Node Classification in Uncertain Graphs

Michele Dallachiesa
University of Trento, Italy
dallachiesa@disi.unitn.it

Charu Aggarwal
IBM T.J. Watson
Research Center
charu@us.ibm.com

Themis Palpanas
Paris Descartes University
themis@mi.parisdescartes.fr

ABSTRACT

In many real applications that use and analyze networked data, the links in the network graph may be erroneous, or derived from probabilistic techniques. In such cases, the node classification problem can be challenging, since the unreliability of the links may affect the final results of the classification process. In this paper, we focus on situations that require the analysis of the uncertainty that is present in the graph structure. We study the novel problem of node classification in uncertain graphs, by treating uncertainty as a first-class citizen. We propose two techniques based on a Bayes model, and show the benefits of incorporating uncertainty in the classification process as a first-class citizen. The experimental results demonstrate the effectiveness of our approaches.

1. INTRODUCTION

The problem of collective classification is a widely studied one in the context of graph mining and social networking applications. In this problem, we have a network containing nodes and edges. A subset of the nodes in this network may be labeled. Typically, such labels may represent some properties of interest in the underlying network.

In such networks, only a small fraction of the nodes may be labeled, and these labels may be used in order to determine the labels of other nodes in the network. This problem is popularly referred to as *collective classification* or *label propagation* [6, 8, 9, 26], and a wide variety of methods have been proposed for this problem. The problem of data uncertainty has been widely studied in the database literature [7, 2, 3], and also presents numerous challenges in the context of network data [29].

In many real networks, the links[1] are uncertain in nature, and are derived with the use of a probabilistic process, with a probability value associated to each edge. Consider for example a biological network, whose links are derived from probabilistic processes and the edges have uncertainty associated with them; or a human interaction network that naturally includes uncertain links.

[1]In the rest of this paper we use the terms *network* and *graph*, as well as *link* and *edge*, interchangeably.

SSDBM '14, June 30 - July 02 2014, Aalborg, Denmark
Copyright 2014 ACM 978-1-4503-2722-0/14/06...$15.00.
http://dx.doi.org/10.1145/2618243.2618277

These networks can be represented as probabilistic networks, in which we have probabilities associated with the existence of links. Recent years have seen the emergence of numerous methods for uncertain graph management [14, 22] and mining [16, 13, 18], in which uncertainty is used directly as a first-class citizen. However, none of these methods address the problem of collective graph classification. One possibility is to use sampling of possible worlds on the edges in order to generate different instantiations of the underlying network. The collective classification problem can be solved on these different instantiations, and voting can be used in order to report the final class label. The major disadvantage with this approach is that the sampling process could result in a sparse or disconnected network which is not suited to the collective classification problem. In such cases, good class labels cannot be easily produced with a modest number of samples.

In this paper, we investigate the problem of collective classification in uncertain networks with a more direct use of the uncertainty information in the network. We design two algorithms for collective classification. The first algorithm uses a probabilistic approach, which explicitly accounts for the uncertainty in the links in the classification. The second algorithm works with the assumption that most of the information in the network is encoded in high-probability links, and low-probability links sometimes even degrade the quality. Therefore, the algorithm uses the links with high probability in earlier iterations, and successively relaxes the constraints on the quality of the underlying links. The idea is that a greater caution in early phases of the algorithm ensures convergence to a better optimum. The extended version of this paper can be found in [1].

In summary, we make the following contributions.

- We introduce the problem of collective classification in uncertain graphs, where uncertainty is associated with the edges of the graph, and provide a formal definition for this problem.
- We introduce two algorithms based on iterative probabilistic labeling that incorporate the uncertainty of edges in their operation. These algorithms are based on a Bayes formulation, which enables them to capture correlations across different classes, leading to improved accuracy.
- We perform an extensive experimental evaluation, using two real datasets from diverse domains. The results demonstrate the effectiveness of the proposed techniques and serve as guidelines for the practitioners in the field.

2. COLLECTIVE CLASSIFICATION

We first define uncertain networks, whose characteristic is that their edges may exist with some probability.

