well-known citation networks[1] [20]—citeseer and cora data sets—for this experiment. In both the data sets, we first make the graph undirected and randomly suppress 30% of the links from the network and use them as the test set for predicting missing links.

Cora Data Set contains publications from the machine learning area, which include the following 7 subcategories: Case-based reasoning, Genetic Algorithms, Neural Networks, Probabilistic Methods, Reinforcement Learning, Rule Learning and Theory. The data set we use contains 2708 nodes, 5429 directed links, and 1433 unique words. Each node corresponds to a paper and is characterized by a 0/1-valued vector indicating the absence/presence of the corresponding word from the title of the paper.

Citeseer Data Set consists of data from 3312 scientific publications. Each publication is labeled as one of 6 classes. The data set we have created contains 4732 links and 3703 unique words.

C. Data Sets for Predicting Future Links

Here, we are given the network link structure and the nodal attributes at a particular time period. Our task is to predict the link formed between the given nodes at a future time.

DBLP Data Set contains all the computer science articles [2] from the proceedings of 28 conferences related to machine learning, data mining and databases from 1997 to 2006. The train set consists of all publications from 1997-2000 and test set contains all publications from 2001-2004. There are 9252 nodes in the train set with 9136 nodal attributes. There are $21,107$ links in the train set and only 6679 links in the test set.

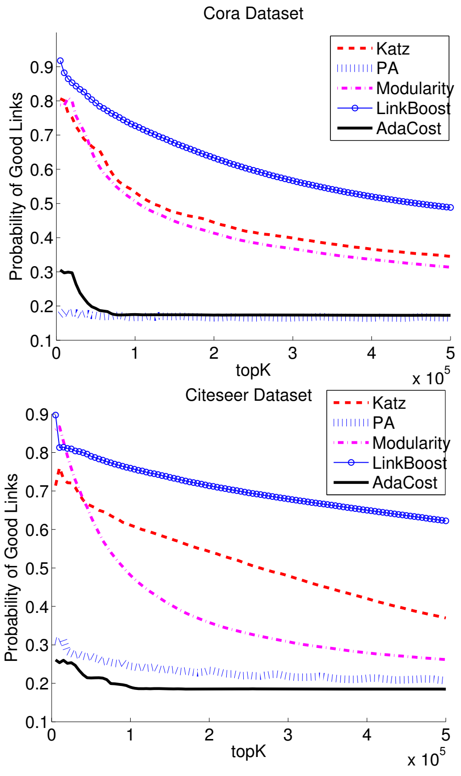
Wikipedia Data Set is a web page network which was crawled from Wikipedia web site by Kossinets[3]. The data set contains edit history of all the pages in Wikipedia from its inception until January 2008. We examined the user-user interaction network (user talk pages). The user interactions in the first 6 months of 2004 is taken as train set and the next 6 months is taken as test set. There are 8178 users and 24891 features for each user.

[1]http://www.cs.umd.edu/projects/linqs/projects/lbc/index.html

[2]http://dblp.uni-trier.de/

[3]G. Kossinets. Processed Wikipedia Edit History. Stanford large network dataset collection.

Figure 1. Proportion of within community links (good links) as function of topK values



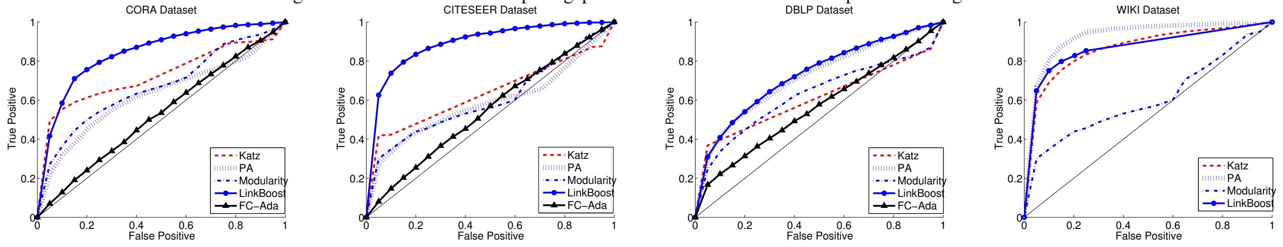
D. Links Within Community

First, we evaluated the performance of the LinkBoost algorithm in terms of its ability to predict links within community. For both cora and citeseer data sets we use the ground truth community label to verify the proportions of links formed within community for each of the link prediction algorithms. We sort the link potentials and declare the top-K largest link potentials as the possible missing or future links. Figure 1 shows the plot of the proportion of within community links or good links as function of top-K values. Clearly, the `LinkBoost` algorithm outperforms both modularity and Katz measures, thus validating the claim that our algorithm indeed strives to identify links within a community. The proportion of good links identified by modularity and Katz are quite high for smaller values of topK, but falls significantly for larger topK values.

