Learning Algorithms for Link Prediction based on Chance Constraints

Janardhan Rao Doppa[1], Jun Yu[1], Prasad Tadepalli[1] and Lise Getoor[2]

[1] School of EECS, Oregon State University, Corvallis, OR 97330 USA
{doppa, yuju, tadepall}@eecs.orst.edu
[2] Computer Science Dept., University of Maryland, College Park, MD 20742 USA
{getoor}@cs.umd.edu

Abstract. In this paper, we consider the link prediction problem, where we are given a partial snapshot of a network at some time and the goal is to predict the additional links formed at a later time. The accuracy of current prediction methods is quite low due to the extreme class skew and the large number of potential links. Here, we describe learning algorithms based on chance constrained programs and show that they exhibit all the properties needed for a good link predictor, namely, they allow *preferential bias* to positive or negative class; handle *skewness* in the data; and *scale* to large networks. Our experimental results on three real-world domains—co-authorship networks, biological networks and citation networks—show significant performance improvement over baseline algorithms. We conclude by briefly describing some promising future directions based on this work.

1 Introduction

Network analysis, performed in domains including social networks, biological networks, transaction networks, and the web, has received a lot of interest in recent years. These networks evolve over time and it is a challenging task to understand the dynamics that drives their evolution. Link prediction is an important research direction within this area. The goal here is to predict the potential future interaction between two nodes, given a partial view of the current state of the network.

This problem occurs in several domains. In many cases, we are interested in the links that are likely to form in the future. For example, in citation networks describing collaboration among scientists, we want to predict which pairs of authors are likely to collaborate in future; in social networks, we would want to predict new friendships; in query graphs, we want to predict the related queries in the context of web search and in biological networks we want to predict which proteins are likely to interact. On the other hand, we may be interested in anomalous links; for example, in financial transaction networks, the unlikely transactions might indicate fraud, and on the web, they might indicate spam.

There is a large literature on link prediction. Early approaches to this problem are based on defining a measure for analyzing the *proximity* of nodes in the network [1, 19, 14]. For example, shortest path, common neighbors, Katz measure, Adamic-adar etc., all fall under this category. More recently, Sarkar et al. [22] gave a theoretical justification of these link prediction heuristics. Liben-Nowell and Klienberg studied the usefulness of all these topological features by experimenting on bibliographic datasets

[14]. It was found that, no one measure is superior in all cases. Statistical relational models were also tried with some success [7, 8, 24, 20]. Recently, the link prediction problem is studied in the supervised learning framework by treating it as an instance of binary classification [9, 11, 4, 25, 27]. These methods use the topological and semantic measures defined between nodes as features for learning classifiers. Given a snapshot of the social network at time t for training, they consider all the links present at time t as positive examples and consider a large sample of absent links (pair of nodes which are not connected) at time t as negative examples. The learned classifiers performed consistently across all the datasets unlike heuristic methods which were inconsistent, although the accuracy of prediction is still very low. There are several reasons for this low prediction accuracy. One of the main reasons is the huge class skew associated with link prediction. In large networks, it's not uncommon for the prior link probability on the order of 10^{-4} or less, which makes the prediction problem very hard, resulting in poor performance. In addition, as networks evolve over time, the negative links grow quadratically whereas positive links grow only linearly with new nodes. Further, in some cases we are more concerned with *link formation*, the problem of predicting new positive links, and in other cases we are more interested in *anomalous link detection* [21], the problem of detecting unlikely links. In general, we need the following properties for a good link predictor: allow *preferential bias* to the appropriate class; handle *skewness* in the data; *scale* to large networks.

Chance-constraints and Second-Order Cone Programs(SOCPs) [15] are a special class of convex optimization problems that have become very popular lately, due to the efficiency with which they can be solved using fast interior point methods. They are used in a variety of settings such as feature selection [3], dealing with missing features [23], classification and ordinal regression algorithms that scale to large datasets [18], and formulations to deal with unbalanced data [17, 10]. In this work, we give a scalable cost-sensitive formulation based on chance-constraints which satisfies all the requirements needed for learning a good link predictor mentioned above and show how it can be used for link prediction to significantly improve performance. The chance constraints can be converted into deterministic ones using Chebyschev-Cantelli inequality, resulting in a SOCP. The complexity of SOCPs is moderately higher than linear programs and they can be solved using general purpose SOCP solvers like SeDuMi[3] or YALMIP[4].

The main contributions of this paper include: 1. We identify important requirements of the link prediction task and propose a new cost-sensitive formulation based on chance constraints satisfying all the requirements. We describe its connections to other frameworks like biased classification and uneven margin algorithms. 2. We perform a detailed evaluation on multiple datasets from three real-world domains–co-authorship networks, biological networks and citation networks– to investigate the effectiveness of our methods. We show significant improvement in link prediction accuracy.

2 Cost-sensitive learning for Link Prediction

In this work, we consider the link prediction problem as an instance of binary classification. We are given training data $\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)\}$ where, each $x_i \in \Re^n$ is a feature vector defined between two nodes and $y_i \in \{-1, +1\}$ is the

[3] http://sedumi.ie.lehigh.edu
[4] http://users.isy.liu.se/johanl/yalmip/

corresponding label that stands for the presence or absence of an edge between the two nodes. In our case, we have a huge class imbalance problem, i.e., the number of negative examples $\gg$ the number of positive examples. There are two different ways of addressing the class imbalance problem. In the first approach, it is turned into a balanced problem either by over-sampling the minority class or under-sampling the majority class. However, both these sampling methods have their drawbacks. By doing under-sampling, we lose some information and over-sampling introduces noise into the data. In the second approach, class imbalance problem is addressed by reducing it to a cost-sensitive learning problem where misclassification costs are unknown. Then, the ratio of misclassification costs is varied to find out the best decision function based on the validation set. In this work, we will follow the second approach which is considered to be more principled. In particular we are interested in a cost-sensitive formulation in the max-margin framework. We require a solution which is scalable to large data sets; this is very important for the link prediction task. For now, we work with only linear decision functions of the form $f(x) = w^T x - b$. However, all the formulations described in this work can be kernelized to construct non-linear classifiers.

