Selective Sampling on Graphs for Classification

Quanquan Gu[†], Charu Aggarwal[‡], Jialu Liu[†], Jiawei Han[†]
[†]Dept. of Computer Science, University of Illinois at Urbana-Champaign, IL 61801, USA
[‡]IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA
{qgu3,jliu64,hanj}@illinois.edu, charu@us.ibm.com

ABSTRACT

Selective sampling is an active variant of online learning in which the learner is allowed to adaptively query the label of an observed example. The goal of selective sampling is to achieve a good trade-off between prediction performance and the number of queried labels. Existing selective sampling algorithms are designed for vector-based data. In this paper, motivated by the ubiquity of graph representations in real-world applications, we propose to study selective sampling on graphs. We first present an online version of the well-known Learning with Local and Global Consistency method (OLLGC). It is essentially a second-order online learning algorithm, and can be seen as an online ridge regression in the Hilbert space of functions defined on graphs. We prove its regret bound in terms of the structural property (cut size) of a graph. Based on OLLGC, we present a selective sampling algorithm, namely Selective Sampling with Local and Global Consistency (SSLGC), which queries the label of each node based on the confidence of the linear function on graphs. Its bound on the label complexity is also derived. We analyze the low-rank approximation of graph kernels, which enables the online algorithms scale to large graphs. Experiments on benchmark graph datasets show that OLLGC outperforms the state-of-the-art first-order algorithm significantly, and SSLGC achieves comparable or even better results than OLLGC while querying substantially fewer nodes. Moreover, SSLGC is overwhelmingly better than random sampling.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; I.5.3 [**Pattern Recognition**]: Classification

General Terms

Algorithms, Experimentation, Performance

KDD'13, August 11–14, 2013, Chicago, Illinois, USA.

Keywords

Selective sampling on graphs, Online learning, Regret bound, Mistake bound, Label complexity

1. INTRODUCTION

Selective sampling [13] [8] is an active variant of online learning [9] in which the learner is allowed to adaptively query the labels of a sequence of examples. The learner's goal is to achieve a good trade-off between error rate and the number of queried labels. This can be viewed as an abstract protocol for interactive learning applications. Recently, several advanced selective sampling algorithms [6] [24] were proposed, demonstrating more promising results than traditional passive online learning. However, we note that existing selective sampling algorithms are specifically designed for vector-based data.

Graphs have recently received significant attention because of their increasingly important role in real life applications. Examples include the friendship network in *Facebook*[1], co-author and citation networks in *DBLP*[2], and the World Wide Web. In these applications, the data (nodes) are not independent and identically distributed (i.i.d.) as is typically assumed in statistical learning applications, because of the impact of the linkage structure of the graph. Learning a function defined on a graph from a set of labeled nodes has been studied extensively in machine learning both in off-line and online settings. More specifically, in the offline learning scenario, a majority of the literature is often referred to as graph-based semi-supervised learning [30] [29]. On the other hand, the pioneering work towards online learning on graphs is probably [19]. Inspired by this work, the state-of-the-art Graph Perceptron Algorithm (GPA) was proposed in [18] and further analyzed in [17] [16].

Based on the above observation, a natural question arises as to whether we can design selective sampling algorithms for graphs. The results of this paper show that the answer is in the affirmative. In this paper, we propose to study selective sampling on graphs. Our work is built on a well-known model on graphs, namely learning with local and global consistency. This model is a state-of-the-art model on graphs and is particularly amenable to analysis in the context of selective sampling. We first present an online version of the well-known Learning with Local and Global Consistency method (OLLGC). It is essentially a second-order online learning algorithm, and can be seen as an on-

[1]http://www.facebook.com
[2]http://www.informatik.uni-trier.de/∼ley/db/

line ridge regression in the Hilbert space of functions defined on a graph. We prove its regret bound in terms of cut size of a graph. Based on OLLGC, we present a selective sampling algorithm, namely Selective Sampling with Local and Global Consistency (SSLGC), which queries the labels of nodes based on the confidence of the linear function on graphs. We also derive a bound on the label complexity of our proposed algorithm. Lastly, in order to scale the proposed algorithms as well as existing online learning algorithms to large graphs, we discuss the low-rank approximation technique for graph kernels. Experiments on benchmark graph datasets show that OLLGC outperforms GPA [18] substantially. Furthermore, the selective sampling algorithm (SSLGC) achieves comparable or even better results than OLLGC, while querying substantially fewer nodes. Moreover, SSLGC provides superior results to random sampling.

The main contributions of this paper are three-fold: (1) we present an online learning with local and global consistency (OLLGC), and prove its regret bound; (2) we present a selective sampling algorithm on graphs based on OLLGC, and derive its bound on label complexity; and (3) we analyze the low-rank approximation of graph kernels, which enables greater scalability of our algorithms as well as existing algorithms, when the graphs are large.

The remainder of this paper is organized as follows. In Section 2, we briefly review the related literature. In Section 3, we present an online version of learning with local and global consistency, followed by its regret bound. In Section 4, we devise a selective sampling algorithm on graphs based on the online algorithm derived in previous section, and analyze its bound on the label complexity. We discuss and analyze the low-rank approximation of graph kernels for online algorithms in Section 5. The experiments on benchmark graph datasets are demonstrated in Section 6. Finally, we present the conclusions in Section 7.