DEFINITION 2.1 (UNCERTAIN NETWORK). *An uncertain network is denoted by $G = (N, A, P)$, with node set N, edge set A and probability set P. Each edge $(i, j) \in A$ is associated with a probability value $p_{ij} \in P$. This is the probability that edge (i, j) exists in the network.*

We assume that the network is undirected, though the method can easily be extended to the directed scenario. We can assume that the $|N| \times |N|$ matrix P has entries which are denoted by p_{ij} and $p_{ij} = p_{ji}$. A node $i \in N$ can be associated with a label, representing its membership in a class. For ease in notation, we assume that node labels are integers. Given a set of labels S drawn from a set of integers $\{1 \ldots l\}$, we denote the label of node i by $L(i)$. If a node i is unlabeled, the special label 0 is used. We can now introduce the definition of the collective classification problem on uncertain graphs.

PROBLEM 2.1 (UNCERTAIN COLLECTIVE CLASSIFICATION). *Given an uncertain network $G = (N, A, P)$ and the subset of labeled nodes $T_0 = \{i \in N : L(i) \neq 0\}$, predict the labels of nodes in $N - T_0$.*

Figure 1 shows an example of an uncertain network. Nodes 1, 2, and 3 are labeled *white*, and nodes 5, 7, and 8 are labeled *black*. The label of nodes 4 and 6 is unknown. The aim of collective classification is to assign labels to nodes 4 and 6.

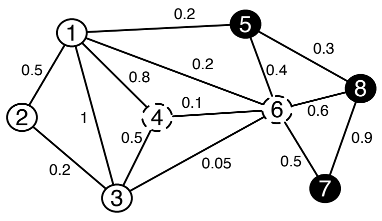

Figure 1: Example of uncertain network. Nodes $\{1, 2, 3\}$ are labeled *white* and nodes $\{5, 8, 7\}$ are labeled *black*, while labels for nodes $\{4, 6\}$ are unknown. Edges between nodes exist with some probability.

3. ITERATIVE PROBABILISTIC LABELING

3.1 Bayes Approach

The overall approach for the labeling process uses a Bayesian model for the labeling. In the rest of the paper, we refer to this algorithm as *uBayes*. Given that we have an unlabeled node r, which is adjacent to s other nodes denoted by $t_1 \ldots t_s$, how do we determine the label of the node r? It should be noted that the concept of adjacency is also uncertain, because the edges are associated with probabilities of existence. This is particularly true, when the edge probabilities are relatively small, since the individual network instantiations are likely to be much sparser and different than the probabilistic descriptions. Furthermore, for each edge (i, j) we need to estimate the probability of the node j having a particular label value, given the current value of the label at node i. This is done with the use of training data containing the labels and edges in the network. These labels and edges can be used to construct a Bayesian model of how the labels on the nodes and edges relate to one another.

The algorithm uses an iterative approach, which successively labels more nodes in different iterations. This is the set T of nodes whose labels will not be changed any further by the algorithm. Initially, the algorithm starts off by setting T to the initial set of (already) labeled nodes T_0. The set in T is expanded to $T \cup T^+$ in each iteration, where T^+ is the set of nodes not yet labeled that are adjacent to the labeled nodes in T. The algorithm terminates when T^+ is empty. The two most important steps are the computation of the edge-propagation probabilities and the expansion of the node labels with the use of the Bayes approach. For a given edge (i, j) we estimate $P(L(i) = p|L(j) = q)$. This is estimated from the data in *each iteration* by examining the labels of nodes which have already been decided. Therefore, the training process is successively refined in each iteration. Therefore, the value of $P(L(i) = p|L(j) = q)$ can be estimated by examining those edges for which one end point contains a label of q. Among these edges, we compute the fraction for which the other end point contains a label of p. For example, in the network shown in Figure 1 the probability $P(L(6) = black|L(5) = black)$ is estimated as $(0.3 + 0.9)/(0.3 + 0.9 + 0.2) = 0.85$. The label of node 6 is unknown, and it is not considered in the calculation. Note that this is simply equal to the probability that both end points of an edge are *black*, if one of them is *black*. Therefore, one can compute the uncertainty weighted conditional probabilities for this in the training process of each iteration.