Cost-Sensitive Learning Problem: In the traditional binary classification problem, all misclassifications are considered to be of the same cost, i.e., $\mathcal{C}_{12} = \mathcal{C}_{21}$ where, $\mathcal{C}_{12}$ is the misclassification cost of predicting a data point of class 1 as class 2 and $\mathcal{C}_{21}$ the misclassification cost of predicting a data point of class 2 as class 1. However, this assumption is not true for many real-world applications like medical domains e.g., predicting whether a patient has breast cancer or not. In these problems, some mistakes are considered more costly than others and are studied under *cost-sensitive* framework. In a cost-sensitive learning problem, we are given a set of training examples, along with the misclassification costs. The goal of learning is to find a hypothesis that minimizes the expected cost of misclassification.

3 Clustering-based Cost-Sensitive formulation

In this formulation, we assume that class conditional densities of positive and negative points can be modeled as mixture models with component distributions. Let k_1 and k_2 denote the number of components in the mixture model for positive and negative class respectively and say $k = k_1 + k_2$. We can cluster the positive and negative points separately, and estimate the first and second order moments (μ, Σ) of all the clusters. Given these second order moments, our goal is to find a discriminating hyperplane $w^T x - b = 0$, which separates these positive and negative clusters in such a way that it minimizes the expected cost of misclassification. To this end, consider the following formulation:

$$\begin{aligned} \min_{w,b,\eta_i} \quad & \frac{1}{2}\|w\|_2^2 \; + \; \mathcal{C}_{reg}\left\{\mathcal{C}_{12}\sum_{i=1}^{k_1}\eta_i + \mathcal{C}_{21}\sum_{i=k_1+1}^{k}\eta_i\right\} \\ \text{s.t. } & Pr(X_i \in \mathcal{H}_2) \le \eta_i, : \forall i = 1,\cdots,k_1 \\ & Pr(X_i \in \mathcal{H}_1) \le \eta_i : \forall i = k_1+1,\cdots,k \\ & 0 \le \eta_i \le 1 : \forall i = 1,\cdots,k \end{aligned} \tag{1}$$

Here $X_i, \forall i = 1, \cdots, k_1$ and $X_i, \forall i = k_1 + 1, \cdots, k$ are random variables corresponding to the components of the mixture models for positive and negative classes respectively; $\mathcal{H}_1$ and $\mathcal{H}_2$ denote the positive and negative half spaces i.e., $\mathcal{H}_1(w, b) = \{x | w^T x - b \geq 0\}$ and $\mathcal{H}_2(w, b) = \{x | w^T x - b \leq 0\}$; η_i stands for the probability with which any point drawn from a mixture component lies on the wrong side of the hyperplane. The objective function consists of two terms: the first term $\frac{1}{2} \|w\|_2^2$ is the standard squared-norm regularizer and second term $\mathcal{C}_{12} \sum_{i=1}^{k_1} \eta_i + \mathcal{C}_{21} \sum_{i=k_1+1}^{k} \eta_j$ is the empirical expected cost of misclassification. $\mathcal{C}_{reg}$ is the regularization parameter that controls the trade off between empirical error and generalization error.

The above probabilistic constraints can be written as deterministic constraints using multivariate Chebyshev-Cantelli inequality [10].

3.1 Conversion of Chance-constraint to Second-Order Cone constraint

This conversion can be done in several different ways [12, 10]. We present the variant based on a multi-variate generalization of Chebyschev-Cantelli inequality [16] which is stated below.

Theorem 1. *Let Z be a random variable whose second order moments are (μ, σ^2). Then for any $t > 0$,*

$$Pr(Z - \mu \geq t) \leq \tfrac{\sigma^2}{\sigma^2 + t^2}$$

We can use the above theorem to do this conversion. Let X be an n-dimensional random variable with second order moments (μ, Σ). By applying the above theorem to random variable $-w^T x$, $w \in \Re^n$ and with $t = w^T \mu - b$, we get

$$Pr(-w^T X \geq -b) \leq \frac{w^T \Sigma w}{w^T \Sigma w + (w^T \mu - b)^2} \tag{2}$$

Now, satisfying the constraint $Pr(w^T X - b \geq 0) \geq \eta$ is same as satisfying $Pr(-w^T X \geq -b) \leq 1 - \eta$. By applying Theorem 1, we can satisfy $Pr(-w^T X \geq -b) \leq 1 - \eta$ if:

$$\frac{w^T \Sigma w}{w^T \Sigma w + (w^T \mu - b)^2} \leq 1 - \eta \tag{3}$$

Re-arranging the terms in the above inequality gives us:

$$w^T \mu - b \geq \kappa \sqrt{w^T \Sigma w} \tag{4}$$

where, $\kappa = \sqrt{\frac{\eta}{1-\eta}}$.

3.2 Separable Case

By using the above conversion with $X = X_i$ and $\eta = 1 - \eta_i$ and re-writing it in the standard SOCP form, we get the following formulation:

$$\begin{aligned}
\min_{w,b,\eta_i} \quad & \mathcal{C}_{12}\sum_{i=1}^{k_1}\eta_i + \mathcal{C}_{21}\sum_{i=k_1+1}^{k}\eta_j \\
\text{s.t. } & w^T\mu_i - b \geq 1 + \kappa_i\sqrt{w^T\Sigma_i w} : \forall i = 1,\cdots,k_1 \\
& b - w^T\mu_i \geq 1 + \kappa_i\sqrt{w^T\Sigma_i w} : \forall i = k_1+1,\cdots,k \\
& 0 \leq \eta_i \leq 1 : \forall i = 1,\cdots,k \\
& W \geq \|w\|_2
\end{aligned} \tag{5}$$

where, $\kappa_i = \sqrt{\frac{1-\eta_i}{\eta_i}}$; W is a user-defined parameter which plays similar role as $\mathcal{C}_{reg}$ in the previous formulation. The geometric interpretation of the above constraints is that of finding a hyperplane which separates the positive and negative ellipsoids whose centers are at μ_i, shapes determined by Σ_i, and the sizes of the ellipsoids, i.e., κ_i (see Figure 1) to be classified correctly in order to minimize the expected cost of misclassification.

