A MULTI-ARMED BANDIT APPROACH FOR BATCH MODE ACTIVE LEARNING ON INFORMATION NETWORKS

BY

DE LIAO

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2016

Urbana, Illinois

Adviser:

Professor Jiawei Han

Abstract

We propose an adaptive batch mode active learning algorithm, `MABAL` (Multi-Armed Bandit for Active Learning), for classification on heterogeneous information networks. Observing the parallels between active learning and multi-armed bandit (MAB), we base `MABAL` on an existing combinatorial MAB algorithm to combine simple strategies to generate query batches. `MABAL` employs a novel error expectation measure for network classification that does not assume assortativity as MAB reward feedback to determine the most fit strategy for the given task. We provide a preliminary optimality analysis of `MABAL` based on performance bounds for combinatorial MAB. A case study illustrates that `MABAL` not only converges quickly to the optimal strategy but also provides insight into the functional roles of the different node types. E-valuations of `MABAL` on real world network classification tasks demonstrate that it achieves performance gains over existing methods independent of the underlying classification model.

To my parents, for their love and support.

Acknowledgments

First and foremost, I would like to express my sincere appreciation to my adviser, Professor Jiawei Han, for his great support and inspiring advice on both my research and career.

I would like to thank Doris Xin for her great support and contribution of this work. She lead this project and completed most of the algorithm details and paper writing. I would like to thank other colleagues, Brandon Norick, Jingbo Shang, Jialu Liu, for their advice and helpful discussion on my research topic.

I am also very grateful to all the other faculties and colleagues in DAIS (Data and Information System) research group at University of Illinois, Urbana-Champaign. I cherish the opportunity to learn from them and gain deeper insights into the field of data mining.

This research was sponsored in part by the U.S. Army Research Lab. under Cooperative Agreement No. W911NF-09-2-0053 (NSCTA), National Science Foundation IIS-1017362, IIS-1320617, and IIS-1354329, HDTRA1-10-1-0120, and grant 1U54GM114838 awarded by NIGMS through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative (www.bd2k.nih.gov). The views and conclusions contained in this document are those of the author(s) and should not be interpreted as representing the official policies of the U.S. Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

Table of Contents

Chapter 1

Introduction

The indisputable ubiquity of networks engendered an avalanche of research activities in network analysis in recent years. The broadest definition describes a *network* as a collection of entities interconnected by *links*, which makes networks an apt description for many systems, e.g. the Internet, academic collaborations, gene regulations. The term *information networks* emphasizes the fact that networks are a constructed concept for organizing information, and that links are channels for information passage. Since information networks are often represented as graphs, we refer to the entities in the network as *nodes.* One common task performed over information networks is classifying constituent nodes by a function learned from existing node labels. For example, molecular biologists are interested in classifying proteins by their functions. Only a fraction of all known proteins have function labels, but proteins form networks based on interactions and co-occurrence. Network analysis can be used for function classification in lieu of the much more expensive wet lab experiments.

In the protein function example, costly and time consuming experimentation is required to determine the function label of a protein. The field of active learning addresses the label acquisition cost problem in classification tasks with such constraints. Instead of passively learning from a given set of examples that is supposed to be representative of the underlying distribution, the active learner examines a large set of unlabeled data and selectively query for the labels of the most informative examples. Active learning thrives on the assumption that the unlabeled data contains patterns and information that can be taken advantage of to study the training objective without labels. Networks, with their rich structures, present an excellent opportunity for such kind of studies, although active learning on networks has not been a research focus until recently. Active learning on information networks has the potential to drive the direction of future scientific research under the emergent concept of *data driven science*, due to the ubiquity and prominence of information networks in many scientific disciplines. In the protein function example above, the important proteins selected by the active learner can serve as guidance on the subject of new experimental functional studies.

Active learning on information networks requires many special considerations that are not applicable to the setting that assumes data independence, i.i.d. for short. Strategies that are effective for i.i.d., such as uncertainty sampling, fall short for networks, as we need to reason about not only the individual examples but also how they affect their neighborhoods and the rest of the network. The node with the highest uncertainty may not gain much for the rest of the network. In i.i.d. queries affect the classifier separating the entire feature space, whereas in networks, queries usually only provide information on a piece of the network we are classifying. When considering the collective gain of a query set, the i.i.d. assumption often allows it to be modeled as a linear combination of individual gains. Such linearity cannot be assumed for networks due to data dependency. Another major challenge to surmount is identifying the structure that connects nodes in the same class. As pointed out in [1], many important network classification tasks do not follow the *assortativity* assumption, which states that nodes in the same class are more densely connected.

Researchers in network analysis continuously seek sophisticated tools and methods for analyzing complex networks. One such methodology that rose to prominence recently is *heterogeneous information networks*, which enriches prior methods for classification, clustering, and ranking on networks with considerations for relation semantics between different types of nodes. Among the various methodologies for analyzing complex networks, the *heterogeneous information networks* formalism provided us the necessary machinery to solve many of the above challenges. A *heterogeneous information network* (HIN) is a network comprised of multiple *types* of nodes connected via links indicating semantic relations [2]. An archetypal HIN is the bibliographic network containing nodes of types *paper*, *author*, *venue*, and *terms*, with links connecting a paper to its authors, the venue that it was published in, and the terms it contains. Previous studies on networked active learning typically focused on networks containing a single type of nodes [1, 3], e.g., the co-authorship graph linking researchers via publication collaborations. In these *homogeneous* networks, strong assumptions about network structures are embedded in the way the links are constructed, limiting the range of analyses that can be performed over the network. The HIN setting allows us to explore the relations that are pertinent to a given classification task, which is advantageous when the connectivity pattern between classes of nodes is unknown. To the best of our knowledge, [4] is the only existing study on HIN active learning. Unlike in [4], our algorithm does not assume assortativity nor dependency on any particular classification model.

We propose an effective and flexible active learning strategy on HINs inspired by the *multi-armed bandit* problem, `MABAL`. We consider the *batch mode* active learning setting in which the learner receives labels for multiple queries at each iteration. `MABAL` employs a combinatorial MAB algorithm to construct query batches based on the suggestion of simple strategies, which are created from the centrality rankings of different types of nodes. Each simple strategy represents a hypothesis about the type of structure that makes a node an informative query. `MABAL` borrows from MAB to handle the tradeoff between exploration of simple strategies and the exploitation of current best strategies, and we provide a preliminary study on its performance bounds based on known MAB results. We evaluate `MABAL` on a number of classification tasks over real world information networks against simple heuristic and literature baselines. We study the adaptability of `MABAL` to different classification tasks over the same network structure as well as its compatibility with different classification models. A case study demonstrates that `MABAL` also offers insight into network structures that are crucial for a given task.

Chapter 2

Related Work

As we recognize the ubiquity of networks, studies centered around network analysis have become increasingly popular. One important type of analyses on networks is *collective classification*, which accounts for data dependencies when classifying objects in a network (see [5] for an in-depth introduction). The body of work on collective classification divides into two schools of thoughts, one that relies on the collective power of local conditional classifiers, and one that views it as a global objective optimization problem, with numerous algorithms in both categories . In our work, we employed two specific collective classification methods, Label Propagation[6], which embodies the second category, and RankClass[7], which is a hybrid of both.

While active learning has a deep rooted history in machine learning research, with early work dating back to the 90's (see [8] for a comprehensive survey), active learning on networks has not been a research focus until recent years. The expected error reduction framework for active learning employed in this work was first proposed in [9] for text classification. [10] presents an adaptation of this framework to graphs, as we have done in Section 4.4. However, their formulation fundamentally relies on the assumption that nodes with the same labels are in close proximity to each other, a limitation excluded from our framework to accommodate a wider range of classification tasks.

Previous work on network active learning without the assortativity assumption include [1, 11]. [1] uses information-theoretic techniques to choose which nodes to explore, and makes no initial assumptions about how the groups connect. [11] is able to effectively explore both similarity and dissimilarity simultaneously. Unlike in our work, they do not have a mechanism to make use of the node type information in the heterogeneous setting. Additionally, [11] requires a pairwise similarity matrix as input, which may not be available for some problem settings. This makes it inapplicable to HINs since it is hardly sensible to compare nodes of different types. [1] runs in exponential time for some settings, while our algorithm will always run in linear time (excluding the computation of centrality). The mutual information query selection criterion in [1] is similar in spirit to our centrality-based primary strategies. [3] presents a network active learning framework that employs both a local classifier based on node attributes and a collective classifier to account for data dependencies. It relies on clustering on node attributes to avoid sampling bias, which is ineffective when node attributes are very sparse or non-existent. In comparison, we avoid bias by directly using the observed label distributions in the primary strategies.