This provides an estimate for the conditional probability. For an unlabeled node r, whose neighbors $i_1 \ldots i_s$ have labels $t_1 \ldots t_s$, we estimate its (unnormalized) probability by using the naive Bayes rule over all the adjacent labeled neighbors:

$$P(L(r) = p|L(i_1) = t_1 \ldots L(i_s) = t_s) \propto$$
$$P(L(r) = p) \cdot \prod_k P(L(i_k) = t_k|L(r) = p)$$

Note that the above model incorporates the uncertainty probabilities directly within the product term of the equation.

3.2 Iterative Edge Augmentation

The approach mentioned above is not very effective when a large fraction of the edges are noisy. In particular, if many edges have a low probability, this can have a significant impact on the classification process. Therefore, we use an iterative augmentation process in order to reduce the impact of such edges, by instead favoring the positive impact of high quality edges in the collective classification process. The idea is to *activate* only a subset of the edges for use on the modeling process. In other words, edges which are not activated are not used in the modeling. We call this algorithm *uBayes+*. We adopt a model inspired by automatic parameter selection in machine learning. Note that, analogous to parameter selection, the choice of a particular subset of high quality links, corresponds to a configuration of the network, and we would like to determine an optimal configuration for our approach. In order to do this, we split the set of labeled nodes T_0 into two subsets: a *training set* denoted by T_{train} and a *hold out* set denoted by T_{hold}. The ratio of the T_0 nodes that are assigned to the training set T_{train} is denoted by β, a user-defined parameter. The purpose of the hold out set is to aid optimal configuration selection by checking the precise value of the parameters at which the training model provides optimal accuracy over the set of nodes in T_{hold}. We use labels of nodes in T_{train} for the learning process, while using labels of nodes in T_{hold} as for the evaluation of accuracy at a particular configuration of the network. (Note that a label is never used for both the training and the hold out set, in order to avoid overfitting.)

We start off considering a small fraction of the high probability edges, iteratively expanding the subset of active edges by enabling some of the inactive edges with the highest probabilities. The ratio of active edges is denoted by the parameter θ. Ideally, we want to activate only the edges that contribute positively to the classification of unlabeled nodes. Given a configuration of active edges, we measure their goodness as the estimated accuracy on labels of nodes in T_{hold}. The value of θ that leads to the highest accuracy, denoted by θ^*, is used as the ratio of edges with the highest probability to activate on the uncertain network G. The resulting network is then used as input for the iterative probabilistic labeling algorithm (*uBayes*).

Despite optimizing accuracy by selecting the best ratio of edges to be considered, the basic model described above is not very efficient, because it requires multiple evaluations of the iterative probabilistic labeling algorithm. In particular, it requires us to vary the parameter θ and evaluate accuracy, in order to determine θ^*. A more efficient technique for identifying θ^* can be obtained by evaluating the accuracy for different values of θ on a sample of the uncertain network G (rather than the full network) as follows. We generate a new uncertain network $G' = (N', A', P')$ by sampling $\alpha \cdot |N|$ nodes from G uniformly at random, and retaining the edges from A and probabilities from P referring to these sampled nodes. α is a user-defined parameter that controls the ratio of nodes sampled from G and it implies the size of the sampled uncertain network G'. The optimal configuration of the θ^* parameter is obtained evaluating the *uBayes+* algorithm on the sampled uncertain network G'. θ^* is then used to activate $\theta^*|N|$ edges with highest probability in G before labeling the nodes of G with the *uBayes+* method.

4. EXPERIMENTAL RESULTS

In this section, we evaluate the proposed techniques under different settings, in terms of both accuracy and performance. We implemented all techniques in C++, and ran the experiments on a Linux machine with Intel Xeon 2.40GHz processor and 16GB of RAM.

Data Sets: In our experiments, we used two data sets derived from the DBLP and US Patent citation networks. Class labels are derived from the bibliographic conference information and standard US Patent classification, respectively. Edge probabilities are estimated as the normalized edge frequency among either paper authors or patent assignees. We also used perturbed data sets to stress-test the methods. Perturbed data sets are generated by adding noisy edges.sets.

