An Impact Criterion for Active Graph Search

Xuezhi Wang
Carnegie Mellon University
xuezhiw@cs.cmu.edu

Roman Garnett
Carnegie Mellon University
rgarnett@andrew.cmu.edu

Jeff Schneider
Carnegie Mellon University
schneide@cs.cmu.edu

1 Introduction

It is often cheap to collect a large amount of unlabeled data, but much more expensive to get labels for it. This has led to increasing interest in active learning algorithms that iteratively choose which labels to query with a goal of optimizing some prediction accuracy criterion based on only a subset of the labels. In this paper, we focus instead on the active search problem, where we seek records belonging to a certain positive class. Although we will still build a predictive model and may request labels to improve it, our algorithm will ultimately be scored only on how many positives it finds.

Although active search applications appear with many different types of data, we restrict our attention to graph data where the graph structure is known and we are interested in labels on the nodes that are expensive to collect. A typical model will use the graph link structure and the labels already collected to make predictions about unlabeled nodes. There are many interesting applications include fraud detection in a financial network, and relevant paper discovery in a citation graph.

One might expect active learning algorithms to be appropriate for active search as well since they will produce a good model that can be used to find positives. However, in active search a good algorithm must trade off the need to exploit (use the current model to collect positives) against the need to explore (develop a better model to more accurately guess the positives in future selections). A traditional active learning method would focus entirely on exploring and only collect positives by accident. The exploration/exploitation feature of the problem might lead one to consider bandit algorithms for active searching, which, however, are hard to adapt since we will never choose the same node more than once. As with active learning and bandit problems, the optimal active learning solution usually requires an intractable look ahead search over an exponential number of possible future queries and label outcomes. Algorithms based on evaluating the expected utility over a truncated look ahead have been proposed and good empirical results have been obtained by using a smart pruning strategy that reduces the cost by orders of magnitude and makes longer look aheads possible [1]. In the same work, it was proven that arbitrarily better performance is always possible with one further step of look ahead. In empirical examples, it seems that much better performance is available from looking ahead much further than is possible even with smart pruning.

Many successful algorithms for active learning and bandit problems do a myopic or 1-step evaluation of a well-crafted surrogate objective rather than directly optimizing expected utility. Inspired by their successes, we propose such a method for active search in graphs. We begin by introducing a soft-label model for graphs, which attaches a pseudo node to each original node and holds the observed labels. As the selection criterion, we propose the probability of a positive (the exploitation) plus a measure of impact based on the number of additional positives likely to be identified (the exploration). Both the model and the impact factor can be efficiently computed using incremental updates to the model matrices. We compare our method to uncertainty sampling, a modified UCB algorithm, and a previously proposed model for graphs. On citation and wikipedia graphs our method outperforms all the others.

2 Related Work

There has been much research on achieving good classification with partially labeled data. In [2], the authors propose a Markov random walk based algorithm to classify unlabeled points using the information of labeled ones as well as the graph structure. In [3], the authors propose a semi-supervised label learning method which is based on the Gaussian random field model. In [4], the authors adopt a relational active learning model to improve both model estimation and prediction after acquiring a node's label.

Active graph search involves an exploration and exploitation dilemma, where the Upper Confidence Bound (UCB) algorithm [5][6] is a popular method of addressing this issue. The basic idea of UCB in multiarmed bandit problems is to sum the current estimate about the reward of each arm (exploitation) and the uncertainty about that arm (exploration). UCB is appealing due to its regret bounds but the setting is too confined to be used in the active search problem. UCB intends to repeatedly select good arms while the active graph search problem does not allow repeated selections.

There is a more subtle issue as well. Ideally, the exploration component of an algorithm would optimize some measure of information gained from a label. In a traditional independent-arm bandit problem this is easily replaced by the uncertainty for a particular arm because the information gain is confined to that arm. When a Gaussian process model is used, information is spread throughout the model. However, because of the symmetric and homogeneous properties of typical kernels, the information gain for sampling at a point can again be substituted with the current model uncertainty at that point. Good graph models offer no such easy way out. The potential information to be gained by choosing a hub can be much larger than that of choosing a disconnected singleton even if the latter is much more uncertain. This property motivates the impact factor in our proposed method.