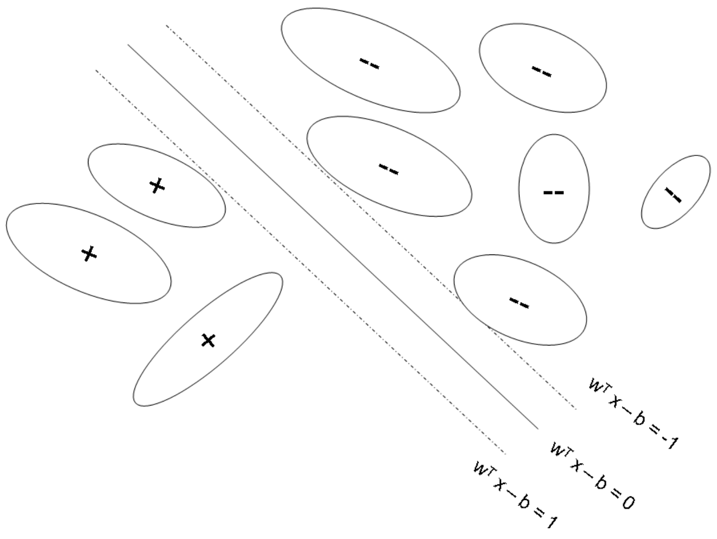


Fig. 1: Geometric interpretation of SOCP formulation

3.3 Non-Separable Case

In the above formulation, if the means of the clusters are not separable, then the optimization problem is infeasible. For example, in the worst case say η_i is 1 for some of the non-separable ellipsoids; but even in this worst case the constraints require the means μ_i of these ellipsoids to lie on the correct side of the hyperplane, i.e., $w^T\mu_i - b \geq 1$ and $w^T\mu_i - b \geq -1$. To avoid this problem, we can introduce slack variables ξ_i as in soft-margin SVM formulation and fix the values of η_1 and η_2, the false-positive and false-negative probabilities, to very small values (say 0.1). Note that, η_1 and η_2 are shared by all the clusters of positive and negative classes respectively. In this case the objective function will be replaced with slack variables ξ_i instead of η_i in the separable case and leads to the following formulation:

$$
\begin{aligned}
\min_{w,b,\eta_i} \quad & \mathcal{C}_{12}\sum_{i=1}^{k_1}\xi_i + \mathcal{C}_{21}\sum_{i=k_1+1}^{k}\xi_j \\
\text{s.t. } & w^T\mu_i - b \geq 1 - \xi_i + \kappa_1\sqrt{w^T\Sigma_i w} : \forall i = 1, \cdots, k_1 \\
& b - w^T\mu_i \geq 1 - \xi_i + \kappa_2\sqrt{w^T\Sigma_i w} : \forall i = k_1+1, \cdots, k \\
& \xi_i \geq 0 : \forall i = 1, \cdots, k \\
& W \geq \|w\|_2
\end{aligned}
\tag{6}
$$

We can see that cost-sensitive SVM is now a special case of this formulation when we consider each data point as a cluster, i.e., the covariance matrix is null. By solving the above SOCP problem using standard SOCP solvers like SeDuMi, we get the optimum values of w and b, and a new data point x can be classified as $\text{sign}(w^T x - b)$.

3.4 Unbalanced data

In the case of skewed class distribution, one class will have more representative data points (majority class) when compared to the other class (minority class). We can handle the unbalanced problem in three different ways.

1) Cost-Sensitive classification: we can transform the unbalanced problem into a cost-sensitive learning problem where costs are unknown and by varying the costs based on a validation set to find the best discriminating hyperplane(CS-SOCP). More specifically, we need to vary the ratio $\mathcal{C}_{min}/\mathcal{C}_{maj}$ where, $\mathcal{C}_{min}$ and $\mathcal{C}_{maj}$ corresponds to the misclassification costs of minority and majority class.

2) Biased classification: we can vary the preferential bias for each class η_1 and η_2 instead of varying the misclassification costs and try to find a maximum-margin hyperplane in the biased classification framework (B-SOCP) [17].

3) Classification with Uneven margins: we can vary the positive margin (τ_+) and negative margin (τ_-) to find the best decision function in the Uneven Margin framework (PAUM) [13]. In the uneven margin setting, the constraints in the above formulation will become $w^T\mu_i - b \geq \tau_+ - \xi_i + \kappa_1\sqrt{w^T\Sigma_i w}$ and $b - w^T\mu_i \geq \tau_- - \xi_i + \kappa_2\sqrt{w^T\Sigma_i w}$ for positive and negative clusters respectively.

We will empirically evaluate these three frameworks for different kinds of link prediction problems.

3.5 Advantages of CCP for Link Prediction

There are several advantages of using chance-constrained programs for the link prediction.

Scalability: The SOCP formulation based on chance constraints is scalable to large datasets because the number of constraints in this formulation is linear in the number of clusters, whereas the number of constraints in the SVM formulation (QP problem) is linear in the number of data points. In addition, there are very efficient interior point algorithms for solving SOCP.

Missing Links: As described before, we consider a large sample of node pairs which are not connected at time t as negative examples. However, some of these negative examples may be noisy, i.e., the link may exist, but was simply not observed at time t. In the case of SVMs the gemoetric interpretation of dual is that of finding the distance between two convex hulls corresponding to the positive and negative points respectively [2]. Conversely, the interpretation of dual for SOCP formulation is that of finding distance between two convex hulls corresponding to the positive and negative ellipsoids. In the presence of noisy labels, The SVM solution is much more sensitive to noisy labels than the solution with ellipsoids.

