Within-network classification with label-independent features and latent linkages

Christopher Alan Ryther
University of Copenhagen
Njalsgade 128-132
DK-2300 Copenhagen S
Denmark
ryther@di.ku.dk

Jakob Grue Simonsen
University of Copenhagen
Njalsgade 128-132
DK-2300 Copenhagen S
Denmark
simonsen@di.ku.dk

Andreas Koch
University of Copenhagen
Njalsgade 128-132
DK-2300 Copenhagen S
Denmark

ABSTRACT

We study within-network classification in sparsely labelled networks, presenting two separate contributions: (A) a thorough reproduction of a label-independent method for classification from recent research using statistical relational learning (SRL) and semi-supervised learning (SSL), and (B) a novel approach that utilizes node attribute information to improve SRL and SSL classifier performance called Attribute Network Propagation (ANP). (B) uses a method of linearly combining predictions with a procedure of transforming node attributes into graph edges. For both contributions we use two well-known real-world datasets: the reality mining (RM) cell phone calls dataset and the Cora Portal publication citations dataset (CORA), both formulated as binary classification problems. We employ an existing classification framework to run 10 individual SRL- and SSL-based classifiers and evaluate performance using the area under the ROC curve (AUC). Results from (A) confirms that label-independent features can improve the performance of some relational classifiers using iterative methods, but in most cases, 26 out of 30, deteriorates performance on existing baselines. Results from (B) show that in 91 out of 100 cases it is possible to improve the performance of relational classifiers with ANP.

CCS Concepts

•**Computing methodologies** → *Machine learning;*

Keywords

Within-Network Classification; Graphs; Latent Features

MGL '16 August 14, 2016, San Fransisco, CA, USA

DOI: NotApplicable.

1. INTRODUCTION

Numerous problems concerning real-world phenomena involve classification of nodes in networks, for instance:

- Anomaly detection: detecting intrusions in networks based on network traffic. Identified intrusions make up for a small fraction of all traffic in and out.
- Social Network Analysis: Categorising and modelling user behaviour to be used in e.g. targeted advertising or anti-terrorism operations. Typically contains large amounts of unlabelled data and a small set of labelled nodes.
- Cell phone fraud: Cell phone fraud is an example where networks are often very sparsely labelled. We have a handful of known fraudsters and legitimate users, but the labels are unknown for the vast majority of users.

Other examples include classification of documents and webpages, protein interactions, and product recommendation systems.

More specifically, the problem of (univariate) within-network classification is the following: *given* a quadruple $G(V, E, W, L)$ where (V, E) is a (possibly directed) graph, a set of labels W, and a subset $L \subseteq V$ such that each node $v \in L$ has one or more known labels from W, *find* labels from W for each of the nodes in $V \setminus L$. Typically, the labels are sparse (i.e., L is small compared to V, for large real-world networks, often $|L|/|V| \leq 1\%$).

To tackle this problem, several modern approaches try to augment the traditional node-centric approach (using only information directly attributed to the individual nodes) with *relational* data. Relational data differs from traditional data in an important way: it violates the instance-independence assumption. The core concept of relational methods is to take advantage of these dependencies between instances. Statistical relational learning (SRL) algorithms have been shown to perform well on within-network classification problems [8], especially when two phenomena are present in the data: homophily and/or co-citation regularity. Homophily is the correlation between two connected nodes and their individual labels. Co-citation regularity is related and holds true when individuals have a tendency to connect to the same objects in networks. Under these circumstances, the label-propagation algorithms can assign labels throughout the network, using edges between nodes as "pathways", to successfully classify previously unlabelled nodes[7]. However

because of the reliance on existing ground truths, relational classifiers can degrade significantly when labels are sparse.

Recent research has proposed various methods to tackle the label sparsity. Collective classification methods have been shown to improve performance when labels are sparse[15, 16]. Other methods synthetically construct additional relational data, e.g. directly adding new edges to the graphs [4], and other approaches employ more complex ways of combining various attributes and graph structural properties, called latent graphs, like in [16, 10]. The idea of latent graph methods is to use existing (node) attributes in graph datasets in order to construct new – so-called *latent* – features or graph augmentations in the underlying graph; with the new features, a supervised classifier can then predict class membership for remaining unlabelled nodes [17, 16, 18].

For instance, Tang uses spectral clustering and modularity maximization [17, 18] to generate a new set of latent features for each node. Another example, the Latent Network Propagation (LNP) algorithm by Shi et al. [16], transforms the original graph by adding weighted edges, where the weights are determined by factors such as attribute similarity or node proximity in the original graph. The LNP algorithm uses quadratic programming to add weight to edges between nodes that are likely to share the same class label. Such indirect edges are called "latent linkages" and are already a part of the dataset, but need processing before becoming part of the graph structure.

A particularly simple version of this approach is simply to generate new edges in the underlying graph; using the augmented graph, with the new edges, a label propagation algorithm makes its final class membership predictions. The idea is that adding edges in this manner allows the ground truths to propagate more effectively throughout the graph[11]. By careful design and choice of latent edges, one can improve classification reliability without the need for relational features or more complex collective inference.