To the best of our knowledge, [4] is the only other work that has studied active learning on HINs. In their study, a combination of clustering using metapaths and uncertainty sampling is used for query selection. We have learned in our studies that finding clusters that correlate well with class labels in an HIN is very sensitive to the relations considered, which is why we opted for an expected error reduction scheme instead. Instead of depending on user guidance in the form of metapaths for performance, we provide to the users information about the network structures that are crucial to their task. Our algorithm not only provides a performance boost over their method but also insights into the functional roles of the different types of nodes in the network.

As a resource allocation model, multi-armed bandit lends itself naturally to the active learning problem. Prior to our work, [12] and [13] have independently drawn up analogies between active learning and multi-armed bandit. Both work explore fully sequential learning, i.e., a single query is made at every iteration. As seen in our work, the correspondence between AL and MAB becomes much more complex when we consider batch mode learning. To this end, we transform the batch mode active learning problem in order to apply a combinatorial MAB algorithm with a proven regret bound[14]. This allows us to reason about the optimality of our algorithm, which is not commonly done in the active learning literature.

Chapter 3

Preliminaries

3.1 Information Networks

We represent an information network as a graph $G = (V, E)$ with V being the set of n nodes corresponding to the entities in the network and $E = \{(n_1, n_2, w_{12}) | n_1, n_2 \in V\}$ the set of m links between these entities, with w_{12} denoting the edge weight reflecting the strength of the tie. $w_{ij} = 1 \forall (i, j) \in E$ in an *unweighted* graph and is therefore omitted. G is often represented by an adjacency matrix A. An element $a_{ij} \in A$ equals the edge weight w_{ij} if the edge (i, j) exists and 0 otherwise. Each node $n \in V$ is associated with a feature vector $\mathbf{x}_n \in \mathcal{X}$, the feature space, where $dim(\mathcal{X}) \geq 0$. In the simplest case where $dim(\mathcal{X}) = 1$, the feature vectors are composed of a single unique identifier for each node.

In a *heterogeneous information network*, each node $v \in V$ is mapped onto a specific *type* $t \in T$ via $\tau : V \rightarrow T$, the set of all types. Let $V_t \subseteq V$ denote the set of nodes with type t. These subsets do not overlap, i.e., $V_t \cap V_{t'} = \emptyset$ $\forall\ t \neq t' \in V$ and $\bigcup_{t \in T} V_t = V$. The ordered pair $r = (t_i, t_j), t_i, t_j \in T$ defines a *relation* over G. Let R be the set of all relations over G. The function $\phi : E \rightarrow R$ maps an edge $e_{ij} = (v_i, v_j)$ onto a relation $r = (t_k, t_l)$ if and only if $\tau(v_i) = t_k$ and $\tau(v_j) = t_l$. For example, the bibliographic network introduced earlier with $T = \{$*paper, author, venue, term*$\}$ contains the relations $\{$*(paper, author), (paper, venue), (paper, term)*$\}$.

3.2 Collective Classification

Given a large information network, a common task is to infer the class label of all constituent nodes based on the labels provided for a subset. This problem is commonly referred to as *collection classification* in the literature[5]. For a specific classification task, let $\mathcal{Y}$ denote the set of class labels. We want to train a classifier f^* to approximate the hidden objective $f : V \to \mathcal{Y}$ that generated the node labels, given the observations $\mathcal{L} = \{(v, y_v) | v \in V, y_v \in \mathcal{Y}\}$. For notational convenience, we use $\mathcal{L}$ to denote both the set of (node, label) pairs and the set of labeled nodes in V. A trained classifier can either predict a single label for an input node v or a probability distribution over all labels, $P(y|v), y \in \mathcal{Y}$. We assume the latter in this work.

3.3 Graph Centrality Measures

Graph centrality measures, also known as centrality indices, provide a means to quantify the importance of nodes in a network based on some definition of *importance*. Many centrality measures, such as degree, PageRank[15], and closeness, were defined over the history of network analysis to serve different applications, e.g., search engines, community detection in social networks. The essence of a centrality measure is a function, $C : V \to \mathbb{R}$, that maps nodes in G onto real values that allow nodes to be ranked by their importance. More formally, the centrality measure C induces a total order on V under $\leq_C$ such that for a pair of nodes $v, v' \in V$, $v \leq_C v' \Leftrightarrow C(v) \leq C(v')$. For a node v, its *rank* in V under C is defined as $rank_{C|V}(v) = \sum_{v' \in V} I(v \leq_C v')$, i.e., the number of nodes with centrality less than that of v. In other words, the most important nodes based on C have the lowest $rank_{C|V}$ values, which place them at the head of the sequence listing the nodes by descending centrality.

In this work, we consider the following commonly used centrality measures: degree, closeness, betweenness, PageRank, eigenvector and Katz. We refer interested readers to [16] for an in-depth discussion on centrality measures and their formal definitions. On HINs, previous studies suggest that higher order network structures can be captured using centrality measures considering relation semantics, such as *metapaths* consisting of a sequence of relations. We defer the investigation of such novel complex centrality measures to future studies.

The definition of a centrality measure describes a structural function of nodes in the network, and the centrality value for a given node quantifies its suitability for serving that function. Two measures are very unlikely

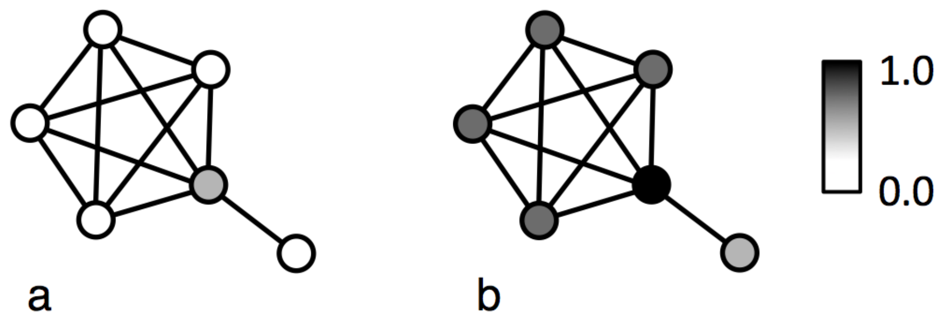

Figure 3.1: a) Nodes in a lollipop graph colored by betweenness. b) Nodes in a lollipop graph colored by closeness.

to produce the same ordering on V. Take *betweenness* and *closeness* for example. By definition, a node v has high *betweenness* if it acts as a *bridge* in the graph, i.e., the shortest path between any pair of nodes in G tends to go through v. *Closeness*, on the other hand, measures the average distance between a node to all other nodes in V. Figure 3.1 shows a graph with different node rankings under *betweenness* and *closeness*. Thus, different centrality measures present different hypotheses about what makes a node important in the network.

3.4 Batch Mode Active Learning

Active learning can be conducted in many different modes based on the constraints of the specific application. [8] provides a comprehensive survey of active learning scenarios. In this work we focus on *batch mode* learning in which the learner starts with a set of unlabeled instance, $\mathcal{U}$, and issues *queries*, $Q \subseteq \mathcal{U}$, to the *oracle* $\mathcal{O}$, such as a human annotator. When applied to learning on information networks, $\mathcal{U} \subseteq V$. In this study, we assume that $\mathcal{O}$ provides a single label $y_q \in \mathcal{Y} \cup \emptyset$ ($\emptyset$ when the label is not available) for each query q and is stable, i.e., it always provides the same label for q whenever queried. This assumption implies zero utility in re-querying the same instance, which allows us to safely remove an instance from $\mathcal{U}$ once it has been queried.

As labeling cost is prohibitive in tasks targeted by active learning, batch mode active learning is parameterized by the *budget* B, the upper bound on the total number of queries, and the batch size b, the number of queries to issue at each iteration over $\lceil B/b \rceil$ iterations [8]. The learner can choose to exhaust the budget in a single iteration in what is referred to as *one-shot* learning, or issue a single query over $\leq B$ iterations in fully sequential learning. With the labels received from $\mathcal{O}$, the learner updates $\mathcal{U} = \mathcal{U} \setminus Q$, $\mathcal{L} = \mathcal{L} \cup Q$ and then its strategy for selecting future queries based on $\mathcal{L}$. Since the compositions of $\mathcal{U}$ and $\mathcal{L}$ are time dependent, we use $\mathcal{U}_i$ and $\mathcal{L}_i$ to denote the two sets after the update at time i for clarity. Batch mode allows the learner to better adapt to the classification task from incremental changes in the observed label distribution. Strategy update often involves retraining the classification model on $\mathcal{L}$. We have *one-shot* learning when $b = B$ and *fully sequential* learning when $b = 1$. We assume uniform labeling cost over all instances in P. Works such as [17, 18, 19] address queries with varying costs.