1.1 Notation

Throughout this paper, we will use lower case letters to denote scalars, lower case bold letters to denote vectors (e.g., $\mathbf{w}$), upper case letters to denote the elements of a matrix or a set, and bold-face upper case letters to denote matrices (e.g., $\mathbf{A}$). $\mathbf{0}$ is a vector of all zeros with appropriate length. $\mathbf{I}$ is an identity matrix with an appropriate size. We use $\mathbf{w}^\top$ denote the transpose of a vector $\mathbf{w}$, and $\mathbf{A}^{-1}$ the inverse of a matrix $\mathbf{A}$. Given a matrix $\mathbf{L}$, $\mathbf{L}^\dagger$ denotes its pseudo inverse. $\mathrm{diag}(\sigma_1, \ldots, \sigma_n)$ denotes a diagonal matrix with diagonal elements equal to σ_i's. Furthermore, we use $\|\cdot\|$ denote the ℓ_2-norm of a vector.

2. RELATED WORK

For ease in exposition, we briefly discuss online learning, active learning and selective sampling, in the context of both vector-based data and graph data.

2.1 Online Learning

Online learning has been studied extensively in the machine learning community. In the past several decades, a variety of online learning algorithms have been proposed. Due to the sequential nature of online learning, it is very suitable to be applied to big data from many real-world applications. Roughly speaking, online learning algorithms can be categorized into first-order algorithms [25] [23] and second-order algorithms [5] [12]. In general, second-order online algorithms are better than first-order online algorithms [20].

The extension of online learning to graph data was originally studied in [19]. After that work, the well-known Graph Perceptron Algorithm (GPA) was proposed in [18] and further analyzed in [17] [16]. It is worth noting that the setting of online learning on graphs is essentially *transductive*, where the whole graph is already provided, but the learner is presented with the nodes in a sequential manner. This is different from the *inductive* paradigm for vector-based online learning. In addition, all the online learning algorithms on graphs mentioned before are first-order algorithms. Note that the first contribution of our paper, i.e., online learning with local and global consistency is a second-order algorithm on graphs, which is better than first-order algorithms.

2.2 Active Learning

Active learning [11] [28] aims to minimize the required level of acquisition of labeled data by actively selecting a few carefully chosen examples to query the oracle for their labels. There are several papers on active learning on graphs. For instance, [1] proposed an effective label acquisition for collective classification. [2] proposed an active learning algorithm for networked data based on ensemble and relational learning. Yet, there is no theoretical guarantee that these methods are better than random sampling. [7] studied active learning on graphs and trees. [21] proposed a nonadaptive active learning method by minimizing the variance of Gaussian Field and Harmonic Function (GFHF) [30]. In our previous work [15], we proposed a nonadaptive active learning approach on graphs, by minimizing the data-dependent error bound of LLGC [29], which was shown to be better than [21].

2.3 Selective Sampling

Selective sampling [8] [6] combines the idea of online learning and active learning. Similar to online learning, a selective sampling algorithm observes examples in a sequential manner. After each observation, the algorithm predicts its label. However, rather than receiving the correct label passively, the algorithm can choose whether to receive feedback indicating whether the label is correct or not. It is obvious that by using selective sampling, we need much less labeling effort, since the labels of many examples can be predicted with very high confidence. In other words, selective sampling is online active learning.

Linear models lend themselves well to selective sampling settings, because the variance of a classifier on an example can be viewed as a measure of confidence for the classification. If this confidence is too low, then the selective sampler will query the label and use it, along with the example, to update the linear model. For graph data, the key question is how to define an *example* as well as a linear model. We will show that learning with local and global consistency can be equivalently formulated as a *linear* model on the graphs.

3. ONLINE LEARNING WITH LOCAL AND GLOBAL CONSISTENCY

In this section, we present an online version of learning with local and global consistency (LLGC) [29]. To make our paper self-contained, we briefly review LLGC.

3.1 Learning with Local and Global Consistency

Given a graph $G = (V, E)$, where $v_i \in V$ is the i-th node of a graph, and $e_{ij} \in E$ is the link (edge) between i-th node and the j-th node. Each link e_{ij} is associated with a weight S_{ij}, which reflects the strength of the link. $\mathbf{S} \in \mathbb{R}^{n \times n}$ is called adjacency matrix of the graph. For undirected graph, $\mathbf{S}$ is a symmetric matrix, while for directed graph, $\mathbf{S}$ is asymmetric. In the setting of transductive classification, some of the nodes in the graph are labeled, i.e., $y_i \in \{\pm 1\}$, while the remainder are unlabeled, i.e., $y_i = 0$. Our goal is to obtain a prediction about the labels of those unlabeled nodes. Through our paper, we assume that the graph G is connected, though our results can be generalized to disconnected graphs with more involved arguments.

The basic assumption of graph regularization is based on the concept of homophily in networks. If two nodes v_i and v_j are linked together, then their labels are likely to be similar. Let $f : V \to \mathbb{R}$ be a nonparametric function defined on the nodes of a graph. For an undirected graph, graph regularization [27] is mathematically written as follows:

$$\frac{1}{2}\sum_{i,j=1}^{n}(f_i - f_j)^2 S_{ij} = \mathbf{f}^\top \mathbf{L}\mathbf{f}, \tag{1}$$

where f_i is the function value on the i-th node, i.e., $f(v_i)$, $\mathbf{f} = [f_1, \ldots, f_n]^T$, $\mathbf{D}$ is a diagonal matrix, which is also referred to as the degree matrix. The ith diagonal entry $D_{ii} = \sum_{j=1}^{n} S_{ij}$, $\mathbf{L} = \mathbf{D} - \mathbf{S}$ is the combinatorial graph Laplacian [10]. Eq. (1) is called *Graph Regularization*. Intuitively, the objective function incurs a heavy penalty, if neighboring nodes v_i and v_j are mapped far apart. Suppose the eigen decomposition of $\mathbf{L}$ is $\mathbf{L} = \mathbf{V}\mathbf{\Sigma}\mathbf{V}^\top = \sum_{i=1}^{n}\sigma_i \mathbf{v}_i \mathbf{v}_i^\top$, where $\mathbf{\Sigma} = \mathrm{diag}(\sigma_1, \ldots, \sigma_n)$, $0 \le \sigma_1 \le \sigma_2 \le \ldots \le \sigma_n$ are eigenvalues, $\mathbf{V} = [\mathbf{v}_1, \ldots, \mathbf{v}_n]$, and $\mathbf{v}_i \in \mathbb{R}^n, i = 1, \ldots, n$ are eigenvectors. One property of the graph Laplacian is that its smallest eigenvalue is 0 (i.e., $\sigma_1 = 0$), and the associated eigenvector is $\mathbf{1}$. For connected graphs, the algebraic multiplicity of the zero eigenvalue is 1 (i.e., $\sigma_2 > 0$).