Missing features: Chance-constrained programs can naturally handle missing features [23]. The key idea here is to use chance constraints to deal with uncertainty in the missing values based on the second order moments. The Gaussian assumption allows us to use EM to impute the missing values. The resulting formulation again leads to an SOCP problem.

Applications: We can use this framework for several applications like recommendations, collaborative filtering, online advertisement and marketing, and anomalous link discovery in financial networks, terrorist networks, power grids and disease transmission networks.

4 Experimental Results and Discussion

In this section, we describe our experimental setup, description of datasets, features used for learning the classifier, evaluation methodology, followed by our results and discussion.

4.1 Datasets:

We use three different kinds of real-world domains namely co-authorship networks, biological networks, and citation networks for evaluating our learning algorithms.

Co-authorship networks. In co-authorship networks, we want to predict which pair of authors are likely to collaborate in future. We use two different co-authorship networks:

1) DBLP dataset: we use a dataset which was generated using DBLP collection of computer science articles [5], and contains all the papers from the proceedings of 28 conferences related to machine learning, data mining and databases from 1997 to 2006.

2) Genetics dataset: The *genetics* dataset contains articles published in 14 journals related to genetics and molecular biology from 1996 to 2005. The genetics dataset was generated from the popular PubMed database[6].

For each dataset we have the data for 10 years. We consider the data from first 9 years for training and the data from the 10th year for testing. We consider all the links

[5] http://dblp.uni-trier.de/
[6] http://www.ncbi.nlm.nih.gov/entrez

formed in the 9th year as positive training examples and among all the negative links (those links that are not formed in the first 9 years), we randomly collect a large sample and label them as negative training examples. Note that the features of these training examples are constructed based on the first 8 years of data. Similarly for the test set, we consider all the links that are formed during the 10th year as positive examples and collect a sample of all the negative links as negative examples. Also the features of these testing examples are constructed based on the first 9 years of data.

Biological networks. We use two biological networks, a protein-protein interaction network[7] and a metabolic network[8]. The details are described below:

1) Metabolic network: This network contains 668 nodes and 2782 links. In the metabolic network, proteins are represented as nodes, and an edge indicates that the two proteins are enzymes that catalyze successive reactions between them. This dataset has several features for each protein based on gene expression, localization, phylogenetic profiles and chemical compatibility along with some kernel features as well.

2) Protein-protein interaction network: This network contains 2617 nodes and 8844 edges. Each protein is described by a 76 dimensional feature vector, where each feature indicates whether the protein is related to a particular function or not.

Since we do not have temporal information for either of these networks, we will choose a random two thirds of the data for training and the remaining one third for testing.

Citation networks. For the citation prediction task, we used the KDD Cup 2003[9] dataset which contains the citation network for both training and testing periods separately. Also for each paper we have all the information including the title, authors, abstract and textual content of the paper. We consider two different kinds of prediction tasks.

1) Complete bibliography prediction: Given a new paper we want to predict the complete bibliography of the paper, i.e., all those papers in the training which will be cited by this paper. In this task, we connect this new paper to all the previous papers written by its authors before the prediction for constructing features.

2) Partial bibliography prediction: In this task, given a new paper and its partial bibliography, we want to predict the remaining entries.

We sample roughly 10 times the number of positive links from the pool of absent links resulting in a positive to negative class ratio of 1:10. The exact number of positive and negative examples used for different link prediction tasks are shown in Table 1.

4.2 Feature description

We use two different kinds of features between two nodes, namely *content features* and *structural features*.

The *content feature function* $\phi_{cont} : \Re^d \times \Re^d \mapsto \Re^n$ is defined based on the attributes of the two nodes. For example, in the case of co-authorship networks the features of each author corresponds to occurrences of a particular word in the author's papers. The content feature function could be the kronecker product of these binary vectors. Similarly for friend recommendation problems in social networks, content features will

[7] http://noble.gs.washington.edu/proj/maxent/
[8] http://web.kuicr.kyoto-u.ac.jp/supp/yoshi/ismb05/
[9] http://www.cs.cornell.edu/projects/kddcup/datasets.html

Prediction Task	TRAINING		TESTING	
	# positives	# negatives	# positives	# negatives
DBLP	1404	14040	1021	10210
Genetics	2422	24220	3017	30170
Metabolic network	618	6180	928	9280
Protein network	1966	19660	2948	29480
Complete citation	3000	30000	3000	30000
Partial citation	3000	30000	3000	30000

Table 1: Details of training and testing data for different link prediction tasks

be defined based on the user profiles – geographic location, college/university, work place, hobbies/interests, browsing/navigation history on the network etc. In the case of protein-protein networks, content features can be defined as the Kronecker product of the features of each protein. Therefore, weights on each of these kronecker features will tell us how likely those proteins will interact.

The *structural feature function* $\phi_{struct} : G_{n_1,n_2} \mapsto \Re^m$ is defined over the local subgraph around the two nodes n_1 and n_2. One can also call them relational features, e.g., approximate Katz measure which is calculated on the ensemble of paths between two nodes (say upto depth 4), number of common neighbors, social connectivity which shows how connected these two nodes are with the other nodes in their neighborhood etc., which are meaningful for each network. For example, in the citation prediction task the network between papers and authors is very complex, i.e., links are between one paper and another–*paper1 cites paper2*, and between an author and a paper–*author1 writes paper1*. Therefore, these complex multi-way relationships could be used to define relational features which will be useful for our link prediction task.

4.3 Evaluation

We use the precision and recall metrics from Information Retrieval context for evaluation, and compare the chance-constraints based algorithms, namely cost-sensitive SOCP (CS-SOCP), biased SOCP (B-SOCP) against cost-sensitive SVMs[10] (CS-SVM) and perceptron with uneven margins (PAUM) [13]. We rank all the test examples according to the margin of the classifiers and calculate precision and recall from *top-k* by varying the value of k. Here, precision is defined as the percentage of true-positive links that are predicted correctly among the *top-k* and recall is defined as the percentage of true-positive links that are predicted correctly out of the total true-positive links. Note that, majority of the applications of link prediction algorithms are in recommendation systems like movie recommendations in Netflix, music recommendation engines like *last.fm*, friends suggestions in social networks etc. Therefore, link prediction algorithms should be evaluated based on the quality of the top-k recommendations produced by them. According to the above definitions of precision and recall, precision need not always monotonically decrease with k. We report the precision and recall curves by varying the value of k along the x-axis. We also report the AUC values calculated for top 20% of the total testing links (see Table 2).