3.5 Multi-armed Bandit

The multi-armed bandit (MAB) problem models the *exploration* v. *exploitation* tradeoff in sequential allocation tasks [20]. In the classic setting, at each iteration i a player makes a play p_i, by pulling one of the K arms on a *bandit*, or slot machine, and received a reward $r(p_i)$. The objective is to maximize the cumulative reward $R = \sum_{i=1}^{T} r(p_t)$ earned over T rounds. This is often modeled as minimizing a player's *regret*, the difference between R and R^{OPT} achieved by the optimal strategy. Each arm has a reward distribution, modeled as a random variable X_i with $E[X_i] = \mu_i$ unknown to the player, who has to make decisions based on empirical rewards from past actions in each round. Let $\bar{\mu}_k^t$ be the player's expectation of reward from arm k in round t. The player can either choose to *exploit* his knowledge about the payoffs by playing $\text{argmax}_{k=\{1,\ldots,K\}} \bar{\mu}_k^t$ or to *explore* arms whose reward distribution he is less certain about, where the definition of *certainty* varies based on the player's belief about the reward distribution. Exploration often entails playing a currently suboptimal strategy in the hope of maximizing cumulative gain over the remaining rounds.

In *combinatorial multi-armed bandit* (CMAB), the player at each round plays multiple arms, or a *super arm* $S \in 2^{[K]}$, and receives feedback for S. There are three types of feedback in CMAB for playing S: 1) *full information:*

a reward is observed for all arms in the bandit; 2) *semi-bandit*: a reward is observed for individual arms in S; 3) *bandit*: a single reward value is observed for S[21]. In this work we make use of the algorithm proposed in [14], which assumes *semi-bandit* feedback and achieves a regret of $O(\log(T))$.

Chapter 4

The Algorithm

In this section we formally describe `MABAL`, our proposed active learning algorithm on information networks. `MABAL` combines *primary* learners using an existing CMAB algorithm, CUCB [14], to compute expected rewards of the primary learners. We first define and motivate our choice for *primary* learners in Section 4.1 and then establish a straightforward correspondence between components of MAB and AL in Section 4.2 to enable the application of CUCB for combining *primary* learners. CUCB requires a function for computing super arm rewards and a method for creating *super arms* given reward estimates. We describe and motivate our choices for these two components in our application in Section 4.4 and 4.5. At a high level, CUCB estimates expected rewards based on the empirical rewards and the number of times an arm is explored. It boosts the reward expectations for arms that have not been explored much in order to prevent us from dismissing a potentially optimal strategy without much evidence.

4.1 Centrality by Type as Primary Learners

Intuition: Suppose we are given an HIN constructed from customer reviews for businesses as in Figure 4.1a). Consider two potential classification tasks on this example HIN: **1.** classifying businesses by customer satisfaction, as in Figure 4.1b); **2.** classifying businesses by geographic location, as in Figure 4.1c). Clearly, the term nodes “terrific” and “terrible” are central to the *satisfactory* and *unsatisfactory* classes respectively for the first task, whereas the customer nodes “Andy” and “April” are the key connections between all businesses in *Town E*. An effective active learning algorithm should be able to prioritize salient term nodes for the first task and key users in the second. It not only needs to identify the node types that are the most pertinent to

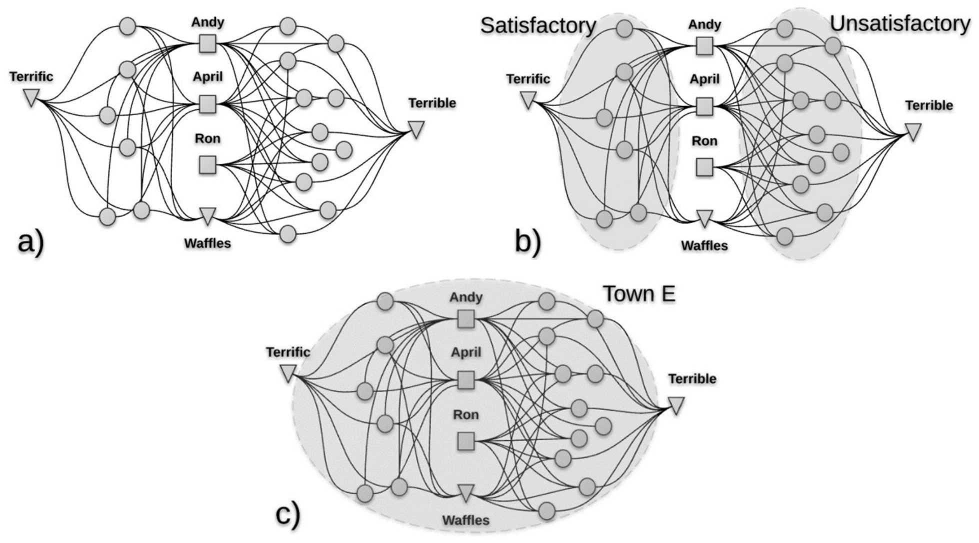

Figure 4.1: a) HIN with *customers* (□), *businesses* (◯), and *keywords* (▽). b) Businesses classified by customer satisfaction. c) Businesses classified by geographic location.

the classification task but also important nodes within the identified types. Since the two tasks are performed on the exact same network structure, the learner needs to adapt its query strategy to the task at hand once it has seen some labels.

Based on the observations above, we create simple query strategies from centrality rankings for each type of nodes. A *primary learner*, λ_t^C is constructed from the ordering induced on V_t by the centrality measure C. We denote the unlabeled set for type t as $\mathcal{U}_t = \mathcal{U} \cap V_t$. When queried in batch mode with batch size b, λ_t^C will return $Q^{\lambda_t^C} = \{v | v \in \mathcal{U}_t, rank_{C|\mathcal{U}_t}(v) \leq b\}$, i.e., the top b unlabeled nodes of type t with the highest centrality C. Note that λ_t^C's with the same t share the same candidate pool $\mathcal{U}_t$ but prioritizes nodes in the pool differently for querying. $\mathcal{C}$, the set of centrality measures used in `MABAL`, can be any arbitrary combination of existing or novel centrality measures.

Let Λ be the set of primary learners in `MABAL`. To account for the possibility that none of the centrality based primary learners serve as adequate active learning strategies, we add to Λ *Random*, a primary learner constructed from a random ordering of the nodes, equivalent to a *passive* learner. Including *Random* in Λ also serves as a mechanism to regulate the active learner from overfitting to the centrality-based strategies. This prevents `MABAL` from doing worse than the passive learner when none of the centrality-based strategies prove to be effective. However, increasing the number of learners can slow the convergence of `MABAL`, which is dependent on $|\Lambda|$. We explore the effect of adding *Random* in Section 5.3.1.

4.2 Correspondence between AL and MAB

We define the *utility* of a query q given the labeled set $\mathcal{L}$ as

$$u(q|\mathcal{L}) = acc(f^*|\mathcal{L} \cup q) - acc(f^*|\mathcal{L}) \tag{4.1}$$

where $f^*|\mathbf{X}$ is the classifier f^* trained on the observations $\mathbf{X}$, and $acc(f^*) = \sum_{v \in V} I(f^*(v) == y_v)/|V|$ is the classification accuracy of f^* on V, where y_v is the true label for v. The utility of a $\mathcal{L}$ is simply

$$u(\mathcal{L}) = acc(f^*|\mathcal{L}) \tag{4.2}$$

Given the label budget B, the objective of the active learner is to find

$$\mathcal{L}^{OPT} = \operatorname*{argmax}_{\mathcal{L} \in \mathcal{P}_B(V)} u(\mathcal{L}) \tag{4.3}$$

Suppose for the time being that $u(q|\mathcal{L})$ can be acquired after q is labeled by $\mathcal{O}$. This allows us to establish a natural correspondence between the objectives of active learning and multi-armed bandit in the following fashion. Each arm in the bandit corresponds to a primary learner $\lambda \in \Lambda$. Let's first consider the fully sequential setting, i.e., $b = 1$. At each iteration i, some $\lambda \in \Lambda$ is picked to issue the query q_i, which corresponds to a play p_i in MAB. The label budget B corresponds to T, the number of rounds played in MAB. With $(q_1, \dots, q_B)$ being the sequence of queries such that $\mathcal{L} = \{q_1, \dots, q_B\}$, $acc(f^*|\mathcal{L})$ corresponds to the cumulative reward R in MAB earned by the sequence of plays $(p_1, \dots, p_T)$. Thus, finding the sequence of optimal plays that maximizes R is equivalent to (4.3). $\square$

The simplest adaptation for $b > 1$ is to choose a single primary learner $\lambda \in \Lambda$ as done above and use Q^λ as the query set Q. However, this strategy can severely limit the diversity of Q and thus reduce $\mathcal{L}$'s coverage of G and $\mathcal{Y}$. We cannot adequately learn a classifier over G if there is a lack of examples for some $y \in \mathcal{Y}$ or major components of G. We propose `Batch`, a function that computes Q by selectively taking *advice* from the primary learners based on their expected rewards $\{\bar{\mu}(\lambda)\}$. In brief, `Batch` combines the *advice* Q^{λ} from λ weighted by its expected reward $\bar{\mu}(\lambda)$ to form Q in a way that also promotes diversity. We describe `Batch` in detail in Section 4.5.

