Validation of Network Classifiers

James Li, Abdullah Sonmez, Zehra Cataltepe, and Eric Bax

Cornell University, Istanbul Technical University, and Yahoo! Inc.
jyl73@cornell.edu, abdullah.sonmez@gmail.com,
cataltepe@itu.edu.tr, baxhome@yahoo.com

Abstract. This paper develops PAC (probably approximately correct) error bounds for network classifiers in the transductive setting, where the network node inputs and links are all known, the training nodes class labels are known, and the goal is to classify a working set of nodes that have unknown class labels. The bounds are valid for any model of network generation. They require working nodes to be selected independently, but not uniformly at random. For example, they allow different regions of the network to have different densities of unlabeled nodes.

Keywords: network classifier, collective classification, validation, error bound, worst likely assignment.

1 Introduction

Networks play fundamental roles in our lives. A network of interactions among genes determines how our bodies grow. Neural networks enable us to think and learn. We participate in social networks. The ecosystem we live in is a complex web of interactions between all living things and our shared environment. Electrical grids supply power; transportation systems bring us goods, and the internet supports information sharing around the world.

Data analysis based on networks has a deep history in social science and telecommunications. Networks are emerging as a basis for data analysis in many other fields, including biology, economics, and engineering [10]. Network data analysis builds on well-established foundations in graph theory [4], including study of small-world [17,16] and other random graphs [3], and statistics [7], including analysis of Markov chains.

Classification of network data [12,9,18], sometimes called collective classification [14], is an emerging sub-field of machine learning. Collective classification techniques have been developed for these transductive network classification problems. These techniques include Loopy Belief Propagation or the Iterative Classification Algorithm [13], which assign an initial set of labels to working nodes, then iteratively apply a model to re-label working nodes based on their network neighbors' input data and labels. Other collective classification techniques, such as Weighted-Vote Relational Classifiers [11], can also be applied in the transductive setting. For a study comparing some collective classification techniques, refer to [14].

In machine learning, data usually consists of examples, each containing input data and output data. In classification problems, the output is a label that

G.L. Gimel' farb et al. (Eds.): SSPR & SPR 2012, LNCS 7626, pp. 448–457, 2012.

takes one of a finite set of values. To illustrate, consider a medical application. Each example corresponds to a person. The input data include the person's age, gender, and levels of some proteins measured in their blood. The output label indicates whether the person has a certain type of infectious disease.

In classification, there is a set of in-sample training examples with known labels and a set of out-of-sample test examples with unknown labels. The training examples are used to develop a classifier, which is used to label the test examples. (The classifier is a function that maps from example inputs to labels.) The primary goal of classification is accuracy – developing a classifier that has a low error rate on the test examples. A secondary goal is to validate classifier accuracy – producing probabilistic error bounds for the classifier. Since accuracy is the primary goal, we prefer to use all available examples for training, introducing the challenge that we lack independent data for validation.

This paper addresses classification problems with network data, where the inputs include links. In our example, a link between two people could indicate they share a water source. In traditional machine learning, examples are assumed to be drawn i.i.d. from an underlying joint input-output distribution. With network data, this is often not the case. For example, medical researchers may have invited an initial "seed" set of people to join the study and be represented by network nodes, and then participants may have invited people they know to join the study, growing the network by non-uniform sampling.

In the transductive setting for machine learning [15,6], the test example inputs are available for training; only the test labels are unknown. In the transductive setting, the test examples are called working examples. This paper focuses on collective classifier validation in the transductive setting, where all node input data and links are known, the training node labels are known, and only the working node labels are unknown. The classifier may take advantage of training node inputs and labels, all links, and working node inputs.

This paper develops worst likely assignment error bounds [2] for network classifiers. These error bounds validate a classifier by computing how much an assignment of labels to the working nodes may disagree with the classifier's labels and still be likely to be the actual unknown labels, according to some statistical criteria. The statistics are based on the fact that training and working nodes are interchangeable a priori, so it is unlikely that the unknown labels for the working examples have very different statistics than the known labels for the training examples.