Our contribution: We extend the idea of using latent linkages by creating new graphs with edges based on attribute similarity, but running inference separately, on both the original graph and the newly constructed latent attribute graph, and finally combining predictions. We also experiment with a simpler method, merging latent attribute features directly into the original graph, to test whether the added complexity of the first approach provides any improvement of classification.

1.1 Related Work

Traditional statistical relational techniques have made use of label-dependent features. Lu and Getoor's approach uses logistic regression to model class membership using neighboring nodes [6], Macskassy and Provost use weighted sums of neighboring nodes' labels [8] for label propagation, and Neville and Jensen [13] use spectral clustering to group nodes based on their local edge structure, which in turn are used in learning classifiers.

Handling label-sparsity has been extensively studied: one approach is to use an iterative procedure, Iterative Classification Algorithm (ICA), that feeds predictions back into the network, which in turn are used to inform subsequent inferences as done by Neville in [12]. ICA is reported as a somewhat robust method which can be governed by a process called Gibbs sampling [6, 5]. Collective classification has also been proposed to overcome this problem [5, 15], where they key idea is to combine supervision knowledge about the graph with edge-structural information from the graph. However, previous work has shown that even collective classification can suffer when subjected to very sparsely labelled graphs [14].

Another approach aims to incorporate extra information, e.g. attribute information, already present in the data into the graph, to strengthen classification by providing more edges for information to travel through. Gallagher et al. propose a method where "ghost" edges are added to the original network to enable flow of information to hard-to-reach unlabelled nodes. A similar design by Macskassy [8] adds additional edges to the graph based on text-similarity between nodes in the data. Shi et al. [16] transform the dataset into a fully connected graph with latent edges, where edge weights are maximized between training data with the same labels. Tang and Liu [17] extract social latent dimensions, like affiliations, combined with discriminative learning to outperform relational collective classification methods. Finally work by Fleming [2] and McDowell et al. [11] compared latent edge methods with state-of-the-art non-latent methods and confirmed that these types of algorithms can perform competitively, but don't necessarily perform consistently. Our proposed method differs from previous approaches by keeping latent edges in separate graphs instead of adding them to the existing edges.

2. METHODOLOGY

To address the problem of label sparsity we propose two methods: **Attribute Network Propagation (ANP)** and **MixedEdges (ME)** that exploit latent linkages between nodes based on attribute information already present in data. For example, in a publication dataset like CORA, in addition to citations between articles we also have a list of keywords present in each of the articles' content. It can be expected that nearly all of these keywords appear multiple times across several papers. By adding edges between articles, that share keywords, it is possible to derive latent linkages in the dataset. The intuition behind this idea is that with the added edges, the ground truths may be able to propagate throughout the network more effectively. The problem lies in selecting the most useful shared attributes to transform into edges and in what way they are best utilized by relational classifiers.

The two methods require generation of two types attribute-augmented graphs, which we name after the dataset they are based on, with added suffixes: **-ANP** or **-ME**. The -ANP graphs are used by Attribute Network Propagation and the -ME graphs are used by MixedEdges.

The -ANP graphs are created by first making a copy of the original graph, without edges, from the original dataset. The -ME graphs are based completely in the original graph, nodes and edges. Then, for both -ANP and -ME graphs, unweighted, undirected edges are added between nodes depending on dataset-specific conditions, explained in detail

in Section 3.4. These conditions are derived from available attribute information in the individual datasets.

For -ME sets all added edges are merged into the original graph, but for the ANP algorithm, the two sets of graphs (original and -ANP) remain separate. Informally, the nodes of the -ANP graph are only connected based on their shared attributes and the nodes of the -ME graph are connected using both shared attributes and the original edges. As described earlier in this section, the motivation behind choosing these attributes is the assumption that they connect unlabelled nodes to more labelled nodes than in the original graph (ME) or that attributes edges alone can uncover more meaningful paths for labels to propagate. The attribute graphs -ANP are each based on multiple attributes, which alternatively could be split into separate graphs, one for each attribute. This was decided against due to very low edge counts, ($\leq$ 100 edges) for most attributes, which we expect would lead to a decrease in performance of label-propagation methods.

For the **ME**, node class membership is determined simply by using relational classifiers directly on the -ME graphs. For **ANP** we first run the relational classifiers on both the original and the -ANP graphs individually. After, ANP combines the results from -ANP sets with the original graphs in a fashion resembling how the individual classifiers are combined in Section 3.2. The final prediction of each node's class is linear combination of results on the original dataset and the shared attribute graph:

$$P(C) = w \cdot P_{baseline}^{original}(C) + (1 - w) \cdot P_{baseline}^{-ANP}(C) \qquad (1)$$

Similar to equation (4) the weight w is calculated based on the individual performance of the baseline over 10-fold cross-validation on the original dataset and the -ANP dataset. The area under the receiver operating characteristic ROC curve (AUC) is calculated for each fold and then an average AUC score for each dataset, $AUC_{baseline}^{original}$ and $AUC_{baseline}^{-ANP}$, is obtained. The weighting parameter w is then defined as:

$$w = \frac{AUC_{baseline}^{original}}{AUC_{baseline}^{original} + AUC_{baseline}^{-ANP}} \qquad (2)$$

The classification processes, ANP and ME on their respective graphs, -ANP and -ME, is shown in Figure 1.