In establishing the correspondence between AL and MAB we assumed knowledge of the utility of query q, $u(q|\mathcal{L})$, which can be used as feedback for $\bar{\mu}(\lambda)$ to directly optimize for (4.3). However, $acc(f^*)$ cannot be computed without the ground truth labels for all nodes, which are not available to the learner. Additionally, we need to define $u(Q|\mathcal{L})$ for the query set Q in batch mode. While it is tempting to think of $u(Q|\mathcal{L})$ as the sum of $u(q|\mathcal{L})$ for $q \in Q$, u is not additive due to data dependencies in a network. Consider a Q made up of two nodes v_1, v_2 connected by an edge. While v_1 and v_2 may have high utility individually, $u(Q)$ is not a sum of their utilities due to the coverage overlap between two adjacent nodes. We propose a novel network-based expected error reduction measure, ∇, as a proxy for u to address these issues. We define ∇ and $\hat{\mu}(\lambda)$, the *empirical* mean reward of λ, in Section 4.4.

Algorithm 1 MABAL

```
1: procedure MABAL($G = (V, E)$, $B$, $b$, $\Lambda$)
2:     $i \leftarrow 0$, $\mathcal{L} \leftarrow \emptyset$, $\mathcal{U} \leftarrow V$
3:     $T_\lambda \leftarrow 0 \quad \forall \lambda \in \Lambda$                ▷ Num. of queries from $\lambda$
4:     $\hat{\mu}(\lambda) \leftarrow 1 \quad \forall \lambda \in \Lambda$               ▷ *Empirical* reward for $\lambda$
5:     while $i \cdot b < B$ do
6:         $b' = \min(b, B - t \cdot b)$
7:         for all $\lambda \in \Lambda$ do
8:             $\bar{\mu}(\lambda) = \hat{\mu}(\lambda) + \sqrt{\frac{3 \ln i}{2T_\lambda}}$
9:         $Q = \text{Batch}(G, b', \Lambda, \mathcal{U}, \{\bar{\mu}(\lambda^t_C)\})$
10:        $\mathcal{L} \leftarrow \mathcal{L} \cup \mathcal{O}(Q)$
11:        $\mathcal{U} \leftarrow \mathcal{U} \setminus Q$
12:        $T_\lambda \leftarrow T_\lambda + |Q \cap Q^\lambda| \quad \forall \lambda \in \Lambda$
13:        Train classifier on $\mathcal{L}$
14:        update $\hat{\mu}(\lambda) \quad \forall \lambda \in \Lambda$ using (4.8)
15:        $i \leftarrow i + 1$
16: return $\mathcal{L}$
```

We present `MABAL`, our proposed algorithm for active learning on networks, in Algorithm 1. `MABAL` takes an information network represented by G, the label budget B, the batch size b, and the set of primary learners Λ as input. $\hat{\mu}(\lambda)$ is the *empirical* reward of learner $\lambda \in \Lambda$, and T_λ is the number of nodes nominated by λ that were labeled, analogous to the number of a times an arm is played in MAB. In line 8, we use CUCB to compute $\bar{\mu}(\lambda)$, the *expected* reward of λ, which is then used to compute the query set Q by `Batch`. We then query the oracle $\mathcal{O}$ for the labels of nodes in Q and update $\mathcal{L}$, $\mathcal{U}$ and T_λ's accordingly. To update the *empirical* rewards of the λ's, we retrain the classifier on $\mathcal{L}$ and recompute $\hat{\mu}(\lambda)$ using (4.8). `MABAL` assumes no seed nodes at the beginning of the algorithm, although it can be easily adapted to make use of seeds.

4.3 Connection Patterns via Edge Weights By Relations

We capture the importance of semantic relations by assigning weights to edges based on relations. Let $W_{r_{i,j}} \in [0,1]$ denote the weight for the relation $r_{ij} \in R$ and $W = \{W_{r_{i,j}}\}$, the set of weights over all relations. Let $\omega_W : E \to W$ be the function that maps each edge onto a weight value in W based on its type. Intuitively, the weights indicate the strength of each relation. A specific W produces a *representation* of the network that leads to certain conclusions about its structure.

4.4 Entropy Reduction as MAB Reward

In the *expected error reduction* framework for active learning, queries are selected to minimize the generalization error over $\mathcal{U}$ [8]. Strategies in this framework are known to be computationally expensive, since finding the optimal query requires retraining on all possible queries. In batch mode, this becomes combinatorially more expensive.

For a given query q, the expected log-loss over $\mathcal{U}$ with θ trained on $\mathcal{L} \cup q$ is

$$\sum_{y\in\mathcal{Y}} P(y|q) \left(-\sum_{v\in\mathcal{U}} \sum_{y\in\mathcal{Y}} P_\theta(y|v) \log P_\theta(y|v) \right) \tag{4.4}$$

Since $H(\mathcal{Y}|v) = -\sum_{y\in\mathcal{Y}} P(y|v) \log(P(y|v))$ is the entropy in the label distribution of v, the above is equivalent to finding a query that minimizes the expected entropy over $\mathcal{U}$. In our setting where each query receives a single label from $\mathcal{O}$, we can simply drop the factor for marginalizing q over $\mathcal{Y}$. That is to say, (4.4) reduces to $\sum_{v \in \mathcal{U}} H(\mathcal{Y}|v)$ when the oracle provides single labels for queries.

Based on the expected log-loss, we propose a graph specific error reduction measure to distinguish queries by their ability to reduce generalization errors from the results of a single training. The r-order neighborhood of a node v is defined as $N_r(v) = \{v' | d(v, v') \leq r\}$, where $d(v, v')$ is the distance between v and v'. $v \in N_r(v)$ since $d(v, v) = 0$. At iteration i, let θ_i be the model trained on $\mathcal{L}$, which includes labels received from the oracle up to i. We define $\nabla_i(q)$, the local error reduction due to query q at time i as

$$\nabla_i(q) = \sum_{v \in N_r(q)} H_{\theta_{i-1}}(\mathcal{Y}|v) - H_{\theta_i}(\mathcal{Y}|v) \tag{4.5}$$

For a query set Q, we have

$$\nabla_i(Q) = \sum_{v \in N_r(Q)} H_{\theta_{i-1}}(\mathcal{Y}|v) - H_{\theta_i}(\mathcal{Y}|v) \tag{4.6}$$

where $N_r(Q) = \bigcup_{q \in Q} N_r(q)$. $\nabla_i(Q) \neq \sum_{q \in Q} \nabla_i(q)$ when there are overlaps in the neighborhoods of nodes in Q. We use (4.5) to approximate $u(q|\mathcal{L})$, the reward for the query q, and (4.6) to approximate $u(Q|\mathcal{L})$, the reward for the query set Q.