Table I
LINK PREDICTION: THE TABLE SHOWS THE AUC OF PREDICTED MISSING LINKS AND FUTURE LINKS IN EACH OF THE FOUR DATA SETS.

		AUC (% improvement compared to LinkBoost)			
Data	Method	Cora	Citeseer	DBLP	Wiki
Link Only	Katz	0.72 (-16.70%)	0.63 (-41.30%)	0.61 (-17.50%)	0.86 (-2.20%)
	PA	0.63(-14.20%)	0.59(-33.70%)	0.71(-4.00%)	0.90(+2.20%)
	Modularity	0.67(-20.00%)	0.60(-32.50%)	0.63(-14.86%)	0.64(-27.27%)
Link+ Content	AdaCost	0.53 ± 0.09(-36.90%)	0.54 ±0.12 (-39.32%)	0.56 ± 0.2 (-24.32%)	<.50
	`LinkBoost`	**.84**± 0.025	**.89**±0.063	**.74**± 0.18	.88± 0.14

Figure 2. ROC curves comparing performances of different link prediction algorithms.



E. Missing and Future Links

Next we evaluate the performance of `LinkBoost` for the missing and future link prediction problems. The ROC curves are shown in Figure 2. Firstly, notice that the AUC of `LinkBoost` is consistently higher than *modularity* measure. As mentioned earlier, modularity utilizes only the network link structure whereas boosting makes use of both the link and the content information thus resulting in superior performance. The `LinkBoost` consistently outperforms the fixed cost Adaboost as well, highlighting the importance of the proposed variable cost structure.

`LinkBoost` outperforms the Katz measure on both cora and citeseer citation networks. The Katz measure performs better than the fixed cost Adaboost on the DBLP network and as good as `LinkBoost` on Wikipedia network. However it is sensitive to choice of parameter setting. In this paper, we report the results based on the parameters that best fits the test set.

Finally, `LinkBoost` outperforms the preferential attachment measure on both the citation networks. However it is performance is comparable to preferential attachment on DBLP and Wiki networks. Specifically, `LinkBoost` is slightly better than preferential attachment on DBLP network and is slightly worse on Wikipedia network. This is because the preferential attachment algorithm is based on the premise that the *rich gets richer*. We suspect that the user network in Wikipedia exhibit the preferential attachment characteristics where few authoritative users communicate with large number of other users. The average AUC for `LinkBoost` is 0.88 and for preferential attachment is 0.90.

F. Low Degree Nodes

In this section, we demonstrate the ability of the proposed method to identify the links formed between low degree nodes in the citation networks. A node with degree less than 2 is considered to be a low degree node. We compute the models on the training set and estimate the ROC curves for the subgraph consisting of low degree nodes. The results are plotted in Figure 3. As expected, the preferential attachment measure under performs as it ranks the high degree nodes ahead of the low degree nodes. The proposed `LinkBoost` with effective degree sensitive loss function overcomes this problem.

VIII. CONCLUSION

In this paper, we have given a new direction for the supervised link prediction problem in large sparse networks. We have proposed a new degree dependent cost function and has shown that minimization of the associated risk leads to modular link prediction where more links are predicted within community. Such a cost function addresses the skewness in class distribution and skewness in nodal degrees. The proposed algorithm is scalable and easy to implement. Experimental evaluations show the superior performance of the proposed method over existing supervised and unsupervised methods. The proposed method is specially effective in predicting the missing links for the low degree nodes. For future work, we plan to investigate methods for estimating optimal cost parameters and alternate ways for creating the weak learners used in LinkBoost formulation.