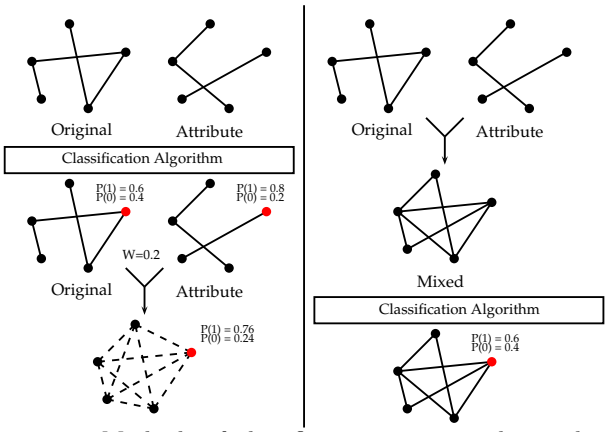

Figure 1: Methods of classification using attribute edges. On the left are is the ANP method using and on the right is the ME method. The red node is an arbitrary node in the graph, used to show the process of calculating its class membership $P(C)$. W is the pre-calculated weight using 10-fold cross-validation.

3. EXPERIMENTAL DESIGN

In the following we first present details about the classifiers chosen and how label-independent features are used. Following is a description of the real-world datasets, their characteristics and how they are sampled. The last part explains the experiment methodology in detail.

3.1 Label-Independent Features

Relational classifiers typically rely on the network edge-structure to make use of label information from neighbouring nodes. Another approach to creating relational features is to use the graph-structural properties of the surrounding nodes, such as node degree or other graph-metrics. The methods of creating relational features can be divided into two categories, label-dependent features and label-independent features.

An example using label-dependent features is the network-only edge-based classifier (nLB) from Lu et al. [6]. It models a node's class based on the classes of neighbouring nodes. A node's neighbourhood is summarized by edge-weighted counts of neighbouring nodes for each class. One disadvantage to this approach is that when labels are sparse, the relational features become useful since neighbouring nodes will tend not to be labelled.

Label-independent (LI) features are calculated using only the structural properties of the graph – labels and attributes are not included. The hypothesis is there is a correlation between the class label of a node and the network structure and that they can be used in cases where label-sparsity impedes traditional relational features. However, results have shown that label-independent features alone are not enough to model class membership in the graph, therefore additional information is needed. From the design of Gallagher & Eliassi-Rad [3] we use a logistic regression classifier (logLI) that is trained on the following four label-independent node features: number of neighbouring nodes (node degree), number of incident edges, the betweenness centrality coefficient, and the clustering coefficient. The results using label-independent features are then combined with predictions from baseline classifiers, described in detail in Section 3.2.

3.2 Classifiers

The work is based on 10 individual classifiers, some of which are hybrid methods combining results from multiple algorithms. In order to use the label-independent features of [3] we combine baseline classifiers with the logistic regression model "logLI", trained on the LI features, from Section 3.1. We use the implementations in NetKit, a modular toolkit for classification in networked data by Masckassy et al.[8]. The full list of methods used is as follows:

- wvRN: The weighted-vote relational neighbor classi-

fier which is a non-learning classifier that uses label-propagation and an average weighted sum of neighboring nodes' labels to determine class membership[8]. The wvRN classifier calculates the class membership of a node i as:

$$P(C_i = c|N) = \frac{1}{L_i} \sum_{j \in N} \begin{cases} w_{i,j} \text{ if } C_i = c \\ 0 \text{ otherwise} \end{cases} \quad (3)$$

where N is set of neighbors for node i, $w_{i,j}$ is the number of edges between node i and j and L_i is the number edges connecting i to labelled nodes.

- wvRNICA: Uses the wvRN classifier with collective classification as described in Section 3.2.1.
- wvRN+li: Is a combination of wvRN and the logLI logistic regression classifier, described in Section 3.1.
- wvRNICA+li: combination of wvRNICA and logLI.
- nLB: The network-only edge-based classifier [6]. Uses a logistic regression model to infer a node's class. A node's summary is the number of neighboring nodes. A node's features are neighboring nodes' summaries.
- nLBICA: Uses the nLB classifier with collective classification as described in Section 3.2.1.
- nLB+li: Is a combination of nLB and logLI, described in Section 3.1.
- nLBICA+li: Is a combination of nLBICA and logLI, described in Section 3.1.
- GRF: Is the semi-supervised Gaussian Random Field classifier of Zhu et al.[19].
- GRF+li: Is a combination of GRF and logLI, described in Section 3.1.