This paper is organized as follows. Section 2 develops an error bound algorithm and some variations. Section 3 presents experimental results showing bound effectiveness. Then Section 4 concludes with ideas for future work.

2 Algorithms

This section presents algorithms to compute error bounds for network classifiers based on worst likely assignments. Subsection 2.1 develops a basic algorithm that is effective for problems with small working sets but requires too much

computation for larger problems. Subsection 2.2 develops an algorithm for larger problems, using the basic algorithm as a component. Subsection 2.3 shows how to modify the algorithm to control the relationship between component and whole-network bound failure probabilities. Subsection 2.4 shows how to adapt the algorithms for network labeling processes where different nodes have different probabilities of being selected for the working set.

2.1 Basic Algorithm

Let n be the number of nodes in the network. Let $(X_1, Y_1), \ldots, (X_n, Y_n)$ be the input-label pairs for the nodes. The inputs may be from any domain. The labels are drawn from a finite set.

Assume all network nodes and links are known, and all node inputs X_i are known. Assume some node labels Y_i are known and others are unknown. Refer to nodes with known labels as training nodes and nodes with unknown labels as working nodes. (Working nodes are sometimes called test nodes.) Define T to be the set of training nodes and W to be the set of working nodes. Let $t = |T|$ and $w = |W|$.

For now, suppose that the working set was selected by applying i.i.d. Bernoulli trials to the nodes, with success indicating the node is a working node and failure indicating the node is a training node. Then each training-working split with t training and w working nodes is equally likely. (The constraint that all nodes are equally likely to be working nodes will be removed later, in Subsection 2.4.)

The algorithm uses a concept of *continuous rank* to assess likelihood. Define the continuous rank $r(s_0, \{s_0, \ldots, s_{m-1}\})$ of a value s_0 among values $s_0, \ldots, s_{m-1}$ to be the sum of

- The number of values in $s_1, \ldots, s_{m-1}$ less than s_0,
- a value drawn uniformly at random from the integers zero to the number of values in $s_1, \ldots, s_{m-1}$ equal to s_0, and
- a value drawn uniformly at random from the real numbers in $[0, 1]$.

Note that if $s_0, \ldots, s_{m-1}$ are drawn i.i.d., then $r(s_0, \{s_0, \ldots, s_{m-1}\})$ is uniformly distributed over the real numbers in $[0, m]$.

The basic algorithm is:

1. Select bound failure probability $\delta > 0$. Select sample size $m > 0$. Choose $m - 1$ comparison training-working splits $(T_1, W_1), \ldots, (T_{m-1}, W_{m-1})$ uniformly at random from the $\binom{t+w}{t}$ splits that have t training and w working nodes. Either choose them without replacement and prohibit the actual training-working split (T, W), or choose them with replacement. Let $(T_0, W_0) = (T, W)$.
2. Use a set M to store error counts for likely assignments. Initially, $M = \emptyset$.
3. Let A be the set of assignments a' that agree with the labels on nodes in T and assign any labels to the nodes in W. $\forall a' \in A$:
   (a) Assign a' to the node labels.

(b) Train $g_0, \ldots, g_{m-1}$, using $T_0, \ldots, T_{m-1}$ as the training nodes and $W_0, \ldots, W_{m-1}$ as the working nodes. (The training procedure for g_i may use all node inputs, all links, and the labels on T_i, but not the labels on W_i.)

(c) For $i \in \{0, \ldots, m-1\}$, let score s_i be the number of errors g_i makes when classifying the nodes in W_i under assignment a':

$$s_i = \sum_{x \in W_i} I(g_i(x) \neq a'(x)),$$

where x is a node, $I()$ is the indicator function – one if the argument is true and zero otherwise, and $a'(x)$ is the label a' assigns to x.