The reward of playing λ at time i is thus defined as

$$\nabla_i(\lambda) = \exp\left(\frac{\nabla_i(Q^\lambda)}{|\nabla_i(V)|} - 1\right) \tag{4.7}$$

To avoid a large reduction in a single round from biasing the algorithm towards any particular λ for the remainder of the query budget, we normalize $\nabla_i(Q^\lambda)$ by $|\nabla_i(V)|$, the absolute value of total entropy reduction incurred in the ith iteration. We use the absolute value of the global reduction for normalization to avoid a false positive reward signal as an artifact of global entropy increase. We transform the ratio via $e^{(x-1)}$ so the reward for any λ is always positive and mostly in the $[0, 1]$ range. We allow $\nabla_i(\lambda) > 1$ since it does not interfere with CUCB or our convergence analysis later on. The *empirical* reward mean for λ at time i can be computed as

$$\hat{\mu}_i(\lambda) = \frac{(i-1) \cdot \hat{\mu}_{i-1}(\lambda) + \nabla_i(\lambda)}{i} \tag{4.8}$$

4.5 Query Batch Selection

Algorithm 2 Query batch selection

```
1: procedure BATCH($G$, $b$, $\Lambda$, $\mathcal{U}$, $\{\bar{\mu}_i(\lambda)\}$)
2:     $Q^* \leftarrow \bigcup_{\lambda \in \Lambda} Q^\lambda$
3:     $s_t \leftarrow 1 - \frac{b}{|V_{\tau(q)} \cap \mathcal{U}|+1} \quad \forall t \in T$
4:     $\mu'(\lambda) \leftarrow s_{\tau(\lambda)} \cdot H(\mathcal{Y}|\lambda) \cdot \bar{\mu}(\lambda) \quad \forall \lambda \in \Lambda$
5:     for all $q \in Q^*$ do
6:         $\bar{\mu}(q) \leftarrow \sum_{\lambda \in \Lambda(q)} \mu'(\lambda) \cdot v_\lambda(q)$
7:     $S = sort_{desc}(Q^*, \{\bar{\mu}(q)\})$
8: return $S[: b]$
```

Let $MAB_{\mathcal{C}}$ be the *super learner* comprised of an ensemble of primary learners $\{learner^t_C | t \in T, C \in \mathcal{C}\}$. At each iteration i, $MAB_{\mathcal{C}}$ solicits *advice* from the primary learner by asking each of them to submit their top k candidates, $Q^{(i)}_{tC}$. Clearly, $|Q^i| > k$ since $\{P_t\}$ partitions V, where $Q^{*i} = \bigcup_{t \in T, C \in \mathcal{C}} Q^i_{Ct}$. $MAB_{\mathcal{C}}$ needs to determine the top k most informative queries in Q^{*i} to form the final query set $Q^i, |Q^i| = k$, submitted to $\mathcal{O}$. We make a simple transformation on $MAB_{\mathcal{C}}$ in order to use existing CMAB algorithms to find Q_i. For each primary learner $learner^t_C$, we create k arms, $learner^{t(1)}_C, \ldots, learner^{t(k)}_C$, for each of the k candidate slots. Note that multiple arms could represent the same node at time i if centrality rankings coincide at that node. Finding the optimal Q_i now boils down to the problem of finding an optimal super arm whose union contains k distinct nodes. [14] provides an algorithm, CUCB, with theoretical guarantees that makes very little assumption on the reward structure and the process by which super arms are picked.

We present `Batch` in Algorithm 2, the subroutine in `MABAL` for selecting the optimal query batch once the expected rewards for the primary learners are computed. Recall that the objective of AL is to select queries that result in the largest reduction in classification error. As previously observed in

Section 4.2, an optimal query set should yield good coverage both in terms of the network and the classes in $\mathcal{Y}$. `Batch` adjusts the expected rewards of the primary learners to account for these two factors via *type bias correction* s_t and *label diversity* $H(\mathcal{Y}|\lambda)$. Based on the adjusted primary learner expectations, `Batch` computes the expected reward of each query node as the sum of *votes* weighted by the rewards of the primary learners. It then selects the top b nodes with the highest votes as the query batch to be used by `MABAL`. We first present two alternatives for computing the expected query reward $\bar{\mu}(q)$ in line 6 of Algorithm 2 and then formally introduce the two adjustment factors.

4.5.1 Expected Query Reward

The objective of `Batch` is to find a query set Q^{OPT} of size b that yields the highest reward expectation, i.e.,

$$Q^{OPT} = \operatorname*{argmax}_{Q \in \mathcal{P}_b(\mathcal{U})} \mathrm{E}[u(Q|\mathcal{L})] \tag{4.9}$$

Note that we do not need to know the exact value of the true maximum reward, which is uncomputable as explained in Section 4.2, in order to find the optimal set. Therefore, we can instead use (4.6) to search for Q^{OPT}. Computing Q^{OPT} then boils down to finding a query set that would effect the largest entropy reduction in its immediate neighborhood, a quantity that is dependent on the underlying collective classification model. In order to be classification-model-agnostic, `Batch` approximates expected entropy reduction via the expected rewards for the primary learners, which are functions of observed entropy reductions.

For each $q \in Q^* = \bigcup_{\lambda \in \Lambda} Q^{\lambda}$, let $\Lambda(q)$ be the set of primary learners that selected q as a candidate. The expected reward of q under *weighted centrality vote* is defined as:

$$\bar{\mu}_C(q) = \sum_{\lambda_t^C \in \Lambda(q)} \bar{\mu}(\lambda_t^C) C(q) \tag{4.10}$$

The expected reward of q under *weighted Borda count* is:

$$\bar{\mu}_b(q) = \sum_{\lambda_t^C \in \Lambda(q)} \bar{\mu}(\lambda_t^C)(b - rank_{C|Q^{\lambda_t^C}}(q)) \tag{4.11}$$

A primary learner λ_t^C submits $v^C_{\lambda_t^C}(q) = C(q)$ as its *centrality vote* for q. Although $C(v) \in [0,1] \quad \forall v \in V, C \in \mathcal{C}$, the typical range of different centrality measures can be very different based on their definitions. For example, in Figure 3.1, the average betweenness of a node is much lower than the average closeness in the same graph. To counter any negative effects of this artifact, we devised the second vote counting strategy using the *Borda count*, a positional voting system that dates back to 1770 [22]. As seen in (4.11), the number of votes q receives from λ_t^C under *Borda count*, $v^B_{\lambda_t^C}(q)$, is b minus its centrality rank in $Q^{\lambda_t^C}$, which allows for a fair comparison between the learners without being affected by the range difference intrinsic in the centrality definitions. We compare the performance of $v^C(q)$ and $v^B(q)$ in Section 5.3.1. We define the expected reward of Q as a function of $\bar{\mu}(\lambda)$ as follows,

$$\bar{\mu}(Q) = \sum_{q \in Q} \sum_{\lambda \in \Lambda(q)} \bar{\mu}(\lambda) \cdot \bar{\mu}(q) \tag{4.12}$$

(4.12) makes an important assumption about the additivity of query rewards, namely, if a node is in the neighborhood of multiple queries, the effects of all queries on its expected entropy reduction are additive. For specific collective classification models and tasks this may not be the case. This assumption allows us to derive a fast greedy approximation of (4.9), which proved to be effective in empirical evaluations. We reason about the optimality of `Batch` in Section 4.6. And we consider two more factors before making the final selection of k nodes based on ranking, which are discussed in details as follows.

4.5.2 Type Bias Correction

Learners of type t_1 are much more likely to produce overlapping query sets than learners of type t_2 when $|V_{t_1}| \ll |V_{t_2}|$. In this scenario, a type t_1 node will on average receive more votes than a type t_2 node simply because type t_1 learners have a much smaller candidate pool. We correct for *type bias* by discounting nodes with type t by

$$s_t = 1 - \frac{b}{|V_t \cap \mathcal{U}| + 1} \tag{4.13}$$

where $\frac{b}{|V_t \cap \mathcal{U}|}$ is the probability that a node is included in a size b random sample. We add 1 to the denominator for smoothness.

4.5.3 Label Diversity

We enforce label coverage in the query set by promoting learners that have suggested queries with diverse labels. Let $\mathcal{L}^\lambda$ be the set of labeled nodes that were nominated by λ. We compute the label diversity of λ as the entropy in $\mathcal{L}^\lambda$,

$$H(\mathcal{Y}|\lambda) = -\sum_{y \in \mathcal{Y}} P(y|\mathcal{L}^\lambda) \log P(y|\mathcal{L}^\lambda) \tag{4.14}$$

(4.14) achieves the highest value when $\mathcal{L}^\lambda$ contains an equal number of n-odes with each label. This scheme is preferable to distance based coverage enforcement since we do not assume assortativity. Without assortativity, nodes with dissimilar labels are not necessarily far apart, which implies requiring a minimum distance between queries does not lead to guaranteed coverage. Furthermore, $H(\mathcal{Y}|\lambda)$ is much cheaper to compute than a set with a minimum distance.

The Kullback-Leibler divergence can be used instead for datasets with skewed label distributions. If n were included in Q_i, all primary learners in $M^i(n)$ receives a partial reward observation for n. This introduces dependency between the different primary learners.

This formulation presents an interesting opportunity for us to simultaneously optimize for the utility of Q_i and the number of primary learners explored. The intuition is to select nodes that are popular among primary learners with high empirical rewards.