The methods which use logLI to improve predictions, calculate the probability of each class as:

$$P(C) = w \cdot P_{baseline}(C) + (1 - w) \cdot P_{logLI}(C) \quad (4)$$

where $P_{baseline}(C)$ is the class membership prediction of the baseline, i.e. wvRN, nLB, GRF, and $P_{logLI}(C)$ is the prediction from logLI. The weight w is calculated once per dataset and is used throughout all trials for that specific graph. The calculation is based on the individual performance of the baseline algorithm and logLI over 10-fold cross-validation on the labelled nodes of the dataset. The area under the receiver operating characteristic (ROC) curve (AUC) is calculated for each fold and an average AUC score for each classifier, $AUC_{baseline}$ and AUC_{logLI}, is obtained. The weighting parameter w is then defined as:

$$w = \frac{AUC_{baseline}}{AUC_{baseline} + AUC_{logLI}} \quad (5)$$

thereby enabling utilization of label-independent features. The intuition is that the label-independent features should be used to a degree based on their expected performance.

3.2.1 Collective Classification

To perform collective classification (CC) we use the Iterative Classification Algorithm (ICA) up to 1000 iterations with both the nLB (nLBICA) and wvRN (wvRNICA). We chose the Iterative Classification Algorithm over other methods, eg. Gibbs Sampling, to follow the methodology described in [3]. For partially labelled datasets, there are two approaches when using ICA: Perform collective classification on the entire graph or perform collective classification only on the core set of nodes. The latter is chosen because it is has been shown to perform better[3].

3.3 Datasets

We employ data from two sources, chosen for their appearance in the research papers that inspired this article: Reality Mining (RM) cell phone calls/texts[1] and CORA Research Paper Classification Dataset.[2]. CORA is a co-citation dataset originally created by crawling the internet for machine learning articles [9]. Shi et al. use the dataset to perform multi-class classification using latent graphs [16]. The publications themselves correspond to nodes in the graph and citations correspond to edges between publications. Each node is categorized into one of seven classes: Case Based, Genetic Algorithms, Neural Networks, Probabilistic Methods, Reinforcement Learning, Rule Learning and Theory. To pose a binary classification problem in the context of the CORA dataset, we formulate the classifiers' task as: identify papers with the topic "Probabilistic Methods".

The CORA data is already cleaned and pre-processed by [15]. Each publication in the dataset is represented by a label (machine learning topic) and a 0/1-valued word vector indicating the absence/presence of the corresponding word from the dictionary. The dictionary consists of 1433 unique words and there are 7 topics as the set of labels.

The RM data originates from an experiment at a college campus where 100 mobile phones were tracked over the course of 9 months [1]. Gallagher et al. used the collected data to create a graph where each user is considered a node and cell phone communications are considered edges between users [3]. They used the graph to evaluate a classification method using label-independent features. For the Reality Mining data our task is to identify whether or not a person is a student. In the RM graph there exists a subset of nodes, we call "core" nodes, for which we known the true class labels. For the rest of the RM nodes there exists no true class labels.

With RM we remove from the dataset any participants who do not communicate with other individuals. The resulting graph is then sampled using breadth first search, as done in [3], following edges based on communication. The pseudocode for the procedure is show in algorithm 1. Nodes are sampled using BFS starting from a core node. When 1000 nodes have been chosen, all edges present between them are added to the sample. The breadth-first sampling of the RM datasets in algorithm 1 is sensitive to which core node is chosen as the seed. To overcome this we create

[1] http://realitycommons.media.mit.edu/realitymining.html
[2] http://linqs.umiacs.umd.edu/projects/projects/lbc/

Algorithm 1: BFS Dataset Sampling

```
input: Graph G_i(E_i, V_i), SampleSize
output: Graph G_o(E_o, V_o)
begin
  nodequeue ← getCoreNode(V_i)
  G_o(E_o, V_o) = ∅
  while notEmpty(nodequeue) and |V_o| < SampleSize
    v_i ← pop(nodequeue)
    V_o ← V_o ∪ v_i
    nodequeue ← nodequeue ∪ (getNeighbors(v_i) \ V_o)
    if |V_o| == SampleSize
      break
  end
  foreach v_o in V_o
    E_o ← E_o ∪ allEdges(v_o, V_o)
  return G_o(E_o, V_o)
end
```

Table 1: Dataset characteristics

Data Set	$\|V\|$	$\|E\|$	$\|L\|$	$P(+)$
RM Full	10.050	140.703	101	0.63
RM Samples	1K	16k-42k	18-92	0.04-1.00
RM-ANP Samples	1K	100-1800	18-92	0.04-1.00
RM-ME Samples	1K	16k-44k	18-92	0.04-1.00
CORA	2708	5429	2708	0.16
CORA-ANP	2708	220k	2708	0.16
CORA-ME	2708	225k	2708	0.16

|core nodes| = $|L|$ number of datasets $G_o^i(E_o^i, V_o^i)$ for RM, one for each possible core node seed. The order of RM seed core nodes is unimportant since the experiments are run on the resulting samples independently of each other.

Table 1 contains the following information about the original and sampled datasets from RM as well as the CORA co-citation network: $|V|$ is the total number of nodes in the dataset, $|L|$ is the number of labelled nodes, $|E|$ is the number of edges and $P(+)$ is the fraction of labelled nodes which have a positive class label. Since the RM dataset is sampled multiple times, we give intervals in which the individual characteristics lie. The sets with suffix '-ANP' and '-ME' are datasets based on latent attribute graphs described in Section 2.