(d) If

$$r(s_0, \{s_0, \ldots, s_{m-1}\}) \leq (1-\delta)m,$$

then insert s_0 into set M, because a' is a likely assignment: $a' \in L$.

4. Return $\max M$ as the error bound, because it is the maximum error count for a likely assignment.

Theorem 1. *With probability at least $1-\delta$, the basic algorithm returns a valid error bound:*

$$\mathbb{P}\{\sum_{x \in W} I(g(x) \neq a(x)) \leq \max M\} \geq 1-\delta,$$

where $\max M$ is the value returned by the basic algorithm, g is the algorithm's g_0 when $a' = a$, and $a(x)$ is the actual (unknown) label for x.

Proof. Let $s_i(a')$ be the value of s_i for assignment a' in the algorithm. Then the theorem states:

$$\mathbb{P}\{s_0(a) \leq \max M\} \geq 1-\delta.$$

At the end of the algorithm,

$$M = \{s_0(a') | r(s_0(a'), \{s_0(a'), \ldots, s_{m-1}(a')\}) \leq (1-\delta)m\}.$$

Since each training-working split with t training and w working nodes is equally likely to be each of $(T_0, W_0), \ldots, (T_{m-1}, W_{m-1})$, the values $s_0(a), \ldots, s_{m-1}(a)$ are i.i.d. So

$$r(s_0(a), \{s_0(a), \ldots, s_{m-1}(a)\}) \sim U(0, m).$$

Hence

$$\mathbb{P}\{r(s_0(a), \{s_0(a), \ldots, s_{m-1}(a)\}) \leq (1-\delta)m\} = 1-\delta.$$

So

$$\mathbb{P}\{s_0(a) \in M\} \geq 1-\delta,$$

which implies

$$\mathbb{P}\{s_0(a) \leq \max M\} \geq 1-\delta.$$

The basic algorithm requires $\mathrm{O}(w^c m)$ classifier trainings, where c is the number of different class labels, and $\mathrm{O}(w^c m w)$ node classifications. So the basic algorithm requires too much computation to be feasible for large working sets.

2.2 Partition the Working Set to Speed Computation

Since the basic algorithm has running time exponential in the size of the working set, one way to reduce computation is to partition the working set, apply the basic algorithm to each partition, and combine error bounds over partitions to produce an error bound over the whole working set. When applying the basic algorithm to a subset of the working nodes, the remaining working nodes are also unlabeled. Modify the basic algorithm to accommodate these *reserved nodes* as follows:

- Withhold the reserved nodes from all training-working splits, so all comparison splits have the same reserved nodes.
- Leave the reserved nodes unlabeled in assignments a'.
- Allow the training procedure to use the reserved nodes, but not their labels, which are unknown.

The partitioning algorithm is:

1. Select bound failure probability $\delta > 0$.
2. Partition the working set W into subsets $W^1, \ldots, W^p$.
3. For each partition W^i, apply the basic algorithm, using $\delta_p = \frac{\delta}{p}$ as the bound failure probability and $W - W^i$ as the reserved nodes. Let b_i be the bound returned.
4. Return $b_1 + \ldots + b_p$ as an error bound.

Theorem 2. *With probability at least $1 - \delta$, the partitioning algorithm returns a valid error bound:*

$$\mathbb{P}\{\sum_{i=1}^{p} \sum_{x \in W^i} I(g^i(x) \neq a(x)) \leq b_1 + \ldots + b_p\} \geq 1 - \delta,$$

where g^i is the classifier trained as g_0 in the modified basic algorithm when $a' = a$, with working set W^i and reserved nodes $W - W^i$.

Note that for most (non-random) training procedures, $g^1, \ldots, g^p$ are the same classifier, because each g_0 classifier training in the modified basic algorithm ignores all labels in W.