Learning with Local and Global Consistency (LLGC) [29] was originally proposed for semi-supervised learning and latter successfully used for classification on graphs [22]. In the setting of binary classification, it solves the following problem,

$$\min_{\mathbf{f}} \frac{1}{2}\|\mathbf{f} - \mathbf{y}\|^2 + \frac{\mu}{2}\mathbf{f}^\top \mathbf{L}\mathbf{f}, \tag{2}$$

where $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$ is the label vector, $\mu > 0$ is a regularization parameter, which controls the balance between the squared loss and the graph regularization.

3.2 An Equivalent Formulation

In order to derive the online version of LLGC, we derive an equivalent formulation of LLGC as follows. Specifically, we consider the dual problem of Eq. (2). Using the definition of graph kernel [27], we have

$$\mathbf{f} = \mathbf{L}^\dagger \boldsymbol{\alpha}, \tag{3}$$

where $\mathbf{L}^\dagger$ is the inverse (or pseudo inverse) of $\mathbf{L}$, i.e., $\mathbf{L}^\dagger = \sum_{i=2}^{n}\frac{1}{\sigma_i}\mathbf{v}_i\mathbf{v}_i^\top$. Without loss of generality, we assume that $\|\boldsymbol{\alpha}\|^2 \le C$, where $C > 0$ is a constant.

Substituting Eq. (3) back into Eq. (2), we have

$$\min_{\boldsymbol{\alpha}} \frac{1}{2}\|\mathbf{L}^\dagger\boldsymbol{\alpha} - \mathbf{y}\|^2 + \frac{\mu}{2}\boldsymbol{\alpha}^\top \mathbf{L}^\dagger \boldsymbol{\alpha}. \tag{4}$$

We assume that $\mathbf{L}^\dagger = \mathbf{M}^T\mathbf{M}$, where $\mathbf{M} \in \mathbb{R}^{d \times n}$. We define $\mathbf{w} = \mathbf{M}\boldsymbol{\alpha}$. The optimization problem in Eq. (4) can be rewritten as follows:

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{M}^\top \mathbf{w} - \mathbf{y}\|^2 + \frac{\mu}{2}\|\mathbf{w}\|^2. \tag{5}$$

Now we can see that the above objective function is essentially a ridge regression, where each column of $\mathbf{M}$ can be seen as a vector-based example. This insight enables us adapt the technique from online ridge regression to derive an online version of LLGC. We will discuss the selection of $\mathbf{M}$ in Section 5.

3.3 Online Learning

Now we are ready to propose the online version of LLGC. Before that, let us state the formal problem setting of online learning on graphs. From now on, we assume $T = n$. Let $\mathbf{M} = [\mathbf{m}_1, \ldots, \mathbf{m}_T]$, where $\mathbf{m}_i \in \mathbb{R}^d$ is the i-th column of $\mathbf{M}$. Online learning operates on a sequence of nodes. In round t, the algorithm receives an incoming node $\mathbf{m}_t \in \mathbb{R}^d$, and predicts its label $\hat{y}_t \in \{-1, +1\}$. After the prediction, the true label $y_t \in \{-1, +1\}$ is revealed and the loss $\ell(y_t, \hat{y}_t)$ is evaluated. The goal of online learning is to minimize the cumulative number of mistakes over the entire graph.

Given $\{(\mathbf{m}_1, y_1), (\mathbf{m}_2, y_2), \ldots, (\mathbf{m}_t, y_t)\}, 1 \le t \le T$, where $\mathbf{m}_t \in \mathbb{R}^d$ and $y_t \in \{-1, 1\}$, online LLGC aims at solving the following optimization problem:

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \frac{1}{2}\sum_{i=1}^{t}(\mathbf{m}_i^\top \mathbf{w} - y_i)^2 + \frac{\mu}{2}\|\mathbf{w}\|^2. \tag{6}$$

It is worth noting that the above problem is a *Follow-the-Regularized-Leader* problem [26], which has been extensively studied in the online learning community.

The optimal solution for $\mathbf{w}_{t+1}$ to Eq. (6) is

$$\mathbf{w}_{t+1} = (\sum_{i=1}^{t}\mathbf{m}_i\mathbf{m}_i^\top + \mu\mathbf{I})^{-1}\sum_{i=1}^{t}\mathbf{m}_i y_i. \tag{7}$$

