Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)

Handling Class Imbalance in Link Prediction Using Learning to Rank Techniques

Bopeng Li, Sougata Chaudhuri, and **Ambuj Tewari**
Department of Statistics, University of Michigan
1085 South University Ave.
Ann Arbor, Michigan 48109-1107
{bopengli, sougata, tewaria}@umich.edu

Abstract

We consider the link prediction (LP) problem in a partially observed network, where the objective is to make predictions in the unobserved portion of the network. Many existing methods reduce LP to binary classification. However, the dominance of absent links in real world networks makes misclassification error a poor performance metric. Instead, researchers have argued for using ranking performance measures, like AUC, AP and NDCG, for evaluation. We recast the LP problem as a learning to rank problem and use effective learning to rank techniques directly during training which allows us to deal with the class imbalance problem systematically. As a demonstration of our general approach, we develop an LP method by optimizing the cross-entropy surrogate, originally used in the popular ListNet ranking algorithm. We conduct extensive experiments on publicly available co-authorship, citation and metabolic networks to demonstrate the merits of our method.

Introduction

Since the publication of Liben-Nowell and Kleinberg's seminal paper (2003), the link prediction (LP) problem has received much attention. Many existing techniques treat LP as a binary classification problem and apply classification methods designed for minimizing misclassification error (see references within (Menon and Elkan 2011)). The number of present links is orders of magnitude smaller than the number of absent links in most real world networks. For LP treated as classification problem, this creates *highly imbalanced* classes. General classification techniques will learn a poor classifier because a low error rate can be achieved by trivially predicting everything to be negative.

In order to overcome the imbalance problem, researchers have argued for using ranking measures like AUC (Menon and Elkan 2011), precision-recall curves (Doppa et al. 2010), and NDCG (Yang et al. 2011), during *evaluation*. Still, most existing methods minimize misclassification error during *training*, but coupled with certain class balancing techniques, such as undersampling (Chawla, Japkowicz, and Kotcz 2004). However, there are problems with undersampling and (Menon and Elkan 2011) proposed to learn a classification function by directly optimizing a surrogate of ranking measure AUC during training. This leads to the possibility that other, more effective ranking techniques can be used during training to systematically handle the class imbalance problem.

Learning to rank is a well studied problem in information retrieval (Liu 2009). In learning to rank, an instance consists of a query and an associated list of documents. The supervision is in terms of a relevance vector, which indicates how relevant each document is to the query. During training, a set of queries along with their associated document lists and relevance vectors is used to learn a ranking function. For a test query, the learnt ranking function ranks the documents and a ranking measure evaluates the ranked list against the true relevance vector. There are many well established learning to rank techniques in the literature. To the best of our knowledge, the only other work that uses a learning to rank technique during training in LP is (Yazdani, Collobert, and Popescu-Belis 2013). However, the paper does not explicitly formulate LP as a learning to rank problem and uses a specific pairwise ranking surrogate that suits only their method.

Our main contribution is to give a formal procedure for recasting LP as a learning to rank problem. We define queries, documents and relevance vectors on a network. We then develop an LP method based on the cross-entropy surrogate, originally used in the popular ListNet ranking algorithm (Cao et al. 2007). We conduct extensive experiments on co-authorship, citation and metabolic networks and show the benefit of our method over other baselines.

Link Prediction as Learning to Rank Problem

We first give a formal definition of the learning to rank problem. In learning to rank, each query-document pair is jointly represented by a feature vector. Thus, a list of m documents pertaining to a query q is represented as a matrix $X = (x_1, \ldots, x_m)^T \in \mathbb{R}^{m \times d}$, where $x_i \in \mathbb{R}^d$ represents the joint feature vector of query q and the i^{th} document. The relevance vector for the document list is represented as $R = (r_1, \ldots, r_m)^T$, where $r_i \in \{0, 1\}$ denotes relevance of i^{th} document to query q (where 1 means relevant, 0 means irrelevant). For learning, a training set consisting of n queries, along with their list of documents and relevance vectors, is provided, denoted as $\{(X^1, R^1), \ldots, (X^n, R^n)\}$.

A common technique used to learn a ranking function is

to learn a scoring function f, from which we can get the ranked list by sorting the scores. For a document list X, f gives a score vector $s = f(X) \in \mathbb{R}^m$, where s_i is score of the i^{th} document. Any ranking measure can be used to judge the quality of a ranking function, by comparing the output ranked list to the ground truth relevance vector.

In the LP problem, a partially observed network is represented as $G = (V, E)$, where V is the set of nodes and E is the set of links. Let there be N nodes with each node having a d dimensional feature vector (node information). We make the natural assumption that the presence of a link between 2 nodes is a function of their joint feature vector. This assumption is made for any feature based LP model. The training data is constructed as follows. Each node in V, in turn, acts as a query node while all other nodes work as documents. For query node q, the joint feature vector of q and document node i is $\text{vec}(x_q x_i^\top) \in \mathbb{R}^{d^2}$, where $\text{vec}(\cdot)$ is the vectorization operator on matrices. The feature matrix for the list of documents associated with node q is represented as $X^q \in \mathbb{R}^{(N-1)\times d^2}$, where the i^{th} row is the joint feature vector for node i and q. The relevance vector is obtained from the network topology, i.e., $R^q \in \{0,1\}^{N-1}$, where, the ith entry of R^q is 1 if and only if there is a link present between nodes q and i. Thus, the training data consists of N (matrix, vector) pairs, i.e. $\{(X^1, R^1), \ldots, (X^N, R^N)\}$, where (X^q, R^q) corresponds to node q acting as a query.