[10] LIBSVM with -wi option to specify costs

	CS-SOCP	B-SOCP	CS-SVM	PAUM
DBLP	**0.4019**	0.3707	0.3186	0.0682
Genetics	**0.2314**	0.1981	0.1526	0.0638
Metabolic	0.619	0.6183	**0.6447**	0.0816
Protein	0.2754	**0.2786**	0.2471	0.1274
Complete citation	**0.3684**	0.3186	0.3252	0.3586
Partial citation	0.4994	0.469	**0.5356**	0.3607

Table 2: AUC values for top 20% of the total testing links for different learning algorithms

We use $k_1 = k_2 = 100$ clusters for all clustering-based SOCP formulations and k-means++[11], a variant of k-means algorithm which is fast and proven to be near optimal for clustering in our experiments. We observe that the number of clusters will not make much difference in the results as long as they are not too small a number of clusters. As the number of clusters increases SOCP based algorithms will tend to move closer towards their SVM counterparts. Note that, SOCP and SVM based algorithms are exactly the same when we consider each data point as a cluster, i.e., the covariance matrix is null. We use diagonal covariance for our SOCP experiments. We report the best results for CS-SVM and CS-SOCP by varying the ratio $\mathcal{C}_+/\mathcal{C}_-$ on validation set. Similarly, we give the best results for B-SOCP by varying η_1 and η_2 . For PAUM, we pick the best values for τ_- from $\{-1.5, -1, -0.5, 0, 0.1, 0.5, 1\}$ and for τ_+ from $\{-1, -0.5, 0, 0.1, 0.5, 1, 2, 5, 10, 50\}$ based on the validation set. We run PAUM for a maximum of 1000 iterations or until convergence.

	Training time		Classification time	
	CS-SVM	CS-SOCP	CS-SVM	CS-SOCP
DBLP	39.68s	0.69s	7.64s	0.46s
Genetics	3m 34s	9s	1m 44s	27s
Metabolic	15.1s	4.89s	7.31s	4.29s
Protein	42m 23s	56.64s	1m 53s	19.64s
Complete citation	3m 17.6s	8.92s	1m 16.6s	13.92s
Partial citation	5m 19.5s	10.21s	1m 3.3s	13.78s

Table 3: Training and classification time results

4.4 Results and Discussion

The precision and recall curves for all the 6 datasets are shown in Figures 2,3 and 4. As we can see, both CS-SOCP and B-SOCP outperform CS-SVM in precision and recall for majority of the datasets namely, DBLP, genetics, complete and partial bibliographic prediction tasks. Particularly, they achieve significantly higher recall on the complete bibliography prediction task (72.4% and 66.16% compared to 52.53% of CS-SVM) and

[11] http://en.wikipedia.org/wiki/K-means++

partial bibliographic prediction task (82.96% and 75.13% compared to 70.13% of CS-SVM). Similarly, if we look at the AUC values in Table 2, SOCP based algorithms significantly outperform the other algorithms on 4 out of 6 prediction tasks, including the protein-protein interaction network which is a very high-dimensional dataset. We conjecture that noisy labels for the missing links (explained in the previous section) might have contributed to the bad performance of CS-SVM in both these tasks. Also note that the prediction accuracies significantly improve in the case of partial prediction task compared to the complete prediction task because of additional information in the form of partial references of each paper. These results show the strength of rich information present in the link structure. It is important to note that, even in the other cases like metabolic and protein networks, performance of SOCP formulations are comparable to CS-SVM. In our experiments, we noticed that behavior of PAUM was not consistent across all the datasets. For example, it had the best performance for complete bibliographic prediction task and worst performance for the metabolic network. This may be partly due to our restricted search over the margin space. It appears that varying costs or probabilities might be easier than varying margins to handle the problem of unbalanced data. Since one of the main advantages of SOCP based formulations is scaling, we report the training and classification time[12] of both CS-SVM and CS-SOCP for all the datasets in Table 3 (m stands for mins and s for secs). Note that, training time for CS-SOCP includes clustering time and time taken to solve the SOCP problem. We can see that CS-SOCP is orders of magnitude faster than CS-SVM. Furthermore, CS-SOCP requires less time for classification when compared to that of CS-SVM. Since the learned link predictors need to be deployed in real-time systems like recommendation engines, it is important to have low test time computational cost. Note that, the classification time in SVMs is proportional to the number of support vectors and support vectors grow linearly with size of the data. On the other hand, the number of support vectors in CS-SOCP is bounded by the number of clusters k and does not depend on the size of the data.

5 Conclusions and Future Work

In this work, we proposed a new cost-sensitive formulation based on chance constraints and described its connections to other frameworks like biased classification and uneven margin algorithms. We showed how learning algorithms based on chance-constraints can be used to solve different kinds of link prediction problems and showed empirical evidence with experiments on several real-world datasets. It is interesting to note that we could formulate link-prediction as a complex structured prediction problem with exponential number of constraints. The manner in which the absent links are sampled to be used as negative examples for our classification problem, is roughly equivalent to randomly sampling the constraints for the structured prediction problem [6, 5]. We believe that this is a very fruitful direction towards solving some of these hard problems. Wick et al. use similar ideas for their SampleRank algorithm and got some impressive results [26]. In future, we would like to extend the current framework to a relational setting similar to Taskar's work [24]. However, formulating it as relational or structured prediction poses an enormous inference problem, especially in large networks.

[12] All experiments were run on a machine with 2GB RAM and 2.16 GHz Intel dual core processor

One possible approach is to take a middle path between complete independence and arbitrary relational structure.