We define $\mathbf{A}_0 = \mu\mathbf{I}$, $\mathbf{A}_t = \mu\mathbf{I} + \sum_{i=1}^{t}\mathbf{m}_i\mathbf{m}_i^\top$, $\mathbf{b}_0 = \mathbf{0}$, and $\mathbf{b}_t = \sum_{i=1}^{t} y_i\mathbf{m}_i$, Then, we have:

$$\mathbf{w}_{t+1} = \mathbf{A}_t^{-1}\mathbf{b}_t. \tag{8}$$

The calculation $\mathbf{A}_t^{-1}$ seems to be computationally expensive. Fortunately, we do not need to calculate $\mathbf{A}_t^{-1}$ explicitly. In fact, $\mathbf{A}_t^{-1}$ can be incrementally calculated by the Sherman-Morrisan Identity [14]. In addition, the above update is performed in each iteration, which is not sufficiently efficient for large graphs. To resolve this problem, inspired by the mistake-driven algorithms such as Second-Order Perceptron (SOP) [5], we let our online algorithm update the model parameters ($\mathbf{A}$, $\mathbf{b}$ and $\mathbf{w}$) only when it incurs a mistake ($\hat{y}_t \ne y_t$). Note that this modification does not affect the soundness of our algorithm, as will be seen in our theoretical analysis. Furthermore, our algorithm is different from SOP either, because it does not use the current node to update the weight vector ($\mathbf{w}$) until the label of current node is revealed. In summary, we show the proposed online LLGC in Algorithm 1.

Algorithm 1 Online Learning with Local and Global Consistency (OLLGC)

Input: Adjacency matrix $\mathbf{S}$, rank d, regularization parameter μ
Output: $\mathbf{w}_T$
Compute $\mathbf{L} = \mathbf{D} - \mathbf{S}$ and $\mathbf{M}$ from $\mathbf{L}$
Initialize: $\mathbf{A}_0 = \mu\mathbf{I}$, $\mathbf{b}_0 = \mathbf{0}$, $\mathbf{w}_0 = \mathbf{0}$
for $t = 1$ to T **do**
  Receive $\mathbf{m}_t \in \mathbb{R}^d$ and Predict $\hat{y}_t = \text{sign}(\mathbf{w}_{t-1}^\top \mathbf{m}_t)$
  Receive the correct label $y_t \in \{\pm 1\}$
  if $\hat{y}_t \neq y_t$ **then**
    Update $\mathbf{A}_t = \mathbf{A}_{t-1} + \mathbf{m}_t\mathbf{m}_t^\top$
    Update $\mathbf{b}_t = \mathbf{b}_{t-1} + y_t\mathbf{m}_t$
    Update $\mathbf{w}_t = \mathbf{A}_t^{-1}\mathbf{b}_t$
  else
    $\mathbf{A}_t = \mathbf{A}_{t-1}$, $\mathbf{b}_t = \mathbf{b}_{t-1}$, $\mathbf{w}_t = \mathbf{w}_{t-1}$
  end if
end for

Note that in each iteration of our algorithm, whenever an update is invoked, the time complexity is $O(d^2)$.

3.4 Theoretical Analysis

Now we will prove the regret bound of OLLGC. This bound shows that, for any ordering of nodes on a graph, our algorithm cannot perform much worse than the best predictor learned in hindsight. The proof technique is adapted from potential-based gradient descent [9] (a.k.a., mirror descent [26]), as well as SOP [5].

First, we define the regret of OLLGC as follows:

$$R_T = \sum_{t=1}^{T} \ell_t(\mathbf{w}_t) - \sum_{t=1}^{T} \ell_t(\mathbf{u}), \tag{9}$$

where $\ell_t(\mathbf{w}_t) = \frac{1}{2}(\mathbf{w}_t^\top \mathbf{m}_t - y_t)^2$ and $\ell_t(\mathbf{u}) = \frac{1}{2}(\mathbf{u}^\top \mathbf{m}_t - y_t)^2$.

For the ease of proof, we define a set $\mathcal{M} = \{t : \text{sign}(\mathbf{w}_{t-1}^\top \mathbf{x}_t) \neq y_t\}$, which is the set of round indices for which an algorithm makes a mistake. We rewrite Eq. (6) as a potential-based gradient descent problem:

$$\mathbf{w}_{t+1} = \arg\min_{\mathbf{w}} \frac{1}{2}(\mathbf{m}_t^\top \mathbf{w} - y_t)^2 + D_{\phi_{t-1}}(\mathbf{w}, \mathbf{w}_t), \tag{10}$$

where $D_{\phi_{t-1}}(\mathbf{w}, \mathbf{w}_t)$ is the Bregman divergence [4], defined as follows:

$$D_{\phi_{t-1}}(\mathbf{w}, \mathbf{w}_t) = \phi_{t-1}(\mathbf{w}) - \phi_{t-1}(\mathbf{w}_t) + \langle \nabla\phi_{t-1}(\mathbf{w}_t), \mathbf{w} - \mathbf{w}_t \rangle, \tag{11}$$

and ϕ_t is a potential function, defined as follows:

$$\phi_t(\mathbf{w}) = \frac{1}{2}\mathbf{w}^\top \mathbf{A}_t \mathbf{w} - \mathbf{w}^\top \mathbf{b}_t + \frac{1}{2}\sum_{i=1}^{t} y_i^2. \tag{12}$$

It is easy to verify that the optimization problems in Eqs. (6) and (10) are equivalent. Moreover, we have $\nabla\phi_t(\mathbf{w}_{t+1}) = 0$, and $\ell_t(\mathbf{u}) = \phi_t(\mathbf{u}) - \phi_{t-1}(\mathbf{u})$. Since we incorporated the mistake-driven update strategy into OLLGC, $\mathbf{A}_t$ is actually defined as $\mathbf{A}_t = \sum_{i=1}^{t} \mathbf{m}_i\mathbf{m}_i^\top \mathbb{I}[i \in \mathcal{M}]$.

We begin with three technical lemmas, which facilitate the proofs of the main theoretical result of OLLGC. The first lemma is a property of potential-based gradient descent.