3.4 Experimental Methodology

Classifiers have access to the entire graph during both training and evaluation, but in the experiments we hide a proportion of the labelled nodes' labels, so they can be used for evaluation. The proportion of labelled nodes used for training is varied from 0.1 to 0.9 in 0.2 size increments. For each proportion we run 30 *trials* with each classifier. For each trial we perform a class-stratified sampling containing $100 \times$ (proportion labeled)% nodes as a training set and the remaining as a test set. The samples are chosen carefully so that each node appears the same amount of test sets over all trials. The train/test splits are the same for all classifiers and across the -ME -ANP types as well. The experiment setup described is repeated for each graph G_o^i sampled of RM. The process is shown in algorithm 2.

In this work we extract latent linkages from two datasets: a network of communications (RM) and an article citation network (CORA). The attributes extracted from RM are based on a questionnaire filled out by participants in the

Algorithm 2: RM -ANP -ME Experiment Setup

```
input: Graphs G_o^i(E_o^i, V_o^i)
output: AUC
begin
for graph G_i(E_i, V_i) in G_o^i(E_o^i, V_o^i)
  for c in classifiers
    10-fold cross-validation on G_i using c
  Calculate w from averaged AUC
  for p in proportion {0.1,0.3,0.5,0.7,0.9}
    for trial in 30 trials:
      testset = getClassStratifiedSampling
      trainingset = G\testset
      Initialize uniform priors
      Run classifiers using trainingset labels
      Use weight w to combine predictions
      Calculate AUCs for trial
    end
    Average AUC per ratio for all trials
  end
end
return avg. AUC per ratio for all graphs G_o^i
```

original paper and in CORA we use article keywords. As described in Section 2 we add edges depending on dataset-specific conditions. The conditions are the same for both -ANP and -ME graphs:

- RM: Two nodes are connected if, based on their answers in the dataset's questionnaire, they both consider each other a friend *or* they spend any time in the same physical vicinity. The intuition here is that students tend to be friends, and spend time on campus, with other students and less so with people who are not students.
- CORA: Two nodes are connected if they share any of the first 1200 word-vector features, sorted by ascending frequency, which are included in the dataset. Without using a cutoff – thereby including all word-vectors – resulted in a heavily connected graph ($\approx$ 2.5M edges) where inference was computationally infeasible. The idea behind is that there exists a strong correlation between articles' use of globally infrequent words, and the topic of the article. By connecting articles that use the same words we hope to improve relational classifiers.

We use the area under the ROC curve (AUC) to compare the classifiers' performance since the class distribution, shown in table 1 under $P(+)$, is skewed and the accuracy measure alone therefore is not discriminative enough. For the RM samples we compute average AUCs for every sample and finally report the harmonic mean over all samples' AUCs.

4. RESULTS

Results for baselines and addition of label-indpendent features are shown in Figures 2 and 3. Clearly, the baselines' performance suffers when the ratio of labelled nodes is around ≈ 0.1 on the original datasets. The lacklustre performance of the baselines is most likely due to the sparse labelling: the algorithms employed rely on label propagation, but without enough labels the original network may not propagate correct labels effectively. In addition, the baselines cannot, in their original setup, make use of latent

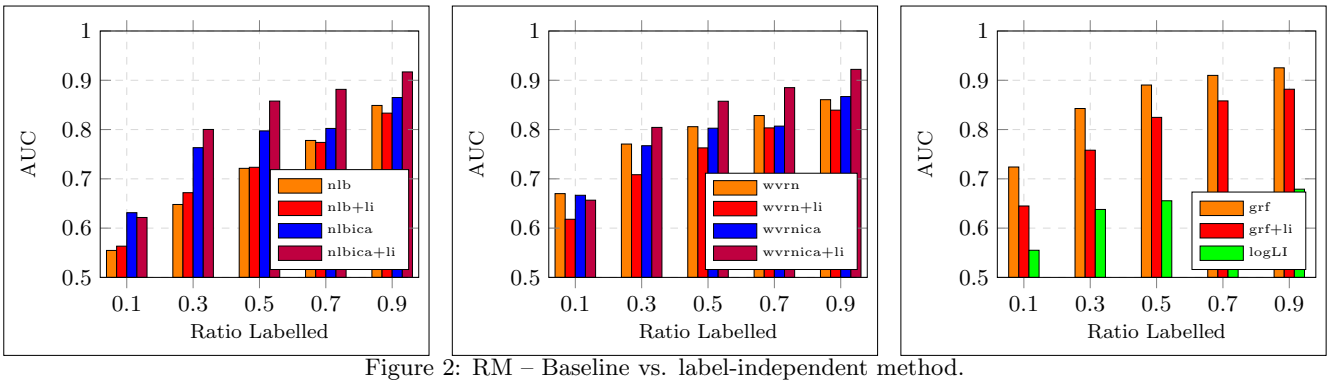

Figure 2: RM – Baseline vs. label-independent method.