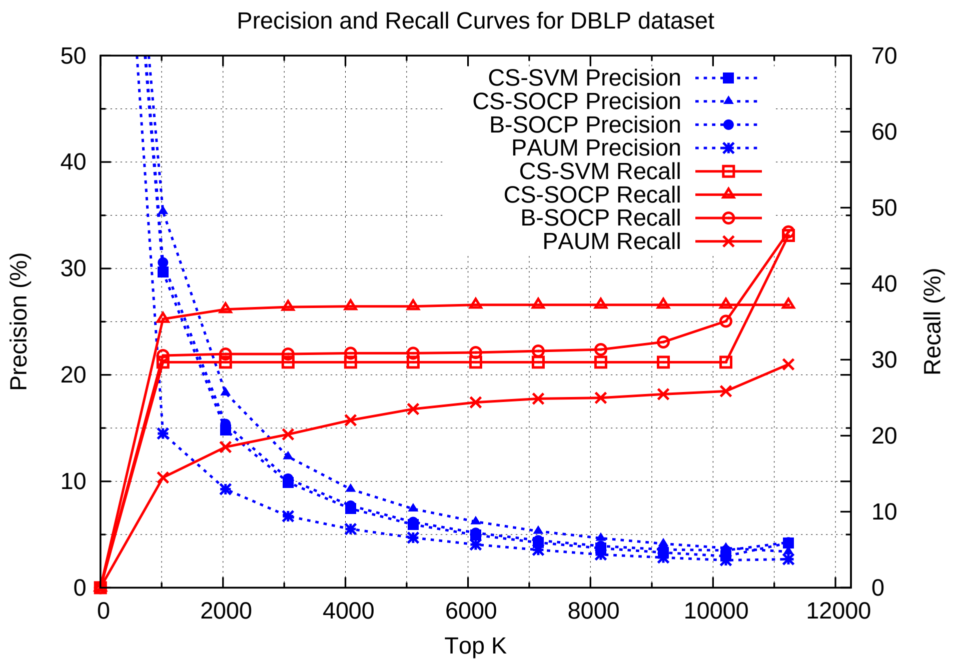
6 Acknowledgements

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. FA8750-09-C-0179. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the DARPA, or the Air Force Research Laboratory (AFRL). This work is also partially funded by NSF Grant No. 0746930. We would like to thank Saketha Nath for useful discussions which helped our understanding on chance-constrained programs.

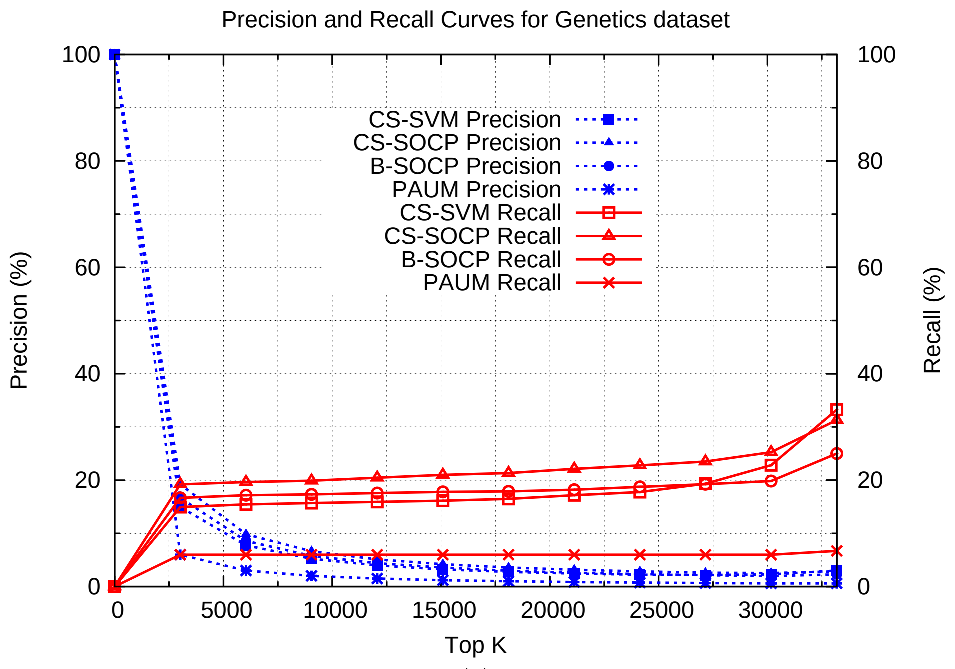
References

1. L. Adamic and E. Adar. Friends and neighbors on the web. *Social Networks*, 25:211–230, 2001.
2. K. P. Bennett and E. J. Bredensteiner. Duality and geometry in svm classifiers. In *International Conf. on Machine Learning(ICML)*, pages 57–64, 2000.
3. C. Bhattacharyya. Second order cone programming formulations for feature selection. *Journal of Machine Learning Research(JMLR)*, 5:1417–1433, 2004.
4. M. Bilgic, G. M. Namata, and L. Getoor. Combining collective classification and link prediction. In *Workshop on Mining Graphs and Complex Structures at the IEEE International Conference on Data Mining(ICDM)*, 2007.
5. G. Calafiore and M. Campi. Uncertain convex programs: Randomized solutions and confidence levels. *Mathematical Programming*, 102:25–46, 2005.
6. D. P. D. Farias and B. V. Roy. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29(3):462–478, 2001.
7. L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of relational structure. In *International Conference on Machine Learning(ICML)*, 2001.
8. L. Getoor, N. Friedman, D. Koller, and B. Taskar. Learning probabilistic models of link structure. *Journal of Machine Learning Research*, 3:679– –707, 2002.
9. M. Hasan, V. Chaoji, S. Salem, and M. Zaki. Link prediction using supervised learning. In *SDM workshop on Link Analysis Counter-terrorism and Security*, 2006.
10. S. N. Jagarlapudi. *Learning Algorithms using Chance-Constrained Programming.* Ph.d dissertation, Computer Science and Automation, IISc Bangalore, 2007.
11. H. Kashima and N. Abe. A parameterized probabilistic model of network evolution for supervised link prediction. In *International Conference on Data Mining(ICDM)*, 2006.
12. G. Lanckriet, L. E. Ghaoui, C. Bhattacharya, and M. Jordan. Minimax probability machine. In *Annual Conference on Neural Information Processing Systems(NIPS)*, 2001.
13. Y. Li, H. Zaragoza, R. Herbrich, J. Shawe-Taylor, and J. S. Kandola. The perceptron algorithm with uneven margins. In *International Conference on Machine Learning(ICML)*, pages 379–386, 2002.
14. D. Libennowell and J. Kleinberg. The link prediction problem for social networks. In *International Conference on Knowledge Management(CIKM)*, 2003.
15. M. S. Lobo, L. Vandenberghe, S. Boyd, and H. Lebret. Applications of second-order cone programming. *Linear Algebra and its Applications*, 238:193–228, 1998.
16. A. Marshall and I. Olkin. Multivariate chebyshev inequalities. *Annals of Mathematical Statistics*, 31:1001–1014, 1960.