Lemma 1 *For any* $\mathbf{u}$*, we have*

$$\sum_{t=1}^{T}(\ell_t(\mathbf{w}_t) - \ell_t(\mathbf{u})) \leq D_{\phi_0}(\mathbf{u}, \mathbf{w}_1) + \sum_{t=1}^{T} D_{\phi_t}(\mathbf{w}_t, \mathbf{w}_{t+1}). \tag{13}$$

The second lemma is an upper bound of $\sum_{t\in\mathcal{M}} \mathbf{m}_t^\top \mathbf{A}_t^{-1}\mathbf{m}_t$. Similar lemma has been proved in [5] [9].

Lemma 2 *Assume that* $\|\mathbf{m}_t\|^2 \leq B$ *for all* t*, then for Algorithm 1, we have*

$$\sum_{t\in\mathcal{M}} \mathbf{m}_t^\top \mathbf{A}_t^{-1}\mathbf{m}_t \leq \sum_{i=1}^{d} \log\left(1 + \frac{\lambda_i}{\mu}\right) \leq d\log\left(1 + \frac{|\mathcal{M}|B}{d\mu}\right), \tag{14}$$

where $\lambda_i, i = 1, \ldots, d$ *are the eigenvalues of* $\sum_{t=1}^{T} \mathbf{m}_t\mathbf{m}_t^\top \mathbb{I}[i \in \mathcal{M}]$.

The third lemma relates the norm $\|\mathbf{u}\|^2$ with the structural property of a graph.

Lemma 3 *Suppose* G *is a connected graph, for any* $\mathbf{u} = \mathbf{M}\boldsymbol{\alpha}$*,* $f_i \in \{-1, 1\}, i = 1, \ldots, n$ *and* $\|\boldsymbol{\alpha}\|^2 \leq C$*, we have*

$$\|\mathbf{u}\|^2 = \mathbf{f}^\top \mathbf{L}\mathbf{f} = 2\Phi(\mathbf{f}), \tag{15}$$

where $\Phi(\mathbf{f})$ *is the cut size corresponding to the class assignment of* $\mathbf{f}$*.*

Theorem 4 *(Regret Bound) Let* $S = \{(\mathbf{m}_1, y_1), \ldots, (\mathbf{m}_T, y_T)\} \in (\mathbb{R}^d \times \{\pm 1\})^T$*. Then for any* $\mathbf{u} \in \mathbb{R}^d$ *such that* $(\mathbf{u}^T\mathbf{m}_t - y_t)^2 \leq \gamma$*,* $\mathbf{f} \in \{-1, 1\}^T$*, and* $\|\boldsymbol{\alpha}\|^2 \leq C$*, we have*

$$R_T \leq \mu\Phi(\mathbf{f}) + \gamma\sum_{i=1}^{d} \log\left(1 + \frac{\lambda_i}{\mu}\right) \tag{16}$$

Proof. Using Lemma 1, we have

$$\begin{aligned} R_T &\leq D_{\phi_0}(\mathbf{u}, \mathbf{w}_1) + \sum_{t=1}^{T} D_{\phi_t}(\mathbf{w}_t, \mathbf{w}_{t+1}) \\ &= \frac{\mu}{2}\|\mathbf{u}\|^2 + \sum_{t=1}^{T} D_{\phi_t^*}(\nabla\phi_t(\mathbf{w}_{t+1}), \nabla\phi_t(\mathbf{w}_t)), \end{aligned} \tag{17}$$

where $\phi^*(\cdot)$ is the Fenchel conjugate function [3] of $\phi(\mathbf{w})$, and here we used a very useful property of Bregman divergence [9]. Since $\ell_t(\mathbf{w}_t) = (\mathbf{w}_t^\top \mathbf{m}_t - y_t)\mathbf{m}_t$, and $D_{\phi_t^*}(\mathbf{u}, \mathbf{w}) = (\mathbf{u} - \mathbf{w})^\top \mathbf{A}_t^{-1}(\mathbf{u} - \mathbf{w})$, we have

$$\begin{aligned} D_{\phi_t^*}(0, \nabla\ell_t(\mathbf{w}_t)) &= (\mathbf{w}_t^\top \mathbf{m}_t - y_t)^2 \mathbf{m}_t^\top \mathbf{A}_t^{-1}\mathbf{m}_t \\ &\leq \gamma\mathbf{m}_t^\top \mathbf{A}_t^{-1}\mathbf{m}_t. \end{aligned} \tag{18}$$

Using Lemmas 2 and 3 completes the proof. □

In fact, we can also bound the number of mistakes made by Algorithm 1 for any ordering of nodes on a graph.

Corollary 5 *(Mistake Bound) Let* $S = \{(\mathbf{m}_1, y_1), \ldots, (\mathbf{m}_T, y_T)\} \in (\mathbb{R}^d \times \{\pm 1\})^T$*. Then for any* $\mathbf{u} \in \mathbb{R}^d$ *such that* $(\mathbf{u}^T\mathbf{m}_t - y_t)^2 \leq \gamma$*, we have*

$$|\mathcal{M}| \leq \min_{\mathbf{f}} \frac{1}{2}\|\mathbf{f} - \mathbf{y}\|^2 + \frac{\mu}{2}\mathbf{f}^\top \mathbf{L}\mathbf{f} + \gamma d\log\left(1 + \frac{TB}{d\mu}\right) \tag{19}$$

This bound is very interesting, because it directly implies that the better the off-line LLGC works on a graph, the smaller the number of mistakes made by OLLGC. This is consistent with our intuition.

4. SELECTIVE SAMPLING WITH LOCAL AND GLOBAL CONSISTENCY

In this section, we will present a selective sampling algorithm based on OLLGC proposed in previous section. First of all, we formally give the definition of selective sampling on graphs.