4.6 Optimality Analysis

In MAB, the player is constantly faced with the choice to *exploit*, i.e., make an optimal play based on current reward observations, or to *explore* options with uncertain rewards. The player is incentivized to explore by the possibility that uncertain options might yield higher rewards than the current best in the long run. In the analogy we have constructed between MAB and AL, this translates into selecting queries using current optimal primary learners vs. exploring candidate nodes from learners that have resulted in very few actual queries. In both problems, our goal is to minimize *regret*, the difference between the realized and the optimal rewards incurred by over-exploration and myopic plays.

In our setting, the *regret*, as defined by (4.2) and (4.3), is the difference between $u(\mathcal{L}^{OPT})$ and $u(\mathcal{L})$. Since $\mathcal{L}^{OPT}$ is infeasible to compute, we instead consider the *pseudo-regret*

$$u(Q^{\lambda^*}) - u(\mathcal{L}) \tag{4.15}$$

where Q^{λ^*} is the labeled set produced by the best primary learner $\lambda^* = \text{argmax}_{\lambda \in \Lambda} \hat{\mu}_T(\lambda), T = \lceil B/b \rceil$. Based on our usage of CUCB in `MABAL`, results in [14] imply that (4.15) is bounded by $O(\log T)$ if the following conditions are satisfied:

- **Monotonicity**: given two sets of expected primary learner rewards $\bar{\mu}'(\lambda) \geq \bar{\mu}(\lambda) \forall \lambda \in \Lambda$, $\bar{\mu}'(Q) \geq \bar{\mu}(Q)$.
- **Bounded smoothness**: $\exists$ a strictly increasing function f such that $|\bar{\mu}(Q) - \bar{\mu}'(Q)| \leq f(A)$ if $\max_{\lambda \in \Lambda} |\bar{\mu}(\lambda) - \bar{\mu}'(\lambda)| \leq A$.

Additionally, the regret bound holds as long as Q computed by `Batch` is an $(\alpha - \beta)$ approximation of Q^{OPT}, i.e. $P(\mathrm{E}[u(Q|\mathcal{L})] \geq \alpha \cdot \mathrm{E}[u(Q^{OPT}|\mathcal{L})]) \geq \beta$.

Monotonicity is trivially satisfied by the definition of $\bar{\mu}(Q)$ because the reward expectation of Q is a simple sum of the expected rewards of nodes in Q, which are nonnegative linear combinations of $\bar{\mu}(\lambda)$. Since $|\bar{\mu}(Q) - \bar{\mu}'(Q)| = \sum_{q \in Q} \sum_{\lambda \in \Lambda} |\bar{\mu}(\lambda) - \bar{\mu}'(\lambda)| \cdot \bar{\mu}(q)$, the function that satisfies the bounded smoothness requirement is simply $f(x) = b \cdot x$.

We base the optimality analysis of `Batch` on the assumption that $\bar{\mu}(q)$ serves as an α-approximation of utility expectation for the classification model $\mathcal{F}$, i.e.,

$$\bar{\mu}(q) \geq \alpha \cdot \mathrm{E}[u(q|\mathcal{L})] \tag{4.16}$$

for some $\alpha \in [0, 1]$. First, observe that the result returned by `Batch` is trivially $\text{argmax}_{Q \in \mathcal{P}_b(\mathcal{U})} \bar{\mu}(Q)$. Let β be the probability that the additivity assumption in (4.12) holds for $\mathcal{F}$. Under the assumption, we have

$$\begin{aligned}\bar{\mu}(Q) = \sum_{q \in Q} \bar{\mu}(q) &\geq \sum_{q \in Q} \alpha \cdot \mathrm{E}[u(q|\mathcal{L})] \\ &= \alpha \cdot \mathrm{E}\left[\sum_{q \in Q} u(q|\mathcal{L})\right] = \alpha \cdot \mathrm{E}[u(Q|\mathcal{L})]\end{aligned}$$

Thus, with probability β, `Batch` finds a query set with expected reward $\geq \alpha \cdot \mathrm{E}[u(Q^{OPT}|\mathcal{L})]$. □

Chapter 5

Experiments

We wish to substantiate the following claims about our active learning algorithm via empirical evidence:

1. Our algorithm is independent of the underlying classification model.
2. Our algorithm is able to reduce label cost without compromising classification accuracy.
3. Our algorithm can adapt to different classification objectives on information networks.

We evaluate `MABAL` on three classification tasks over two real world datasets against simple heuristic and literature active learning baselines. To demonstrate that `MABAL` is not dependent on any particular collective classification method, we measure its gain on two different network classification models, RankClass[7] and Label Propagation[6]. RankClass is designed to specifically handle classification on HINs, whereas Label Propagation does not give special consideration to node types and handles all nodes equally. Both produce probability distributions for label predictions, which are compatible with our entropy reduction framework. Note that centrality values only need to be computed once for each network and can be shared across classification tasks.

5.1 Datasets

We evaluate our algorithm on HINs constructed from the *DBLP*[1] database and the MovieLens database[23]. DBLP is a bibliographic database of computer science publications. The DBLP HIN contains node types *author,*

[1]http://dblp.uni-trier.de/

paper, venue, and *term*, connected via three relations: (paper, author), (paper, venue), (paper, term). The DBLP network used in our experiments contains 14,376 papers, 20 conferences, 14,475 authors and 8,920 terms, with a total of 170,794 links. The MovieLens HIN contains nodes of type *movie, crew, origin, tag, user*, with *crew* being actors and directors associated with a movie, *origin* being a movie's country of origin, and *tag* being a user annotated phrase for a movie. In addition to the relations (movie, crew), (movie, origin), (movie, tag), the (user, movie) relation connects users to movies they have rated. Our MovieLens network contains 3415 movies, 14,692 actor/directors, 2113 users, 2678 tags and 42 countries, with a total of 434,861 links.

5.2 Baselines

We compare against the following baseline strategies:

- Random: b nodes are randomly selected from $\mathcal{U}$.
- Single centrality strategies: nodes in V are queried in descending order by a single centrality measure. We consider the following centrality measures: degree, PageRank, eigenvector, Katz, betweenness, closeness.
- ALFNET [3]: a cluster-based active learning algorithm on *homogeneous* networks with node features. ALFNET first clusters nodes based on their features to avoid sampling bias. It incorporates two classification models, `CO`, trained solely on the node features, and `CC`, trained on both node features and neighbor labels, and uses the disagreement between `CC` and `CO` as criterion for selecting queries within each cluster.
- MI [1]: an active learning algorithm on *homogeneous* networks that does not assume assortativity. Its query selection criterion is mutual information (MI) between a node's label and labels in the rest of the network according to the Gibbs distribution.
- HINAL[4]: an active learning algorithm on heterogeneous networks, which first performs clustering using *metapahts* and then selects queries within each cluster based on uncertainty sampling.

5.3 Experiments and Results

For the DBLP network, we consider the task of classifying nodes by their associated research area. Authors, papers and venues in the network are classified into one of the four areas {*Data Mining, Database, Machine Learn, Artificial Intelligence*}, with ground truth labels obtained in the same fashion as in [7]. We consider two classification tasks over the MovieLens network: 1) genre classification of movies and actors, 2) domestic vs. foreign classification for movies and actors. The network in our experiments contains three genres: {*Action, Romance, Thriller*} and about twice as many domestic films as foreign films.

For fairness of comparison, we perform classification using the same models, RankClass and Label Propagation, on all active learning strategies in the experiments. For ALFNET and MI, which are designed for homogeneous networks, we discard the node type information and use the same network topology for input to avoid loss in structural information due to any projection. Additionally, we created a view of the DBLP network in which the term nodes are removed and a binary term vector is created for every remaining node as input for ALFNET. The term vector for each paper node indicates the term nodes it directly links to, whereas the term vectors for the authors and conferences are a superposition of all term vectors for papers they link to.

Figure 5.1 shows the classification accuracies achieved in each AL algorithm by the number of input labels for the three classification tasks. For readability, we only include the best and worse centrality-based strategies to provide reference on the range of performance for `MABAL`. Note that the centrality baselines are not exactly the primary learners used in `MABAL`, since each learner only contains a single type of nodes. Due to this fact, `MABAL` may not fully converge to the best centrality baseline for some tasks. On the other hand, it also leads to `MABAL` outperforming all centrality baselines in tasks that contains one type of nodes that is much more informative than the others, as in Figure 5.1(e).