3 Approach

3.1 Problem Description and the Soft-Label Model

Here we formally define a binary graph active search problem. We are given a finite set of n nodes, indexed $\{1, ..., n\}$, the weight matrix $W = [w_{ij}]$, and an unknown set of labels $Y = \{y_1, ..., y_n\}$ where $y_i \in \{0, 1\}$ and we want to identify the nodes for which $y_i = 1$. Initially all nodes belong to the unlabeled set, U. At each iteration we choose a node i, find out y_i, and move node i to the labeled set, L. Our performance after k iterations is the sum of the y_i in L.

We begin by considering models for predicting the unknown values of Y. In [3], the authors propose a harmonic function $f = D^{-1}Wf$ (W is the weight matrix and D is the diagonal matrix with $D_{ii} = \sum_j W_{ij}$), with each entry f_i in the harmonic function as an indicator of the probability that a random walk starting from node i will hit a label 1 before it hits a label 0. However, this model has some problems, especially for active search. Suppose we first discover the hub node i of a star structure with label $y_i = 0$, which is connected to many nodes with label $y_j = 1$ in its immediate neighborhood. Discovering any number of nodes with label $y_j = 1$ in N_i will never increase any remaining element of f_u from N_i since a random walk will always stop at the 0 label of node i.

Following the idea about dongle nodes in [3], we propose a soft-label model, which attaches a pseudo node to each labeled original node i to hold its label. The edge between the pseudo node and the original node adjusts the probability of (1) hitting a labeled node and stopping versus (2) ignoring the label and continuing the random walk. Leaving f as the estimate of the original nodes and letting x_l represent the labeled pseudo nodes (with entry 0 for unlabeled nodes), we get: $f = D_*^{-1} \begin{bmatrix} W & D_l \end{bmatrix} \begin{bmatrix} f & x_l \end{bmatrix}^T$, where W is the weight matrix, and D_l is an $n \times n$ diagonal matrix with $D_{l(ii)} = \frac{\eta}{1-\eta} \sum_j w_{ij}$ for all $i \in L$ and $D_{l(ii)} = 0$ for all $i \in U$, which indicates that there is a transition probability η from a labeled node i to its labeled pseudo node. D_* is also a diagonal matrix with $D_{*(ii)} = \sum_j w_{ij} + D_{l(ii)} = \frac{1}{1-\eta} \sum_j w_{ij}$ for $i \in L$, acting as a row normalizing factor.

It is often useful to include prior information on labels and we can attach a pseudo node to the unlabeled original nodes for this purpose. We set the pseudo node label to be the value of the prior. The weight of the attached edge represents the strength of the prior. We set the weight of node i to be $\omega_0 D_{ii}$, where ω_0 is the strength, and D_{ii} is the degree of node i. By a similar derivation as above and absorbing the row normalizing factor we get: $f = \begin{bmatrix} A & D' \end{bmatrix} \begin{bmatrix} f & x \end{bmatrix}^T$ which leads to $f =$

$(I - A)^{-1}D'x$, where $A_{ij} = \begin{cases} (1-\eta)(D^{-1}W)_{ij} & i \in L \\ \frac{1}{1+\omega_0}(D^{-1}W)_{ij} & i \in U \end{cases}$ and $D'_{ii} = \begin{cases} \eta & i \in L \\ \frac{\omega_0}{1+\omega_0} & i \in U \end{cases}$. Here x is a vector with labels in the entries corresponding to labeled nodes, and a value π for the prior in the entries corresponding to unlabeled ones. f will be a vector with $f_i = P(y_i = 1|L)$ for all the nodes, but we only care about those entries i with $i \in U$.

3.2 The Selection Criterion

We propose the selection criterion with the following form: $score_i^{(t)} = f_i^{(t)} + \alpha \times IM_i^{(t)}$, where $f_i^{(t)}$ indicates the model's prediction for node i after seeing t labels, $IM_i^{(t)}$ is the expected impact on future positives found by choosing node i now, and α is a parameter trading off exploration and exploitation. We choose the node with highest $score_i$ at each iteration.