4.1 Problem Definition

Selective sampling is a modification of the online learning protocol for binary classification. At each round t, the learner receives a node $\mathbf{m}_t \in \mathbb{R}^d$, and outputs a binary prediction $\hat{y}_t \in \{-1, 1\}$. After each prediction, the learner may observe the true label y_t only by querying for it. Hence, if no query is issued at time t, then y_t remains unknown. Since the learner's performance is deemed to improve as more labels are observed, the goal of selective sampling is to trade off predictive performance and the number of queries.

4.2 Algorithm

In our paper, following [6], we assume that $\Pr(Y_t = 1|\mathbf{m}_t) = \frac{1+\mathbf{u}^\top \mathbf{m}_t}{2}$ for some $\mathbf{u} \in \mathbb{R}^d$. Hence $\mathbb{E}[Y_t|\mathbf{m}_t] = \mathbf{u}^\top \mathbf{m}_t$. Here $\mathbf{u}$ is the Bayes classifier of unknown norm $\|\mathbf{u}\|$ which satisfies $|\mathbf{u}^\top \mathbf{m}_t| \leq 1$ for all t. We also define $\Delta_t = \mathbf{u}^\top \mathbf{m}_t$. We further define $\hat{\Delta}_t = \mathbf{w}_t^\top \mathbf{m}_t$, which is an estimator of Δ_t.

Our algorithm is motivated by the Bound on Bias Query (BBQ) algorithm [6] [24]. We introduce the following relevant quantities,

$$\begin{aligned} B_t &= \mathbf{u}^\top (\mathbf{I} + \mathbf{m}_t \mathbf{m}_t^\top) \mathbf{A}_t^{-1} \mathbf{m}_t \\ r_t &= \mathbf{m}_t^\top \mathbf{A}_t^{-1} \mathbf{m}_t. \end{aligned} \tag{20}$$

where B_t is the bias of the estimator for the margin $\hat{\Delta}_t$, and r_t is a bound on the variance.

Different from BBQ algorithm, the learner in our algorithm does not necessarily update the model whenever it queries the label. Instead, it updates the model when it queries the label and a mistake is detected. This makes SSLGC computationally more efficient, without significantly affecting the theoretical properties. In summary, we show the selective sampling with local and global consistency in Algorithm 2.

Intuitively speaking, our algorithm issues a query when a common upper bound on the bias and variance of the current estimate of $\hat{\Delta}_t$ is larger than a given threshold vanishing as $t^{-\kappa}$, where $0 \leq \kappa \leq 1$ is an input parameter. When this upper bound on bias and variance gets small, we infer by a simple large deviation argument that the margin of OLLGC on the current example is close enough to the margin of the Bayes optimal classifier. Hence the learner can safely avoid issuing a query in that round. In each iteration of the algorithm, whenever an update is invoked, the time complexity is $O(d^2)$.

4.3 Theoretical Analysis

We define the regret of our selective sampling algorithm as follows:

$$R_T = \sum_{t=1}^{T} \left(\Pr(Y_t \hat{\Delta}_t < 0) - \Pr(Y_t \Delta_t < 0) \right), \tag{21}$$

uniformly over the number T of prediction rounds. Following previous papers [6] [24], our bound can depend on the number of rounds where the label Y_t are close to being random. According to our model, this is captured by ϵT_ϵ where $T_\epsilon = |\{1 \leq t \leq T : |\Delta_t| < \epsilon\}|$.

Our main theoretical result provides bounds on the cumulative regret and the number of queried labels (label complexity) for Algorithm 2. We begin with a technical lemma.

Algorithm 2 Selective Sampling with Local and Global Consistency (SSLGC)

Input: Adjacency matrix $\mathbf{S}$, rank d, regularization parameter μ, and κ.
Output: $\mathbf{w}_T$
Compute $\mathbf{L} = \mathbf{D} - \mathbf{S}$ and $\mathbf{M}$ from $\mathbf{L}$
Initialize: $\mathbf{A}_0 = \mu\mathbf{I}$, $\mathbf{b}_0 = \mathbf{0}$, $\mathbf{w}_0 = \mathbf{0}$
for $t = 1$ to T **do**
  Receive $\mathbf{m}_t \in \mathbb{R}^d$ and Predict $\hat{y}_t = \text{sign}(\mathbf{w}_{t-1}^\top \mathbf{m}_t)$
  if $r_t > t^{-\kappa}$ **then**
    Query the correct label $y_t \in \{\pm 1\}$
    if $\hat{y}_t \neq y_t$ **then**
      Update $\mathbf{A}_t = \mathbf{A}_{t-1} + \mathbf{m}_t \mathbf{m}_t^\top$
      Update $\mathbf{b}_t = \mathbf{b}_{t-1} + y_t \mathbf{m}_t$
      Update $\mathbf{w}_t = \mathbf{A}_t^{-1} \mathbf{b}_t$
    else
      $\mathbf{A}_t = \mathbf{A}_{t-1}$, $\mathbf{b}_t = \mathbf{b}_{t-1}$, $\mathbf{w}_t = \mathbf{w}_{t-1}$
    end if
  else
    $\mathbf{A}_t = \mathbf{A}_{t-1}$, $\mathbf{b}_t = \mathbf{b}_{t-1}$, $\mathbf{w}_t = \mathbf{w}_{t-1}$
  end if
end for

Lemma 6 *For all $\epsilon > 0$, we have*

$$\begin{aligned} & \sum_{t=1}^{T} \Pr(|\hat{\Delta}_t - \Delta_t| \geq \epsilon) \\ \leq\ & \lceil \frac{1}{\kappa} \rceil ! \left(2(\frac{8}{\epsilon^2})^{\frac{1}{k}} + e(\frac{4(B\|\mathbf{u}\|^2 + \epsilon)}{\epsilon^2})^{\frac{1}{k}} \right) \\ +\ & \left(\frac{16}{e\epsilon^2} + \frac{4(B\|\mathbf{u}\|^2 + \epsilon)}{\epsilon^2} \right) d \ln \left(1 + \frac{N_T}{\mu d} \right) \end{aligned} \tag{22}$$

where N_T is the total number of queries issued in the first T rounds.