17. J. S. Nath and C. Bhattacharyya. Maximum margin classifiers with specified false positive and false negative error rates. In *SIAM International Conference on Data Mining(SDM)*, 2007.
18. J. S. Nath, C. Bhattacharyya, and M. N. Murty. Clustering based large margin classification: a scalable approach using socp formulation. In *International Conference on Knowledge Discovery and Data Mining(KDD)*, 2006.
19. M. Newman. Clustering and preferential attachment in growing networks. *Physical Review Letters*, 64, 2001.
20. A. Popescul, R. Popescul, and L. Ungar. Statistical relational learning for link prediction. In *IJCAI workshop on Learning Statistical Models for Relational Data*, 2003.
21. M. Rattngon and D. Jensen. The case for anomalous link discovery. *SIGKDD Explorations*, 7(2):41–47, 2005.
22. P. Sarkar, D. Chakrabarti, and A. Moore. Theoretical justification of popular link prediction heuristics. In *International Conference on Learning Theory(COLT)*, pages 295–307, 2010.
23. P. K. Shivaswamy, C. Bhattacharyya, and A. J. Smola. Second order cone programming approaches for handling missing and uncertain data. *Journal of Machine Learning Research (JMLR)*, 7:1283–1314, 2006.
24. B. Taskar, M. fai Wong, P. Abbeel, and D. Koller. Link prediction in relational data. In *Annual Conference on Neural Information Processing Systems(NIPS)*, 2003.
25. C. Wang, V. Satuluri, and S. Parthasarathy. Local probabilistic models for link prediction. In *International Conference on Data Mining(ICDM)*, 2007.
26. M. Wick, K. Rohanimanesh, A. Culotta, and A. McCallum. Samplerank: Learning preferences from atomic gradients. In *Neural Information Processing Systems (NIPS) Workshop on Advances in Ranking*, 2009.
27. E. Zheleva, L. Getoor, J. Golbeck, and U. Kuter. Using friendship ties and family circles for link prediction. In *2nd ACM SIGKDD Workshop on Social Network Mining and Analysis(SNA-KDD)*, 2008.

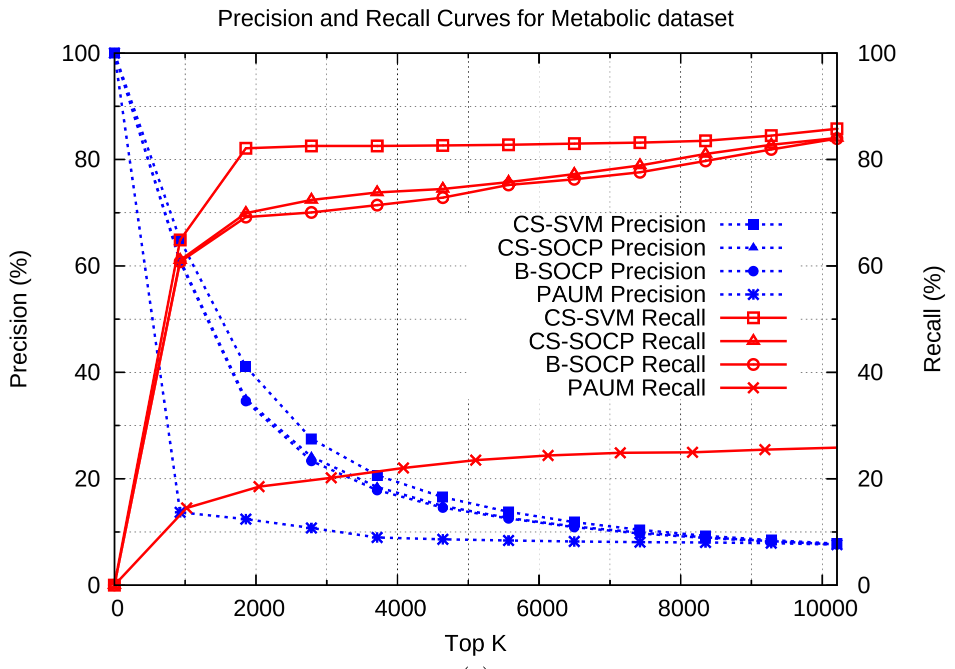


(a)