Once the training data is prepared, a scoring function f will be learnt from it. We use a linear scoring function parametrized by a matrix M, and the score vector produced for instance X^q is $f_M(X^q) = s^q \in \mathbb{R}^{N-1}$, where $s_i^q = \langle \text{vec}(M), \text{vec}(x_q x_i^\top) \rangle$ ($\langle \cdot, \cdot \rangle$ is inner product). Note that this is same as the bilinear model $s_i^q = x_q^\top M x_i$. In the general framework, the parameter matrix M is learnt via minimization of (regularized) empirical version of some ranking surrogate $\phi : \mathbb{R}^{N-1} \times \{0,1\}^{N-1} \mapsto \mathbb{R}$, as follows:

$$\min_M L(M) = \lambda\, \Omega(M) + \sum_{q \in V} \phi(s^q(M), R^q). \qquad (1)$$

Here, $s^q(M)$ shows explicit dependence of the score vector on M, Ω is a regularization function to prevent overfitting, and λ is a tuning parameter. As an instantiation of the general framework, we give mathematical definitions of the cross-entropy loss, which was used as a ranking surrogate in ListNet (Cao et al. 2007). For score vector $s \in \mathbb{R}^{N-1}$ induced from document matrix $X \in \mathbb{R}^{(N-1)\times d^2}$ and relevance vector $R \in \{0,1\}^{N-1}$, the cross-entropy loss is defined as

$$\phi_{LN}(s, R) = -\sum_{i=1}^{N-1} P_R(i) \log(P_s(i)), \qquad (2)$$

where $P_R(i) = \frac{e^{R_i}}{\sum_{k=1}^{N-1} e^{R_k}}$ and $P_s(i) = \frac{e^{s_i}}{\sum_{k=1}^{N-1} e^{s_k}}$. The gradient $\nabla_M \phi_{LN}(s(M), R)$ can be easily calculated and a gradient descent algorithm can be used for optimization.

We conduct experiments on 5 popular, publicly available network datasets, including 3 citation networks (*CiteSeer* [N=3312], *Cora* [N=2708], *WebKB* [N=877]), 1 coauthorship network (*NIPS* [N=2037]) and 1 metabolic network (*Metabolic* [N=668]). In the experiments, our method is compared against 3 baseline methods: *RankSVM* (Yazdani, Collobert, and Popescu-Belis 2013), *surrogate optimization of AUC* (Menon and Elkan 2011) and *undersampled 0-1 error* (Chawla, Japkowicz, and Kotcz 2004). Due to page limit, here we only show the results of using NDCG@10 and AP as evaluation measures in Table 1 and 2. For both measures, our method (named "Entropy") consistently outperforms baselines in all the datasets. We also observe that *undersampled 0-1 error* performs almost as good as *RankSVM* and *surrogate optimization of AUC*. Further details on dataset descriptions, baselines, experiments setup, and more empirical results can be found in the full version of the paper at http://arxiv.org/abs/1511.04383.

Table 1: NDCG@10

Dataset	Entropy	RankSVM	AUC-logistic	Undersampled 0-1
CiteSeer	**0.169**	0.161	0.161	0.157
Cora	**0.071**	0.063	0.063	0.061
WebKB	**0.174**	0.127	0.127	0.122
NIPS	**0.257**	0.220	0.220	0.222
Metabolic	**0.067**	0.059	0.058	0.057

Table 2: Average Precision

Dataset	Entropy	RankSVM	AUC-logistic	Undersampled 0-1
CiteSeer	**0.114**	0.105	0.105	0.100
Cora	**0.045**	0.040	0.040	0.040
WebKB	**0.177**	0.140	0.140	0.140
NIPS	**0.192**	0.167	0.167	0.168
Metabolic	**0.067**	0.066	0.066	0.066

Acknowledgements

We acknowledge support of NSF under grant IIS-1319810.

References

Cao, Z.; Qin, T.; Liu, T.-Y.; Tsai, M.-F.; and Li, H. 2007. Learning to rank: from pairwise approach to listwise approach. In *Proceedings of the 24th ICML*, 129–136. ACM.

Chawla, N. V.; Japkowicz, N.; and Kotcz, A. 2004. Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter* 6(1):1–6.

Doppa, J. R.; Yu, J.; Tadepalli, P.; and Getoor, L. 2010. Learning algorithms for link prediction based on chance constraints. In *Proceedings of ECML PKDD 2010: Part I*, 344–360. Springer-Verlag.

Liben-Nowell, D., and Kleinberg, J. 2003. The link prediction problem for social networks. In *Proceedings of CIKM 2003*, 556–559.

Liu, T.-Y. 2009. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval* 3(3):225–331.

Menon, A. K., and Elkan, C. 2011. Link prediction via matrix factorization. In *Proceedings of ECML PKDD 2011: Part II*, 437–452. Springer-Verlag.

Yang, S.-H.; Long, B.; Smola, A.; Sadagopan, N.; Zheng, Z.; and Zha, H. 2011. Like like alike: joint friendship and interest propagation in social networks. In *Proceedings of the 20th International Conference on WWW*, 537–546. ACM.

Yazdani, M.; Collobert, R.; and Popescu-Belis, A. 2013. Learning to rank on network data. In *Mining and Learning with Graphs*, EPFL-CONF-192709.