In the DBLP task, the initial strategy in `MABAL` underperforms most baselines on RankClass. However, it quickly converges to the optimal strategy, PageRank in this case, in just a few aggressive steps. Although *closeness* briefly surpasses the accuracy of PageRank, which caused the change in con-

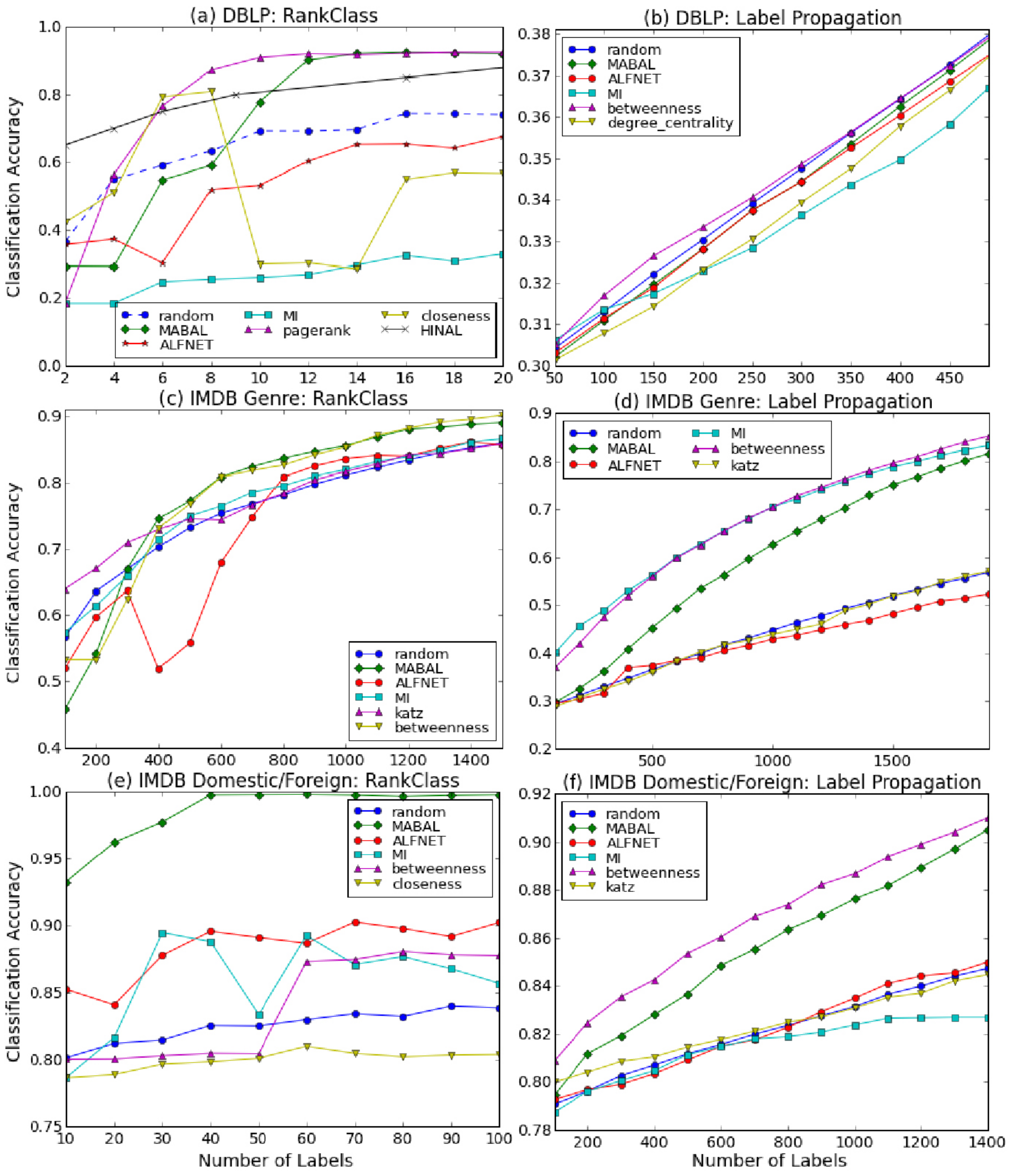

Figure 5.1: Classification accuracy v. number of labels.

vergence rate from 6 to 8, `MABAL` quickly adjusted the learner rewards to avoid a catastrophic performance decline had it continued to rely on *closeness*. In Figure 3(b), it is evident that the network structures used in all active learners fail to adequately capture the underlying classification model. However, unlike MI and ALFNET, `MABAL` provides a safety net against underperforming

Random.

As one would suspect, the centrality-by-type primary learners are more effective when performing classification with RankClass instead of Label Propagation, which disregards types. The slow convergence rate in (d) and (f) are largely due to the fact that all types of nodes receive equal treatment in Label Propagation, while `MABAL` does not contain a single strategy that ranks all types of nodes (other than *Random*). It can be easily remedied by using simple centrality primary learners instead of separating the nodes by types when working with homogeneous classification algorithms. (e) presents an interesting case in which node types play an extremely important role in the classification task. The large disparity between the performance of `MABAL` and the rest of the algorithms is due to `MABAL`'s ability to quickly recognize the importance of the "origin" nodes for the "Domestic/Foreign" task, which testifies to the cruciality of node types in active learning algorithms on HINs.

5.3.1 Hyperparameters

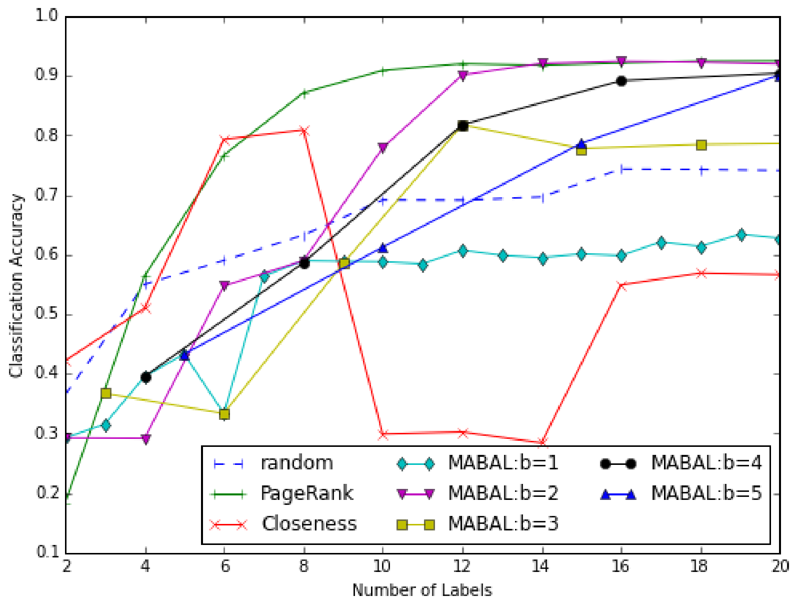

Figure 5.2: MABAL performance by batch size.

`MABAL` contains a number of hyperparameters that are intended to boost performance with prior knowledge on the classification task. In the "Default" setting, `MABAL` uses the Borda count $v^B(q)$, a neighborhood radius of

1, the label diversity factor without the *Random* learner. Table 4 shows the effect of these hyperparameters on the tasks we studied, with the top two options in bold for each task. Overall, the "Default" setting produced the optimal results for most tasks. The fact that label diversity slightly lowered performance for IMDB Genre with RankClass is due to label imbalance in the dataset, which can be addressed by using the K-L divergence instead of entropy. We can either use a prior distribution provided by the user or the overall observed label distribution as the prior in K-L divergence.

The results in Table 1 suggests that the Borda count is a strictly better strategy than centrality vote, and the first degree neighbors are sufficient for most tasks. In fact, since our HINs are bipartite, increasing the neighborhood radius to 2 introduces adverse effects on performance. Adding a single *Random* learner containing all nodes into the set of primary learners significantly hampered performance in the some settings. We intuit that this issue can be resolved by introducing a *Random* learner for each type of nodes instead.

We also investigated the effect of **batch size** on the convergence of our algorithm. As seen in Figure 5.2, batch mode with $b > 1$ provides significant gain over fully sequential learning, i.e., $b = 1$ because it avoids being pigeonholed into a subgraph and losing coverage. While a smaller batch size leads to faster convergence to λ^{OPT}, it also requires more frequent retraining. The choice for batch size thus involves consideration for the tradeoff between label cost and the cost of model training.

5.3.2 A Case Study

To better understand the behavior of `MABAL`, we examine its adaptation of query strategy based on observed labels for classifying the DBLP network by research area using RankClass. Table 2 shows the query order in the different active learning strategies whose performances can be found in Figure 5.1(a). The ranking of primary learners in `MABAL` at each iteration is shown in Table 3, with the top centrality in bold for each node type. The fact that the top queries are all conferences in PageRank, the optimal strategy, suggests that labels for *conference* nodes are the most informative for research area prediction in our network.