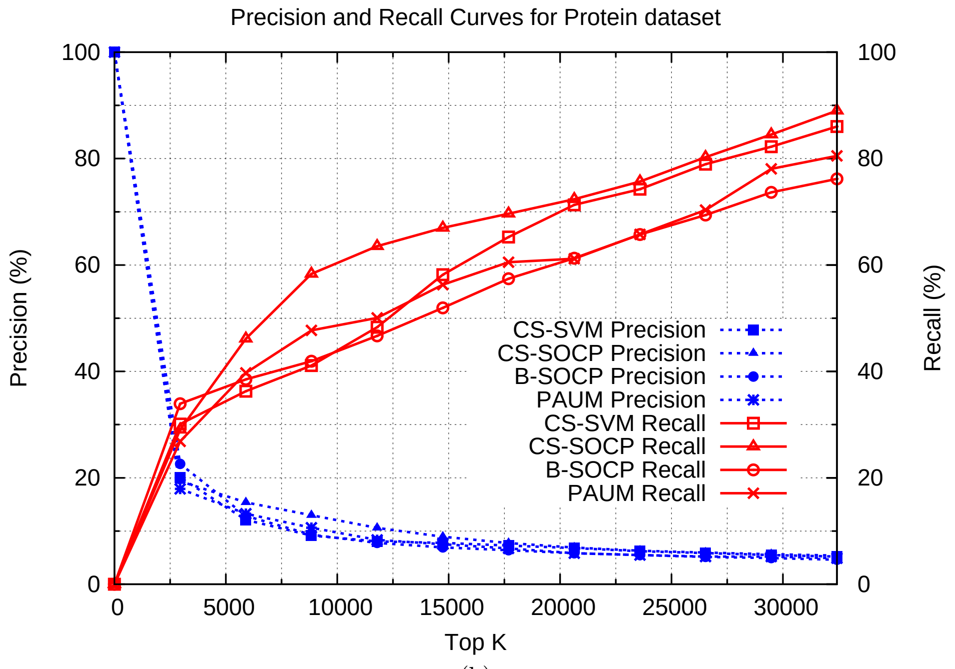


(b)

Fig. 2: Precision and Recall curves for Co-authorship networks (a) DBLP (b) Genetics

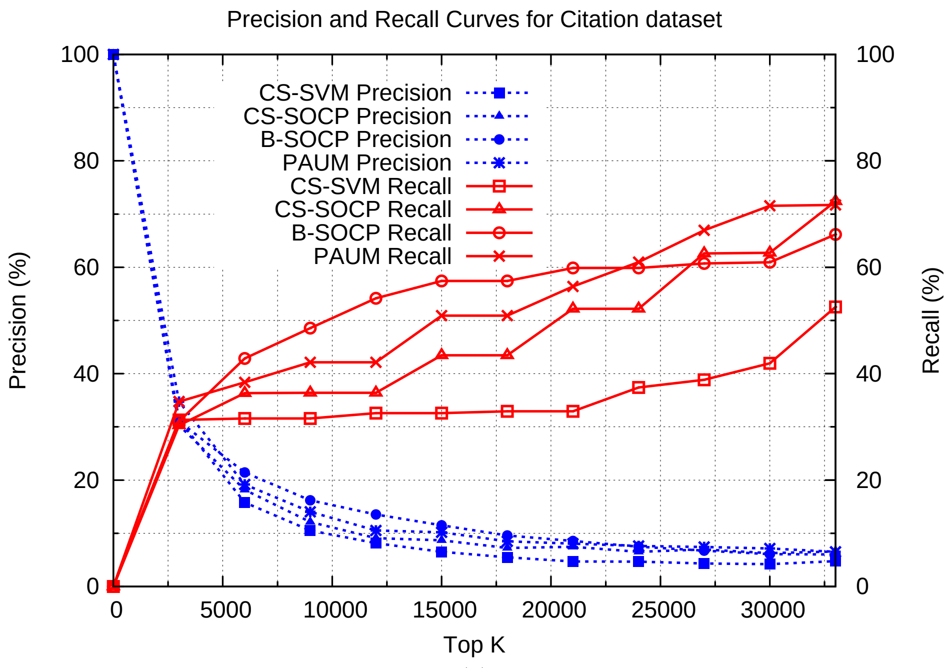


(a)

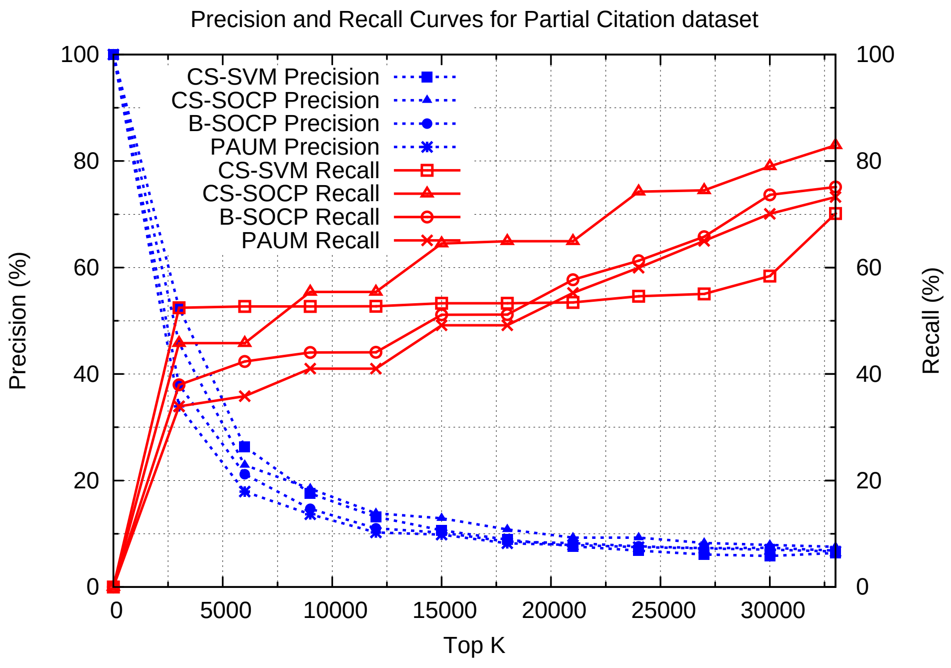


(b)

Fig. 3: Precision and Recall curves for Biological networks (a) Metabolic netowork (b) Protein-protein interaction network



(a)



(b)

Fig. 4: Precision and Recall curves for citation networks (a) Complete bibliography prediction task (b) Partial bibliography prediction task