Evaluation Methodology: The accuracy is assessed by using repeated random sub-sampling validation. We randomly partition the nodes into training and validation subsets which are denoted by N_T and N_V respectively. We use $2/3$ of the labeled nodes for training, and the remaining $1/3$ for validation. We compared our techniques to two algorithms, which are the *wvRN* [20] and *Sampling* methods. We limited the running time of these two algorithms to the time spent by the *uBayes* method. The *wvRN* method estimates the probability of node i to have label j as the weighted sum of class membership probabilities of neighboring nodes for label j. We additionally consider a version of the *wvRN* algorithm, *wvRN-20*, that is not time-bounded, but is bound to terminate after 20 iterations in the label relaxation procedure. The sampling algorithm samples networks in order to create deterministic representations. For each sampled instantiation, the *RN* algorithm [20] is used.

Classification Quality Results: The first experiment shows the accuracy by varying the ratio of noisy edges (ϕ) for the algorithms *uBayes*, *uBayes+*, *wvRN*, *wvRN-20* and *Sampling*. The results for the *DBLP* data set are reported in Figure 2. The *Sampling* algorithm is the worst performer on both data sets, followed by *wvRN* and *wvRN-20*. The *uBayes* and *uBayes+* algorithms are the best performers, with *uBayes+* achieving higher accuracy on the *DBLP* dataset when the ratio of noisy edges is above 200%. We observe that there is nearly no difference among the *uBayes*, *uBayes+*, *wvRN* and *wvRN-20* algorithms on both datasets when $\phi = 0$, while the percentage improvement in accuracy from *wvRN-20* to *uBayes+* when $\phi = 5$ (500%) is up to 49% for *DBLP* and 7% for *Patent*. It is worth noting that, as the ratio of noisy edges increases, the accuracy for *Sampling* increases in the Patent data set. This is due to the high prior probability of one class (0.434).

In the next experiment, we varied the standard deviation of the probability of the noisy edges (σ) for algorithms *uBayes*, *uBayes+*, *wvRN*, *wvRN-20* and *Sampling*. The results for the *DBLP* data set are reported in Figure 3. The *Sampling* algorithm again does not perform well, followed by the *wvRN* and *wvRN-20* algorithms. The *uBayes+* algorithm is consistently the best performer on the *DBLP* dataset, while there is nearly no difference between the *uBayes+* and *uBayes* algorithms on the *Patent* data set. The higher accuracy of *uBayes* and *uBayes+* is explained by their ability to better capture correlations between different class labels, a useful feature when processing noisy data sets. The better performance of *uBayes+* is due to its ability to ignore noisy labels that contribute negatively to the overall classification process. *uBayes+* is more accurate than *wvRN-20* with a percentage improvement up to 83% in the *DBLP* data set and 10% in the *Patent* data set, which represents a significant advantage.

In the following experiment, we evaluate the accuracy when varying the ratio of labeled nodes (Γ) for algorithms *uBayes*, *uBayes+*, *wvRN*, *wvRN-20* and *Sampling*. The results are omitted for brevity. In the *DBLP* dataset, the *wvRN* algorithm performs better than *wvRN-20*, while there is virtually no difference on the *Patent* dataset. The *uBayes+* algorithm is consistently the best performer on the *DBLP* dataset, while it performs slightly worse than *uBayes* on the *Patent* data set when Γ is below 0.2 (20%). We observe that the percentage improvement of *uBayes+* over *wvRN-20* is 50% on the *DBLP* dataset and 11% on the *Patent* dataset. The *Sampling* algorithm exhibits the lowest accuracy.

Efficiency Results: Figure 4 shows the CPU time required by the algorithms when varying the ratio of noisy edges for the *DBLP* data set. Note that *Sampling* has the same time performance as *uBayes*. The *uBayes+* algorithm is nearly three times slower than *uBayes*. This is due to the automatic parameter tuning approach employed by the *uBayes+* algorithm. We observe that the performance of *wvRN-20* is almost always *considerably* worse than both *uBayes* and *uBayes+*. The same observation is true when varying the standard deviation of the probability of the noisy edges (see Figure 5). Note that the inference in the *wvRN* algorithm is based on labeling relaxation, whose complexity is proportional to the size of the network and remains constant across iterations. On the contrary, the iterative labeling that *uBayes* and *uBayes+* uses for their inference model becomes faster with each successive iteration, since it needs to visit a smaller part of the network. As the results show, the standard deviation does not affect the time performance of the algorithms. These experiments demonstrate that the two proposed algorithms effectively combine low running times with high accuracy and robustness levels.