Theorem 7 *If Algorithm 2 is running with input $\kappa \in [0, 1]$, then for any ordering of T nodes on a graph, $\mathbf{f} \in \{-1, 1\}^T$, and $\|\boldsymbol{\alpha}\|^2 \leq C$, the cumulative regret satisfies*

$$\begin{aligned} R_T \leq\ & \min_{0<\epsilon<1} \{ \epsilon T_\epsilon + \lceil \frac{1}{\kappa} \rceil ! \left(2(\frac{8}{\epsilon^2})^{\frac{1}{k}} + e(\frac{4(2B\Phi(\mathbf{f}) + \epsilon)}{\epsilon^2})^{\frac{1}{k}} \right) \\ +\ & \left(\frac{16}{e\epsilon^2} + \frac{4(2B\Phi(\mathbf{f}) + \epsilon)}{\epsilon^2} \right) d \ln \left(1 + \frac{N_T}{\mu d} \right) \} \end{aligned} \tag{23}$$

Moreover, the number of queried nodes is upper bounded as $N_T \leq T^\kappa d \ln \left(1 + \frac{N_T}{\mu d} \right)$.

Proof. We have

$$\begin{aligned} & \Pr(Y_t \hat{\Delta}_t < 0) - \Pr(Y_t \Delta_t < 0) \\ \leq\ & \epsilon \{ |\Delta_t| < \epsilon \} + \Pr(\hat{\Delta}_t \Delta_t \leq 0, |\Delta_t| \geq \epsilon) \\ \leq\ & \epsilon \{ |\Delta_t| < \epsilon \} + \Pr(|\hat{\Delta}_t - \Delta_t| \geq \epsilon) \end{aligned} \tag{24}$$

Hence the cumulative regret can be bounded as follows:

$$R_T \leq \epsilon T_\epsilon + \sum_{t=1}^{T} \Pr(|\hat{\Delta}_t - \Delta_t| \geq \epsilon) \tag{25}$$

Using Lemmas 6 and 3 completes the proof of the regret bound. Finally, in order to derive a bound on the number of queried labels (label complexity), we have

$$N_T \leq \sum_{t:r_t > t^{-\kappa}} \frac{r_t}{t^{-\kappa}} \leq T^\kappa \sum_{t:r_t > t^{-\kappa}} r_t \leq T^\kappa d \ln\left(1 + \frac{N_T}{\mu d}\right) \tag{26}$$

□

Note that the label complexity is $O(dT^\kappa \log(T))$, which is smaller than $O(T)$ when κ is sufficiently small. Roughly speaking, the larger the value of κ, the more nodes the learner will query. One may argue that our regret bound depends on d, which is not desirable. However, rather than the case of vector-based selective sampling, where d could be larger than T, d is smaller than T (or n) in our case. It is worth noting that if we choose $d = T$, the label complexity becomes $O(T^{\kappa+1} \log(T))$, which implies that the learner will query all the nodes. This indicates that in order to make selective sampling really work, we need to choose $d < T$. In this sense, low-rank approximation of $\mathbf{L}^\dagger$ is preferred. On the other hand, since the regret bound is decreasing with κ, a larger value of κ is preferable for superior prediction performance. In other words, it needs to query more nodes to obtain better performance. Therefore, there is a trade-off between label complexity and prediction performance.

5. LOW-RANK APPROXIMATION

Finding the $\mathbf{M}$ given a graph kernel $\mathbf{L}^\dagger$ is not difficult. In fact, $\mathbf{M}$ can be calculated directly from $\mathbf{L}$. Recall that the eigen decomposition of $\mathbf{L}$ is $\mathbf{L} = \sum_{i=2}^{n} \sigma_i \mathbf{v}_i \mathbf{v}_i^\top$ with $\mathbf{v}_i \in \mathbb{R}^n$, we could choose $\mathbf{M}$ as follows

$$\mathbf{M} = \mathrm{diag}(\frac{1}{\sqrt{\sigma_2}}, \ldots, \frac{1}{\sqrt{\sigma_n}})[\mathbf{v}_2, \ldots, \mathbf{v}_n]^\top. \tag{27}$$

In this way, $\mathbf{L}^\dagger$ is reconstructed exactly, but the time complexity of our algorithms becomes $O(n^2)$, which is computationally expensive for large graphs.

In this paper, in order to make our algorithms as well as existing online learning algorithms scalable to large graphs, we propose to choose $\mathbf{M}$ as follows

$$\hat{\mathbf{M}} = \mathrm{diag}(\frac{1}{\sqrt{\sigma_2}}, \ldots, \frac{1}{\sqrt{\sigma_d}})[\mathbf{v}_2, \ldots, \mathbf{v}_d]^\top, \tag{28}$$

where $d \ll n$. Thus, $\mathbf{L}^\dagger$ is approximated by a low-rank matrix $\hat{\mathbf{M}}^\top \hat{\mathbf{M}}$ with rank d. And the time complexity of our online algorithms is $O(d^2) \ll O(n^2)$. In the sequel, we will analyze the impact of such low-rank approximation on our algorithms. Denote $\hat{\mathbf{L}} = \sum_{i=2}^{d} \sigma_i \mathbf{v}_i \mathbf{v}_i^\top$ and $\hat{\mathbf{L}}^\dagger = \hat{\mathbf{M}}^\top \hat{\mathbf{M}} = \sum_{i=2}^{d} \frac{1}{\sigma_i} \mathbf{v}_i \mathbf{v}_i^\top$. According to Eckart-Young-Mirsky theorem [14], $\hat{\mathbf{L}}$ is the best rank-d approximation of $\mathbf{L}$, while $\hat{\mathbf{L}}^\dagger$ is the best rank-d approximation of $\mathbf{L}^\dagger$.