There are many possibilities for defining IM. The entropy in f_i would be an obvious choice, however that does a poor job of capturing how much effect node i has on the rest of the graph and especially how much it will increase the number of positives we find in the future after observing y_i. We can consider $\Sigma_{i \in U} f_i$ as an indication of the number of positives we will find in the future. Therefore, we propose to explicitly condition on the expected value of y_i and measure its potential to increase values of f in the unlabeled part of the graph. We propose:

$$IM_i^{(t)} = P(y_i^{(t)} = 1|L^{(t)}) \sum_{j \in \{U^{(t)} \setminus i\}} (P(y_j^{(t+1)} = 1|y_i = 1, L^{(t)}) - P(y_j^{(t)} = 1|L^{(t)})) = f_i \sum_{j \in \{U^{(t)} \setminus i\}} (f'_j - f_j)$$

where f is the original prediction for each node and f' is the prediction conditioned on adding node i to the training set with label $y_i = 1$. Note that we do not condition on seeing $y_i = 0$. Intuitively you might want to set up the impact criterion to marginalize over the unknown outcome. However, doing that would correspond to estimating the change in expected number of positives in this neighborhood under the assumption that your policy will continue choosing nodes in this neighborhood even if it sees a negative outcome. This is not the policy we will follow. If a negative is observed, the policy will move to some other part of the graph. By doing it the way we propose we are representing both the unknown outcome and the decision that will follow (i.e. to continue choosing nodes in this neighborhood or not).

This impact factor is heuristic and computing the true future expected increase in positives chosen is just as computationally intractable as implementing the full optimal policy. However, this definition of IM is able to tractably imitate a full look ahead by computing the full impact over all the nodes in the graph through the model. An example is enlightening. Imagine a graph of many separate components, each of which is a clique of widely varying size. A smart exploration algorithm would take samples from the cliques in descending order of their sizes. Observe that a truncated lookahead of k steps is only able to distinguish the value between cliques of size less than k. All cliques of size k or greater will look equal to the truncated look ahead algorithm. Such an algorithm will explore somewhat randomly until there are only cliques of size smaller than k left and suffer poor performance as a result. Our proposed IM, however, will exactly give all the nodes scores in proportion to their clique's size and it will make good exploration choices from the beginning.

Evaluation of the objective function requires repeated conditioning on single new label observations, which would require $O(n^3)$ time if we apply the criterion naively. We can reduce the computation by following the efficient update procedure suggested in [7] which will reduce the cost to $O(n^2)$ if we have already computed the matrix inversion at the very beginning.

4 Experimental Results

Data and Experimental Setting. We demonstrate our approach on two real-world datasets. (1) A citation network with 14,117 nodes (papers) and 42,019 edges (citation links, undirected) from citeseer, consisting of papers from the top 10 venues in Computer Science. The 1844 NIPS papers are labeled as targets. (2) 5271 webpages related to Programming Languages from Wikipedia with undirected links as edges. For each webpage we precompute its topic vector using the software available at [8]. 202 webpages highly related to "object oriented programming" are labeled as targets. Then we perform 20 random trials using a single randomly chosen positive node to initialize each trial. The averaged number of positives found as a function of iteration number is recorded.

We set $\eta = 0.5$ for both datasets, π is set to the true prior proportion of positives, and ω_0 is set to $1/n$, where n is the number of nodes. We test $\alpha = \{0, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$.

Baselines. We compare our approach with several baselines. (1) Uncertainty Sampling. We use our proposed f function as an approximation of $P(y_i = 1|L)$ and query the node with f value closest to 0.5. (2) Modified Upper Confidence Bound. UCB is not a natural fit but we modify UCB1 proposed in [5]. We assume that at first each node has been 'pulled' once and the prior is the information we get. We use our proposed f as the current estimation $\overline{x_j}$, and count the number of queried neighbors of node j as n_j. (3) 2-step lookahead from [1]. (4) Harmonic Function (f_u as proposed in [3]).

Results. Figure 1 shows the performance of our proposed model (with its best value of α) compared with the baselines on the two datasets. The results indicate that our proposed model has a clear advantage over baseline methods. The similar performance between our model and uncertainty sampling after about 600 iterations in the right figure is due to the low proportion of targets in the graph (the score falls below 0.5). Figure 2 shows the more detailed results of carrying out paired t-tests among some of the competitive methods.