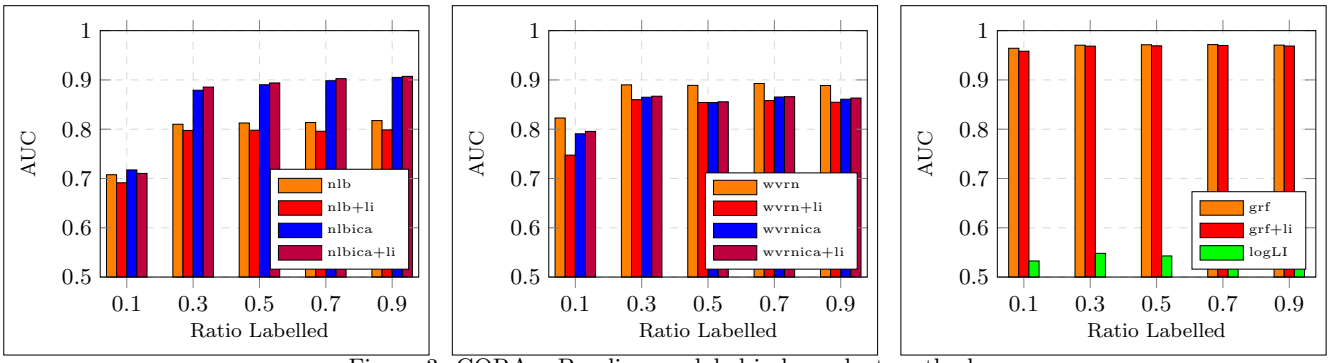

Figure 3: CORA – Baseline vs. label-independent method.

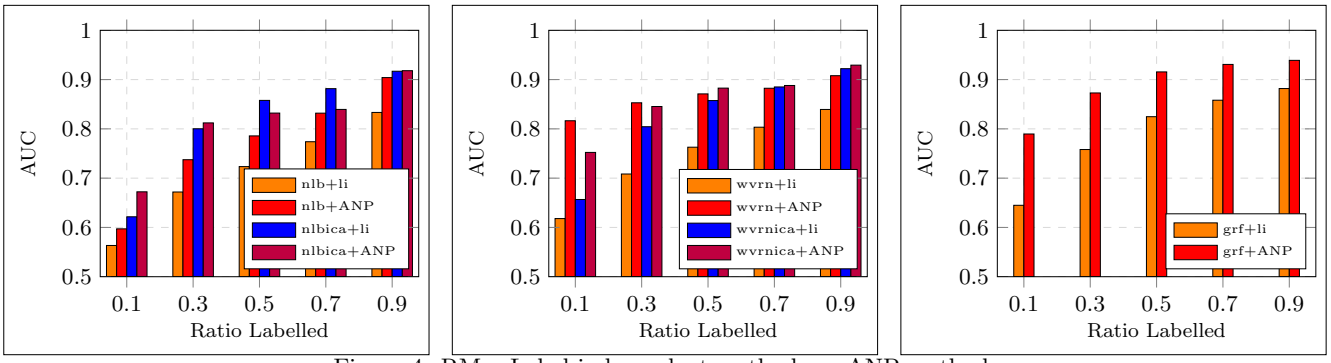

Figure 4: RM – Label-independent method vs. ANP method.

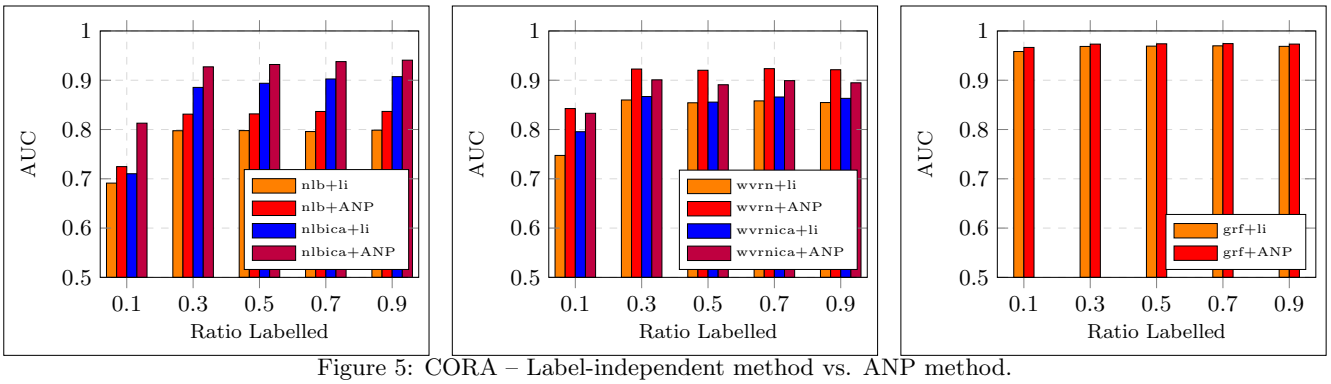

Figure 5: CORA – Label-independent method vs. ANP method.