Proof. By Theorem 1,

$$\forall i \in \{1, \ldots, p\} : \mathbb{P}\{\sum_{x \in W^i} I(g^i(x) \neq a(x)) \leq b_i\} \geq 1 - \frac{\delta}{p}.$$

So

$$\forall i \in \{1, \ldots, p\} : \mathbb{P}\{\sum_{x \in W^i} I(g^i(x) \neq a(x)) > b_i\} \leq \frac{\delta}{p}.$$

Using sum bounds on the probability of a union:

$$\mathbb{P}\{\exists i \in \{1, \ldots, p\} : \sum_{x \in W^i} I(g^i(x) \neq a(x)) > b_i\} \leq p\frac{\delta}{p} = \delta.$$

So

$$\mathbb{P}\{\forall i \in \{1,\ldots,p\} : \sum_{x \in W^i} I(g^i(x) \neq a(x)) \leq b_i\} \geq 1 - \delta,$$

and this implies

$$\mathbb{P}\{\sum_{i=1}^{p} \sum_{x \in W^i} I(g^i(x) \neq a(x)) \leq b_1 + \ldots + b_p\} \geq 1 - \delta.$$

The partitioning algorithm requires $O((|W^1|^c + \ldots + |W^p|^c)m)$ classifier trainings and $O((|W^1|^{c+1} + \ldots + |W^p|^{c+1})m)$ node classifications. So as the number of partitions increases, the computation required decreases. The tradeoff is weaker bounds, because δ is divided by the number of partitions to produce δ_p, the bound failure probability in the basic algorithm for each partition.

2.3 Use Nearly Uniform Validation to Strengthen Bounds

One way to prevent slicing δ too thinly over partitions is to employ *nearly uniform* error bounds [1]. Instead of insisting that all partition bounds b_i hold to ensure that the sum bound $b_1 + \ldots + b_p$ holds, nearly uniform bounds allow some partition bound failures. For example, suppose one partition bound failure is allowed. Then the error bound $b_1 + \ldots + b_p$ becomes

$$b_1 + \ldots + b_p + \max_i(|W^i| - b_i),$$

since when a partition bound fails, all nodes in the partition may be misclassified. Since this bound accounts for a single failure, it is invalid only if there are two or more partition bound failures. The distribution that maximizes the probability of two or more failures has each single bound failure accompanied by exactly one more. So the worst-case probability of two or more failures is at most $\frac{p}{2}\delta_p$, where δ_p bounds each single-bound failure probability. (Recall that the worst-case probability of one or more failures is $p\delta_p$.) So

$$\mathbb{P}\{\sum_{i=1}^{p} \sum_{x \in W^i} I(g^i(x) \neq a(x)) \leq b_1 + \ldots + b_p + \max_i(|W^i| - b_i)\} \geq 1 - \frac{p}{2}\delta_p,$$

making it possible to set partition bound failure probabilities $\delta_p = 2\frac{\delta}{p}$ and still achieve a valid overall bound with probability at least $1 - \delta$.

Similarly, allowing k partition bound failures produces an error bound

$$b_1 + \ldots + b_p + max_{i_1 \neq \ldots \neq i_k}(\sum_{j=1}^{k} |W^{i_j}| - b_{i_j}).$$

Since this bound is valid with probability at least

$$1 - \frac{p}{k+1}\delta_p,$$

setting $\delta_p = (k+1)\frac{\delta}{p}$ for each partition achieves a valid bound with probability at least $1 - \delta$.

2.4 Cohorts and Non-identical Selection for the Working Set

The error bounds and algorithms in the previous subsections require that each node is equally likely to be unlabeled. Consider the case when there are subsets of nodes that are equally likely to be unlabeled, but the probability varies between subsets. Refer to each subset as a *cohort*. Within each cohort, nodes may be selected for the working set by i.i.d. Bernoulli trials or by uniform selection over subsets of a fixed size.

To illustrate, recall the example problem from Section 1. Nodes represent people, and each label indicates whether a person has an infectious disease. Suppose the study began with 100 volunteers, who all received testing for the disease. Then suppose those volunteers recruited another 250 volunteers, of whom 100 were selected at random for testing. In this case, the initial 100 volunteers are one cohort, and the next 250 are another.