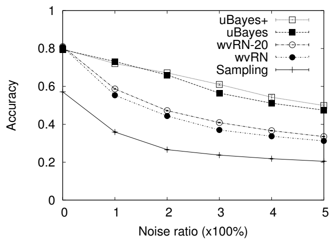

Figure 2: Accuracy with varying ratio of noisy edges.

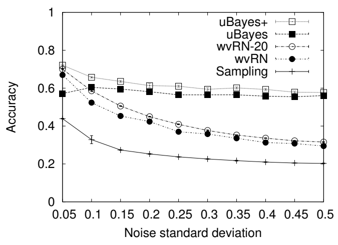

Figure 3: Accuracy with varying standard deviation of probability of noisy edges.

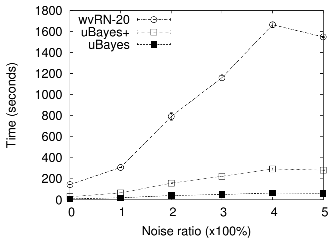

Figure 4: Time performance with varying ratio of noisy edges.

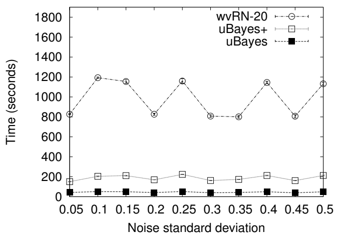

Figure 5: Time performance with varying standard deviation of probability of noisy edges.

5. RELATED WORK

The problem of node classification has been studied in the graph mining literature, and especially relational data in the context of *label or belief propagation* [23, 25, 26]. Such propagation techniques are also used as a tool for semi-supervised learning with both labeled and unlabeled examples [27]. Collective classification [20, 19, 8] refers to semi-supervised learning methods that exploit the network structure and node class labels to improve the classification accuracy. These techniques are mostly based on the assumption of homophily in social networks [21]: neighboring nodes tend to belong to the same class. A technique has been proposed in [19], which uses link-based similarity for node-classification in directed graphs. Recently, collective classification methods have also been used in the context of blogs [8]. In [9], Bilgic et al. discuss the problem of overcoming the propagation of erroneous labels by asking the user for more labels. A method for performing collective classification of email speech acts has been proposed by Carvalho et al. in [10], exploiting the sequential correlation of emails. In [12], Ji et al. integrate the classification of nodes in heterogeneous networks with ranking. Methods for leveraging label consistency for collective classification have been proposed in [25, 26, 27]. A comprehensive review of the proposed models and algorithms can be found in [5]. The problem of uncertain graph mining has also been investigated extensively. The most common problems studied in uncertain graph management are those of nearest neighbor query processing [22], reachability computation [15] and subgraph search [24]. In the context of uncertain graph *mining*, the problems commonly studied are frequent subgraph mining [28], reliable subgraph mining [14], and clustering [13]. Recently, the problem of graph classification has also been studied for the uncertain scenario [16], though these methods are designed for classification of many small graphs, in which labels are attached to the entire graph rather than a node in the graph.

6. CONCLUSIONS

Uncertain graphs are becoming increasingly popular in a wide variety of data domains. In this paper, we formulate the collective classification problem for uncertain graphs, and describe effective and efficient solutions. We describe an iterative probabilistic labeling method, based on the Bayes model, that treats uncertainty on the edges of the graph as first class citizens. In the proposed approach, the uncertainty probabilities of the links are used directly in the labeling process. We have performed an experimental evaluation of the proposed approach using diverse, real-world datasets. The results show significant advantages of using such an approach for the classification process over more conventional methods, which do not directly use uncertainty probabilities.

Acknowledgments

Work of the second author was sponsored by the Army Research Laboratory under cooperative agreement number W911NF-09-2-0053.