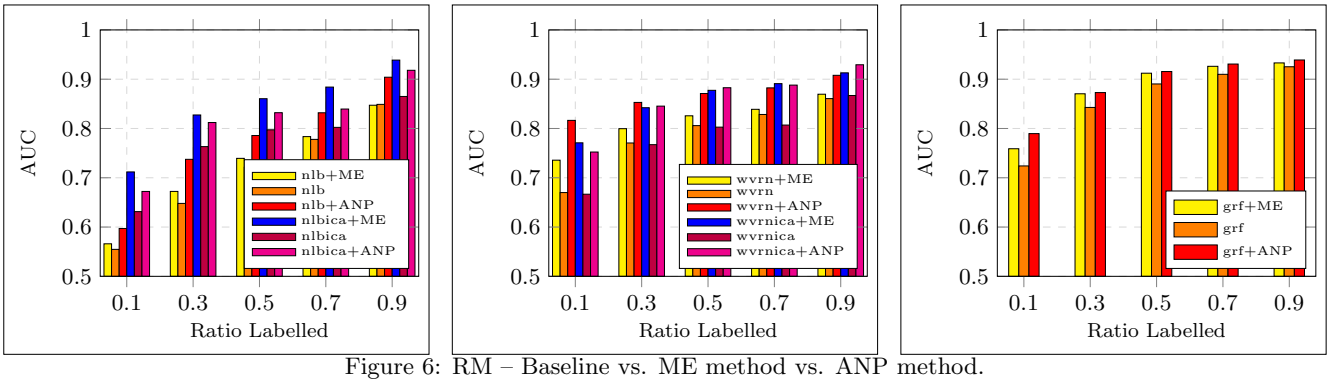

Figure 6: RM – Baseline vs. ME method vs. ANP method.

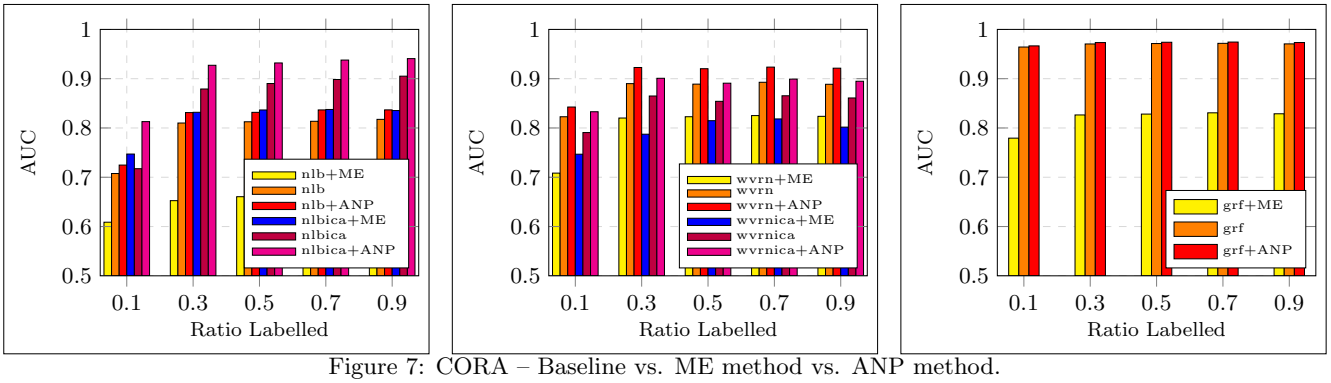

Figure 7: CORA – Baseline vs. ME method vs. ANP method.

features or attributes inherent in the data.

Figures 2 and 3 also show the effects of adding label-independent features to classification methods. In general, the label-independent features do not seem affect the AUCs in an easily predictable way. For 26 out of the 30 different ratios of labeled-to-unlabeled data, the performance of nLB, wvRN and GRF *decreases* when using the LI method, and it is apparent that label-sparsity has a strong negative impact on both baselines and LI-methods. In addition, the logLI logistic regression classifier, run by itself, shows poor performance compared to all other baselines. This could explain why there are so few examples where LI-methods outperform the baseline. The only cases where the LI approach improved AUC are when collective classification (ICA) is used for trials with ratio ≥ 0.3, but even then the improvement of AUC is negligible, $\leq 1\%$, for the CORA dataset.

Results for our proposed method of Attribute Network Propagation are shown in Figures 4 and 5. We compare the results from the label-independent methods against the ANP method described in Section 2. In 48 out of the 50 comparisons, our proposed method outperforms the label-indpendent method; the benefit is most pronounced for lower ratios of labeled-to-unlabed data (e.g., ratio ≤ 0.5 for the RM dataset), A prominent example of this effect can be seen for wvRN+ANP with ratio $= 0.1$ that outperforms wvRN+li, even when the latter is given access to a much higher proportion of labelled nodes ratio $= 0.7$.

Our results from Figure 4 and 5 show that using edges based on shared attributes may improve the AUC when relational classifiers are used on sparsely-labelled problems, but they are not enough to confirm that the complexity of our method of linearly combining predictions with ANP is justified.