To apply the algorithms to cohorts, limit the selection of comparison training-working splits $(T_1, W_1), \ldots, (T_{m-1}, W_{m-1})$ to splits that have the same number of nodes from each cohort as T in each T_i and as W in each W_i. Select the comparison splits uniformly among such splits. Then each such split is equally likely a priori to be (T, W). So the error bounds are valid.

Now consider the general case where nodes are selected for the working set based on independent Bernoulli trials with different probabilities. In the algorithms, select working sets for the comparison training-working splits by independent Bernoulli trials over $T \cup W$, using each node's a priori probability to determine whether it is in the working set. Since the working sets in different comparison splits may have different sizes, consider ranking by error rate (error count divided by size of working set) rather than error count to determine which assignments are likely. Whether ranking by error rate or error count, when $a' = a$, the scores for (T, W) and the comparison splits are i.i.d. over selection of (T, W) and the comparison splits. So the error bounds are valid.

3 Experiments

This section describes experiments using the bounds developed in this paper. The experiments are based on a network generated at random using the procedure outlined in Section 5.1 of [5] (with their $m_s = 32$). The network has 1000 nodes and 7882 edges. Each node has one of two labels.

For each of 100 trials, 400 nodes are selected to be the training set and another 16 nodes are selected to be the working set. The selections are uniformly at random and without replacement. The classifier assigns each working node the label of the majority of its neighboring training nodes, with ties broken at random. The test error rate is the fraction of working set examples misclassified.

For each trial, error bounds are computed for $\delta \in \{0.1, 0.2, 0.3\}$, for numbers of comparison working-training sets $m \in \{100, 200\}$, and with and without partitioning. With partitioning, the working set is divided in half, making 8 examples in each partition, and the bounds are computed using the algorithm in Subsection 2.2. Without partitioning, the algorithm in Subsection 2.1 is used.

Over the 100 trials, the test error rate averages 0.216, with standard deviation 0.098. (We state all results to three decimals.) Table 1 shows the averages and standard deviations of differences between bounds and test error rates for the various bound computation methods and values of δ. The table shows sample standard deviations; standard deviations of the estimates of the means are one tenth of these values, indicating that the differences among means are statistically significant.

The unpartitioned bounds are stronger than their partitioned counterparts – the decreased computation bears a price in bound strength.This is partly because δ is halved for each partition, but also because having smaller working sets in the partitions produces weaker bounds. To see this, observe that the partitioned bounds with $\delta = 0.2$ are weaker than the unpartitioned bounds with $\delta = 0.1$. Increasing the sample size m increases bound strength for all but one bound, but the effect is not as large as for partitioning.

Table 1. Bound - Test Error Rate

	unpartitioned		partitioned	
m	δ	difference	δ	difference
100	0.1	0.170 ± 0.094	0.1	0.284 ± 0.097
	0.2	0.113 ± 0.094	0.2	0.212 ± 0.099
	0.3	0.075 ± 0.099	0.3	0.171 ± 0.098
200	0.1	0.166 ± 0.095	0.1	0.286 ± 0.102
	0.2	0.108 ± 0.096	0.2	0.206 ± 0.106
	0.3	0.069 ± 0.101	0.3	0.168 ± 0.096

4 Discussion

This paper presents a method to validate network classifiers, computing PAC error bounds for a classifier over working examples in a transductive setting. The method is based on the idea that training and working sets of nodes are interchangeable a priori, so the likelihood of substantial differences between the two sets is small. The method does not depend on which underlying process generated the network. The only requirement is that whether each node is labeled is determined independently from the labeling determination of other nodes. The bounds do not require that labeling has the same probability for each node – using cohorts produces error bounds when nodes in some sets are more likely than others to be in the working set a priori.