7. REFERENCES

[1] M. Dallachiesa, C. C. Aggarwal, and T. Palpanas. Node Classification in Uncertain Graphs *arXiv*, 2014, to appear.

[2] M. Dallachiesa, I. F. Ilyas, and T. Palpanas. Top-k Nearest Neighbor Search In Uncertain Data Series *PVLDB*, 2015, to appear.

[3] M. Dallachiesa, B. Nushi, K. Mirylenka, and T. Palpanas. Uncertain time-series similarity: Return to the basics. *PVLDB*, 2012.

[4] L. Eronen and H. Toivonen. Biomine: predicting links between biological entities using network models of heterogeneous databases. *BMC*, 2012.

[5] C. C. Aggarwal and P. S. Yu. A survey of uncertain data algorithms and applications. *TKDE*, 2009.

[6] C. Aggarwal, H. Wang. Managing and Mining Graph Data, Springer, 2010.

[7] C. Aggarwal. Managing and Mining Uncertain Data, Springer, 2009.

[8] S. Bhagat, G. Cormode, I. Rozenbaum. Applying link-based classification to label blogs, *WebKDD/SNA-KDD*, 2007.

[9] M. Bilgic, L. Getoor. Effective label acquisition for collective classification, *SIGKDD*, 2008.

[10] V. de Carvalho, W. Cohen, On the collective classification of email “speech acts", *SIGIR*, 2005.

[11] B. Hall, A. Jaffe, M. Trajtenberg. The NBER patent citation data file: Lessons, insights and methodological tools, *National Bureau of Economic Research*, 2001.

[12] M. Ji, J. Han, M. Danilevsky. Ranking-based classification of heterogeneous information networks. *SIGKDD*, 2011.

[13] G. Kollios, M. Potamias, E. Terzi. Clustering large probabilistic graphs. *TKDE*, 2011.

[14] R. Jin, L. Liu, C. Aggarwal. Discovering Highly Reliable Subgraphs in Uncertain Graphs, *SIGKDD*, 2011.

[15] R. Jin, L. Liu, B. Ding, H. Wang. Distance-constraint reachability computation in uncertain graphs. *VLDB*, 2011.

[16] X. Kong, P. Yu, X. Wang, A. Ragin. Discriminative Feature Selection for Uncertain Graph Classification. *SDM*, 2013.

[17] M. Ley, S. Dagstuhl. *DBLP Dataset*, http://dblp.uni-trier.de/xml/ August, 2012

[18] L. Liu, R. Jin, C. Aggrawal, Y. Shen. Reliable Clustering on Uncertain Graphs, *ICDM*, 2012.

[19] Q. Lu, L. Getoor, Link-based classification, *ICML*, 2003.

[20] S. Macskassy, F. Provost. A simple relational classifier. *Technical report*, 2003.

[21] M. McPherson, L. Smith-Lovin, J. Cook. Birds of a feather: Homophily in social networks. *Annual review of sociology*, 2001.

[22] M. Potamias, F. Bonchi, A. Gionis, G. Kollios. k-nearest neighbors in uncertain graphs. *VLDB*, 2010.

[23] B. Taskar, P. Abbeel, D. Koller, Discriminative probabilistic models for relational data, *UAI*, 2002.

[24] Y. Yuan, G. Wang, H. Wang, L. Chen. Efficient subgraph search over large uncertain graphs. *VLDB*, 2011.

[25] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, B. Schölkopf, Learning with local and global consistency, *NIPS*, 2004.

[26] D. Zhou, J. Huang, B. Schölkopf, Learning from labeled and unlabeled data on a directed graph, *ICML*, 2005.

[27] X. Zhu, Z. Ghahramani, J. Lafferty, Semi-supervised learning using gaussian fields and harmonic functions, *ICML*, 2003.

[28] Z. Zou, H. Gao, J. Li. Discovering frequent subgraphs over uncertain graph databases under probabilistic semantics. *SIGKDD*, 2010.

[29] J. Ren, S. D. Lee, X. Chen, B. Kao, R. Cheng, and D. Cheung. Naive bayes classification of uncertain data. In *ICDM*, 2009.