In the last two figures 6 and 7 we have results from ANP compared with results of the less complex method ME. Its easily apparent from the CORA results in Figure 7 that simply adding edges based on shared attributes with ME can lead to a decrease in AUC scores. For all ratios of labelled-to-unlabelled data in the CORA dataset, the mixed approach +ME performs worse than baseline algorithms on the original dataset. In total, ANP performs better than ME in 43 out of 50 cases. Running inference on separate graphs, like with ANP, can in this case be an advantage. This phenomenon may be due to how relational classifiers like wvRN and nLB use neighboring nodes' labels. The added edges method of ME might cause more confusion when calculating the wvRN weighted sum in equation 3 by adding edges to nodes which have opposite labels of the one being classified, whereas with ANP the separation of edge types potentially avoids this problem. However we have not been able to confirm this yet.

5. CONCLUSIONS AND FUTURE WORK

We have proposed a method for within-network classification where attributes generate new graph edges, thus taking advantage of latent linkages otherwise not present. Our method combines several ideas from recent research: (i) create a parallel attribute-graph for each dataset where edges only indicate shared node attributes, and (ii) run inference separately on both graphs using simple relational classifiers, and combine predictions after. We implement and repro-

duce results from a recent paper that proposed a label-independent approach. In experiments on two disparate real-world datasets, we have compared our methods to baselines and the label-independent method under varying label-sparsity conditions. Our experiments showed that: (1) The baseline methods perform poorly when very few labels are available; (2) Label-independent features do not consistently produce accurate predictions and can in some case worsen the performance of the constituent baseline methods; (3) Linearly combining predictions from multiple subgraphs can be superior to simply adding edges to the original graph.

In our current model we do not assign weight or value to individual attributes. However, we have observed *ad hoc* that certain attributes predict classification more than others others; thus we are currently investigating automatic ways of attribute selection and recombination of predictions. The attribute edges used by ANP and ME represent relations based only on homophily, but future work should incorporate co-citation regularity phenomena as well. Furthermore ME could be modified to handle classes of edges, instead of treating original and added edges the same. Finally, it remains to compare the performance of our method to recent complex methods like [16, 18, 17], as well as multi-relational network methods, on these types of binary classification problems.

6. REFERENCES

[1] N. Eagle and A. Pentland. Reality mining: Sensing complex social systems. *Personal and ubiquitous computing*, 10(4):255–268, 2006.

[2] A. Fleming, L. K. McDowell, and Z. Markel. A hidden treasure? Evaluating and extending latent methods for link-based classification. *Proceedings of the 2014 IEEE 15th International Conference on Information Reuse and Integration*, 2014.

[3] B. Gallagher and T. Eliassi-Rad. Leveraging label-independent features for classification in sparsely labeled networks: An empirical study. *Lecture Notes in Computer Science*, 2009.

[4] B. Gallagher, H. Tong, T. Eliassi-Rad, and C. Faloutsos. Using ghost edges for classification in sparsely labeled networks. *Proceedings of the 14th ACM SIGKDD*, 2008.

[5] D. Jensen, J. Neville, and B. Gallagher. Why collective inference improves relational classification. *Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '04*, page 593, 2004.

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

[7] S. Macskassy and F. Provost. A simple relational classifier. 2003.

[8] S. a. Macskassy and F. Provost. Classification in Networked Data : A Toolkit and a Univariate Case Study. *Journal of Machine Learning Research*, 8(December 2004):935–983, 2007.

[9] A. McCallum, K. Nigam, J. Rennie, and K. Seymore. Automating the construction of internet portals with machine learning. *Information Retrieval*, 3, 2000.

[10] L. McDowell and D. Aha. Labels or attributes?: rethinking the neighbors for collective classification in sparsely-labeled networks. *Proceedings of the 22nd ACM CIKM*, 2013.

[11] L. K. McDowell, A. Fleming, and Z. Markel. Evaluating and extending latent methods for link-based classification. *Advances in Intelligent Systems and Computing*, 346:227–256, 2015.

[12] J. Neville and D. Jensen. Iterative classification in relational data. *AAAI-2000 Workshop on Learning Statistical Models from Relational Data*, 2000.

[13] J. Neville and D. Jensen. Leveraging relational autocorrelation with latent group models. *Proceedings - IEEE International Conference on Data Mining, ICDM*, pages 322–329, 2005.

[14] J. Neville and D. Jensen. Relational Dependency Networks. *Journal of Machine Learning Research*, 8:653–692, 2007.

[15] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Gallagher, and T. Eliassi-Rad. Collective Classification in Network Data. *Association for the Advancement of Artificial Intelligence*, 2008.

[16] X. Shi, Y. Li, and P. Yu. Collective prediction with latent graphs. *Proceedings of the 20th ACM CIKM*, 2011.

[17] L. Tang and H. Liu. Relational learning via latent social dimensions. *Proceedings of the 15th ACM International Conference on Knowledge Discovery and Data Mining*, 14(16):817–825, 2009.

[18] L. Tang and H. Liu. Leveraging social media networks for classification. *Data Mining and Knowledge Discovery*, 23(3):447–478, 2011.

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