One direction for future research is to remove the constraint that nodes are selected independently for the working set. In practice, this constraint may not hold. For example, in a social network, if one person provides the data to label her node, that may encourage her friends to do the same. For some types of correlation, it may be possible to alter the selection of comparison training-working splits, as we do to accommodate cohorts. If a sampling technique, such as snowball sampling [8], is known to mimic how the working set is selected, then perhaps the technique can be used to generate comparably likely training-working splits.

Another challenge for future research is to develop faster algorithms. There is an efficient algorithm to compute error bounds for 1-nearest neighbor classifiers, based on dynamic programming [2]. The method depends on avoiding long cycles of dependency, where the label for one example influences the classification of another, which influences the classification of another, and so on, returning to influence the first example. It may be possible to develop a similar error bound algorithm for network classifiers by partitioning the examples in the working set into subsets in which the examples do not influence one another's classifications. Then the error bounds over subsets could be combined to produce an error bound over the whole working set, as in this paper.

The error bounds developed in this paper apply to the transductive setting, where the working nodes and their links are known. In the future, it would be interesting to extend these bounds to cases where the working nodes are unknown. If the working nodes have a known distribution, that information could be used to sample working nodes to develop samples of bounds. Probabilistic error bounds can be based on the statistics over these samples. Even without a known distribution, the training nodes may be used to estimate a distribution for working nodes, if they are known to be drawn from the same distribution. In some cases, with growing networks, only the most recent training nodes may be drawn from a similar distribution to the next nodes to be drawn, which are the nodes of interest for error bounds.

References

1. Bax, E.: Nearly uniform validation improves compression-based error bounds. Journal of Machine Learning Research 9, 1741–1755 (2008)
2. Bax, E., Callejas, A.: An error bound based on a worst likely assignment. Journal of Machine Learning Research 9, 581–613 (2008)
3. Bollobas, B.: Random Graphs, 2nd edn. Cambridge University Press (2001)
4. Bondy, J.A., Murty, U.: Graph Theory. Springer (2008)
5. Cataltepe, Z., Sonmez, A., Baglioglu, K., Erzan, A.: Collective classification using heterogeneous classifiers. In: 7th International Conference on Machine Learning and Data Mining, MLDM 2011 (2011)
6. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods. Cambridge University Press (2000)
7. Feller, W.: An Introduction to Probability Theory and Its Applications. John Wiley & Sons, New York (1968)
8. Frank, O.: Survey sampling in graphs. Journal of Statistical Planning and Inference 1, 235–264 (1977)
9. Getoor, L., Friedman, N., Koller, D., Taskar, B.: Learning probabilistic models of link structure. Journal of Machine Learning Research 3, 679–707 (2002)
10. Kolaczyk, E.D.: Statistical Analysis of Network Data. Springer (2010)
11. Macskassy, S., Provost, F.: A simple relational classifier. In: Proceedings of the Multi-Relational Data Mining Workshop (MRDM) at the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, pp. 64–76 (2003)
12. Macskassy, S.A., Provost, F.: Classification in networked data: A toolkit and a univariate case study. Journal of Machine Learning Research 8, 935–983 (2007)

13. Sen, P., Getoor, L.: Empirical comparison of approximate inference algorithms for networked data. In: ICML Workshop on Open Problems in Statistical Relational Learning, SRL 2006 (2006)
14. Sen, P., Namata, G., Bilgic, M., Getoor, L., Gallagher, B., Eliassi-Rad, T.: Collective classification in network data. AI Magazine 29(3) (2008)
15. Vapnik, V.: Statistical Learning Theory. John Wiley & Sons (1998)
16. Watts, D.: Six Degrees: The Science of a Connected Age. Norton & Company (2003)
17. Watts, D., Strogatz, S.: Collective dynamics of 'small-world' networks. Nature 393(6684), 440–442 (1998)
18. Zheleva, E., Getoor, L.: To join or not to join: The illusion of privacy in social networks with mixed public and private user profiles. In: 18th International World Wide Web Conference, pp. 531–531 (April 2009)