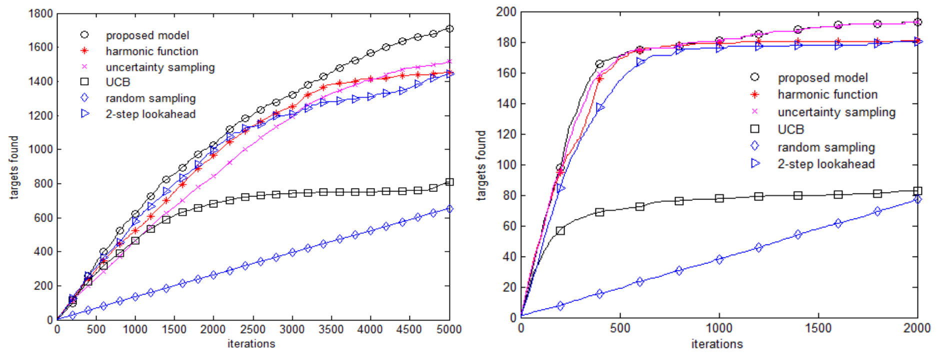

Figure 1: Proposed Model vs. Harmonic Function, Uncertainty Sampling, Upper Confidence Bound, Random Sampling, 2-step lookahead on the citation network (left) and wikipedia dataset (right).

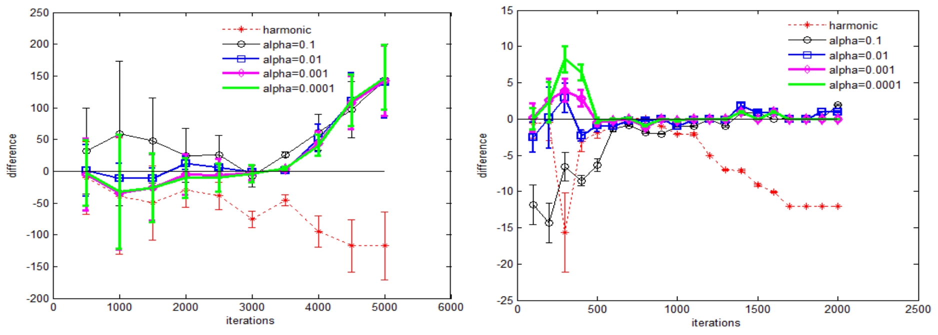

Figure 2: Difference in the number of positives found on the citation (left) and wiki (right) data by (1) Proposed model with $\alpha > 0$ and (2) Harmonic Function, compared to proposed model with $\alpha = 0$ (the zero line).

5 Future Work Discussion

There could be some possible extensions of this work. For example, setting α remains an unresolved issue. An automated method might consider a budget, B, of remaining choices to be made and set it accordingly. A reasonable setting might be $\alpha \sim (B - |L|)/B$, where $|L|$ is the size of labeled set. Alternatively, we might follow the form of the UCB algorithms and determine an adaptive value of α that allows us to derive regret bounds. However, doing so may be challenging given the fact that UCB methods are based on repeated arm pulls, while active search is not.

References

[1] R. Garnett, Y. Krishnamurthy, X. Xiong, J. Schneider, R. Mann. (2012) Bayesian Optimal Active Search and Surveying. *ICML.*

[2] M. Szummer, T. Jaakkola. (2001) Partial Labeled Classification with Markov Random Walks. *NIPS*.

[3] X. Zhu, Z. Ghahramani, J. Lafferty. (2003) Semi-supervised Learning Using Gaussian Fields and Harmonic Functions. *ICML.*

[4] A. Kuwadekar, J. Neville. (2011) Relational Active Learning for Joint Collective Classification Models. *ICML.*

[5] P. Auer, N. Cesa-Bianchi, P. Fischer. (2002) Finite-time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 27.

[6] N. Srinivas, A. Krause, S. Kakade, M. Seeger. (2010) Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. *ICML.*

[7] X. Zhu, J. Lafferty, and Z. Ghahramani. (2003) Combining active learning and semi-supervised learning using Gaussian fields and harmonic functions. *ICML workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining*.

[8] A. McCallum, ”MALLET: A Machine Learning for Language Toolkit.”, mallet.cs.umass.edu, 2002.