Table 5.1: Effect of Hyperparameters

Task	Default	$v_C(q)$	Add Random	$N_2(n)$	No Label Diversity
DBLP: RankClass	**0.924844**	0.590474	0.357587	**0.922451**	0.900191
DBLP: LabelProp	**0.456917**	0.454284	**0.460747**	0.454284	0.456438
IMDB Genre: RankClass	0.884394	0.885248	**0.897950**	0.875854	**0.903822**
IMDB Genre: LabelProp	**0.829419**	0.530102	0.585397	**0.708262**	0.684351
IMDB US: RankClass	**0.997921**	**0.997921**	0.949579	0.997193	0.997401
IMDB US: LabelProp	**0.933049**	0.879093	0.858405	**0.933985**	0.911841

Table 5.2: Query orders for DBLP with $b = 2$

i	`MABAL`	PageRank	ALFNET	MI
1	Jiawei Han, IJCAI	IJCAI, AAAI	B. T. Low, A. Sasturkar	WWW, CIKM
2	Philip S. Yu, AAAI	VLDB, ICDE	E. Hunt, X. Yuan	A. Bandyopadhyay, B. Rea
3	Christos Faloutsos, VLDB	SIGIR, SIGMOD	R. J. Peters, B. Smyth	P7561, P5848
4	ICDE, P4986	KDD, CIKM	C. Kellogg, A. Dasgupta	P13374, H. Wang
5	SIGIR, SIGMOD	ICML, ICDM	D. Plexousakis, D. Kelly	P6993, P3480
6	KDD, CIKM	PODS, PAKDD	K. Ali, O. Y. de Vel	P13458, Hans-Peter Kriegel
7	ICML, ICDM	WWW, EDBT	B. I. Blum, M. Yannakakis	P2353, Q. Yang

Table 5.3: Primary learners reward evolution: DBLP with RankClass

Primary Learner	T1	T2	T3	T4	T5
Paper:PageRank	7	7	7	**9**	**9**
Paper:Betweenness	8	8	8	10	10
Paper:Closeness	**5**	**5**	**5**	11	11
Paper:Katz	6	6	6	12	12
Author:Pagerank	**9**	**9**	**9**	**5**	**5**
Author:Betweenness	10	10	10	6	6
Author:Closeness	11	11	11	8	8
Author:Katz	12	12	12	7	7
Conf:Pagerank	**1**	**1**	**1**	**1**	**1**
Conf:Betweenness	2	2	2	3	3
Conf:Closeness	3	3	3	2	2
Conf:Katz	4	4	4	4	4

`MABAL` started with the belief that reputable conferences and authors are equally important. Although primary learners for papers are on average more highly ranked than authors at the beginning, author nodes were queried because there was more overlap between learners for authors after type bias correction. Note that a single paper query in iteration 4 was sufficient for `MABAL` to recognize uninformativeness of paper labels, which explains the poor performance of MI that mainly queried for papers. We can clearly see that starting in iteration 5, `MABAL` has switched over to the optimal strategy. The fact that ALFNET, which only queried for author labels, performed better than MI implies that authors are more informative than papers, which agrees with the ranking of primary learners in `MABAL` since T4. Thus, in addition to high performance, `MABAL` also provides insights into the functional roles of node types in the overall network.

Chapter 6

Conclusion and Future Work

In this work, we presented a novel and effective active learning algorithm for heterogeneous information networks. We focused on batch mode learning, which we have shown to be more effective on information networks than fully sequential learning. By establishing a correspondence between batch mode active learning on information networks and combinatorial multi-armed bandit, we proposed an expected error reduction based algorithm that combines simple strategies we dubbed *primary learners* to form query sets. Our algorithm employs a novel error expectation measure on networks that is highly adaptable to different classification tasks. Results for classification tasks on real world HINs demonstrated that our algorithm outperforms existing methods when applied to both homogeneous and heterogeneous network classification models. In addition to being adaptable and performant, our algorithm also provides insight into the network structures that are important for the given classification task.

The primary learners employed in this study were different types of nodes ranked by various common centrality measures computed over the whole network. While this choice yielded good performance for our tasks, we conjecture that more complex centrality measures that advantage of relation semantics, such as one using *metapaths* instead of direct links, could achieve even better performance on more sophisticated tasks. Additionally, a more rigorous study on performance bounds can be carried out by exploring the submodularity in the error expectation objective.

References

[1] C. Moore, X. Yan, Y. Zhu, J.-B. Rouquier, and T. Lane, “Active learning for node classification in assortative and disassortative networks,” in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2011, pp. 841–849.

[2] Y. Sun and J. Han, “Mining heterogeneous information networks: principles and methodologies,” *Synthesis Lectures on Data Mining and Knowledge Discovery*, vol. 3, no. 2, pp. 1–159, 2012.

[3] M. Bilgic, L. Mihalkova, and L. Getoor, “Active learning for networked data,” in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 79–86.

[4] C. Wan, X. Li, B. Kao, X. Yu, Q. Gu, D. Cheung, and J. Han, “Classification with active learning and meta-paths in heterogeneous information networks,” in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management.* ACM, 2015, pp. 443–452.

[5] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, “Collective classification in network data,” *AI magazine*, vol. 29, no. 3, p. 93, 2008.

[6] X. Zhu and Z. Ghahramani, “Learning from labeled and unlabeled data with label propagation,” *Technical Report CMU-CALD-02-107, Carnegie Mellon University*, 2002.

[7] M. Ji, J. Han, and M. Danilevsky, “Ranking-based classification of heterogeneous information networks,” in *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining.* ACM, 2011, pp. 1298–1306.

[8] B. Settles, “Active learning literature survey,” *Computer Sciences Technical Report 1648, University of Wisconsin, Madison*, vol. 52, no. 55-66, p. 11, 2010.

[9] N. Roy and A. McCallum, "Toward optimal active learning through monte carlo estimation of error reduction," *Proceedings of the 18th international conference on machine learning (ICML-01)*, pp. 441–448, 2001.

[10] X. Zhu, J. Lafferty, and Z. Ghahramani, "Combining active learning and semi-supervised learning using gaussian fields and harmonic functions," in *ICML 2003 workshop on the continuum from labeled to unlabeled data in machine learning and data mining*, vol. 3, 2003.

[11] W. Tong and R. Jin, "Semi-supervised learning by mixed label propagation," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 22, no. 1. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007, p. 651.

[12] R. Ganti and A. G. Gray, "Building bridges: Viewing active learning from the multi-armed bandit lens," in *Association for Uncertainty in Artificial Intelligence*, 2013, p. 232.

[13] Y. Baram, R. El-Yaniv, and K. Luz, "Online choice of active learning algorithms," *The Journal of Machine Learning Research*, vol. 5, pp. 255–291, 2004.

[14] W. Chen, Y. Wang, and Y. Yuan, "Combinatorial multi-armed bandit: General framework and applications," in *Proceedings of the 30th International Conference on Machine Learning*, 2013, pp. 151–159.

[15] L. Page, S. Brin, R. Motwani, and T. Winograd, "The pagerank citation ranking: bringing order to the web." *Stanford InfoLab Technical Report SIDL-WP-1999-0120, Stanford University*, 1999.

[16] U. Brandes and T. Erlebach, *Network analysis: methodological foundations.* Springer Science & Business Media, 2005, vol. 3418.

[17] A. Kapoor, E. Horvitz, and S. Basu, "Selective supervision: Guiding supervised learning with decision-theoretic active learning." in *Proceedings of the 20th international joint conference on artificial intelligence (IJCAI-07)*, vol. 7, 2007, pp. 877–882.

[18] B. Settles, M. Craven, and L. Friedland, "Active learning with real annotation costs," in *Proceedings of the NIPS workshop on cost-sensitive learning*, 2008, pp. 1–10.

[19] S. Vijayanarasimhan and K. Grauman, "What's it going to cost you?: Predicting effort vs. informativeness for multi-label image annotations," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on.* IEEE, 2009, pp. 2262–2269.

[20] J. Gittins, K. Glazebrook, and R. Weber, *Multi-armed bandit allocation indices.* John Wiley & Sons, 2011.

[21] J.-Y. Audibert, S. Bubeck, and G. Lugosi, "Minimax policies for combinatorial prediction games," in *COLT-24th Conference on Learning Theory-2011*, 2011, pp. in–press.

[22] D. Easley and J. Kleinberg, *Networks, crowds, and markets: Reasoning about a highly connected world.* Cambridge University Press, 2010.

[23] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 5, no. 4, p. 19, 2015.