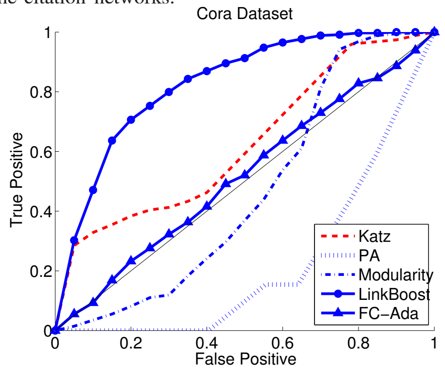
IX. ACKNOWLEDGEMENT

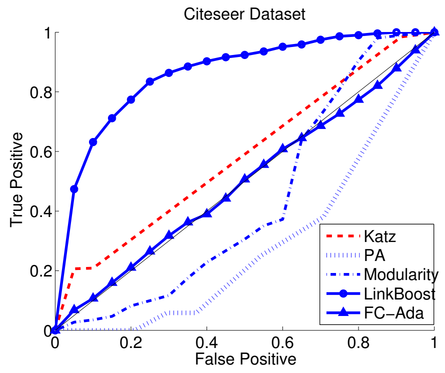
This material is based upon work supported in part by ONR grant number N00014-09-1-0663.

REFERENCES

[1] L. Adamic and E. Adar. Abstract how to search a social network, 2005.

Figure 3. ROC curves comparing performances of different missing link prediction algorithms on the subgraph induced by the low degree nodes in the citation networks.





[2] A.-L. Barabasi and R. Albert. Emergence of scaling in random networks. In *Science*, 286, 1999.

[3] M. Bilgic, G. Namata, and L. Getoor. Combining collective classification and link prediction. In *Proceedings of the ICDM Workshop on Mining Graphs and Complex Structures*, pages 381–386, 2007.

[4] Chebotarev and E. V. Shamis. The matrix-forest theorem and measuring relations in small social groups. In *Automation and Remote Control*, 1997.

[5] A. Clauset, C. Moore, and M. Newman. Structural inference of hierarchies in networks. In *Intl. Conf. on Machine Learning, ICML*, 2006.

[6] A. Clauset, M. Newman, and C. Moore. Finding community structure in very large networks. In *Phys. Rev. E*, number 6, page 066111, 2004.

[7] M. Collins, R. E. Schapire, and Y. Singer. Logistic regression, adaboost and bregman distances. *Machine Learning*, 48(1-3), 2002.

[8] R. Guimera and M. Sales-Pardo. Missing and spurious interactions and the reconstruction of complex networks. In *Proc. Natl. Acad. Sci. U.S.A.*, 2009.

[9] M. A. Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In *In Proc. of SDM 06 workshop on Link Analysis, Counterterrorism and Security*, 2006.

[10] H. Kashima and N. Abe. A parameterized probabilistic model of network evolution for supervised link prediction. In *Proceedings of the ICDM*, pages 556–559, New York, NY, USA, 2006. ACM.

[11] H. Kashima, T. Kato, Y. Yamanishi, M. Sugiyama, and K. Tsuda. Link propagation: A fast semi-supervised learning algorithm for link prediction. In *Proceedings of the SIAM intl conf on Data Mining*, pages 1099–1110, Sparks, NV, USA, 2009.

[12] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, VOL. 18, NO. 1:39– 43, 1953.

[13] D. Liben-Nowell and J. Kleinberg. The link prediction problem for social networks. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 556–559, New York, NY, USA, 2003. ACM.

[14] H. Masnadi-Shirazi and N. Vasconcelos. Cost-sensitive boosting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:294–309, 2011.

[15] M. E. J. Newman. Clustering and preferential attachment in growing networks. *Phys. Rev. E*, 64, 2001.

[16] S. Purnamrita, C. Deepayan, and M. Andrew. Theoretical justification of popular link prediction heuristics. In *COLT*, 2010.

[17] M. Rattigan and D. Jensen. The case for anomalous link discovery. *SIGKDD Explorations*, 7(2):41–47, 2005.

[18] G. Salton and M. J. McGill. Introduction to modern information retrieval. In *McGraw-Hill, Auckland*, 1983.

[19] J. Scripps, P. Tan, F. Chen, and A.-H. Esfahanian. A matrix alignment approach for link prediction. In *Proceedings of the 19th International Conference on Pattern Recognition*, pages 1–4, 2008.

[20] P. Sen, G. M. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective classification in network data. *AI Magazine*, 29(3):93–106, 2008.

[21] B. Taskar, M. Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In *In Advances in Neural Information Processing Systems 16*, 2003.

[22] H. Tong, C. Faloutsos, and J. Y. Pan. Fast random walk with restart and its applications. In *ICDM '06: Proceedings of the Sixth International Conference on Data Mining*, pages 613–622, 2006.

[23] Y.-B. Xie, T. Zhou, and B.-H. Wang. Scale-free networks without growth. In *Physica*, volume 387, 2008.

[24] T. Zhou, L. Lu, and Y.-C. Zhang. Predicting missing links via local information. In *Eur. Phys. J.*, 2009.