Due to space limit, we only analyze the impact of low-rank approximation on OLLGC. The analysis for SSLGC is similar and therefore omitted. By taking a close look at the regret bound of OLLGC in Theorem 5, we can see that there are two terms depending on $\mathbf{M}$ (or $\mathbf{L}^\dagger$ or $\mathbf{L}$). One is $\sigma_2(\mathbf{L})$, the other is $\sum_{i=1}^{d} \log\left(1 + \frac{\lambda_i}{\mu}\right)$.

First, note that $\sigma_2(\mathbf{L})$ is the second smallest eigenvalue of $\mathbf{L}$. Based on the above definitions, the second smallest eigenvalue of $\hat{\mathbf{L}}$ is the same as that of $\mathbf{L}$ provided that $d \geq 2$. Hence, low-rank approximation does not introduce any approximation error in $\sigma_2(\mathbf{L})$ as long as $d \geq 2$.

Second, if we choose the exact $\mathbf{M}$ as in Eq. (27), then $d = n$, and $\lambda_i, i = 1 \ldots, n$ are the eigenvalues of $\sum_{t=1}^{T} \mathbf{m}_t \mathbf{m}_t^\top \mathbb{I}[i \in \mathcal{M}]$. Let us consider the simple case where $\mathcal{M} = \{1, 2, \ldots, T\}$. In this case, $\lambda_i, i = 1 \ldots, n$ are the eigenvalues of $\sum_{t=1}^{T} \mathbf{m}_t \mathbf{m}_t^\top$. Based on some linear algebra manipulations, it is easy to show that $\lambda_i, i = 1 \ldots, n$ are also the eigenvalues of $\sum_{i=2}^{n} \frac{1}{\sigma_i} \mathbf{v}_i \mathbf{v}_i^\top$, i.e., $\lambda_i = \frac{1}{\sigma_i}$ for $i = 2, \ldots, n$. If we choose the approximate $\hat{\mathbf{M}}$ as in Eq. (28), and suppose the eigenvalues of $\sum_{t=1}^{T} \hat{\mathbf{m}}_t \hat{\mathbf{m}}_t^\top$ are $\hat{\lambda}_i, i = 1, \ldots, d$. Again, we can show that $\hat{\lambda}_i, i = 1, \ldots, d$ are actually the top d largest eigenvalues of $\sum_{i=2}^{d} \frac{1}{\sigma_i} \mathbf{v}_i \mathbf{v}_i^\top$, i.e., $\lambda_i = \frac{1}{\sigma_i}$ for $i = 2, \ldots, d$. This implies that, under the condition that σ_i are sufficiently large for $i > d$, the approximate $\hat{\mathbf{M}}$ provides a good approximation for $\sum_{i=1}^{d} \log\left(1 + \frac{\lambda_i}{\mu}\right)$. For the general case of $\mathcal{M}$, the argument is similar but more involved.

The above arguments justify the validity of low-rank approximation for graph kernels.

6. EXPERIMENTAL RESULTS

In this section, we empirically evaluate the effectiveness of the proposed algorithms. All the experiments are performed on a PC with Intel Core i5 3.20G CPU and 48GB RAM and all algorithms in our experiments are implemented in *Matlab*.

6.1 Data Sets

We used four real-world graph data sets to evaluate the online learning and selective sampling algorithms.

Coauthor[2] is an undirected co-author graph data set extracted from the *DBLP* database in four areas: *machine learning*, *data mining*, *information retrieval* and *databases*. It contains a total of 1711 authors, each of which is represented by a node. The edge between each pair of authors is weighted by the number of papers they have co-authored. Each class contains about 400 authors.

Cora[3] contains 2708 scientific publications classified into one of seven classes: *Case Based*, *Genetic Algorithms*, *Neural Networks*, *Probabilistic Methods*, *Reinforcement Learning*, *Rule Learning* and *Theory*. The citation graph contains 5429 links.

IMDB[4] is an international organization whose objective is to provide useful and up-to-date movie information. We create a graph based on the co-actor relationship among 17046 movies from four genres: "Romance", "Action", "Animation" and "Thriller". Each genre is considered as a class.

PubMed[5] contains 19717 scientific publications from the *PubMed* database pertaining to diabetes classified into one of three classes. The citation network consists of 44338 links.

Some graphs in the above data sets are directed, and we simply use $\mathbf{S} \leftarrow \max(\mathbf{S}, \mathbf{S}^\top)$ to transform them into undirected graphs. Table 1 summarizes the characteristics of the data sets introduced above.

[3] http://www.cs.umd.edu/~sen/lbc-proj/data/cora.tgz
[4] http://www.imdb.com/
[5] http://www.cs.umd.edu/projects/linqs/projects/lbc/Pubmed-Diabetes.tgz

Table 1: Description of the data sets

Datasets	#nodes	#links	#classes
Coauthor	1,711	7.507	4
Cora	2,485	10,138	7
IMDB	17,046	993,528	4
PubMed	19,717	88,651	3

6.2 Evaluation Measures

We evaluated the performance of online learning and selective sampling with the use of three measures: (i) cumulative error rate, which reflects the prediction performance of online learning algorithms; (ii) number of queried labels, which reflects the label efficiency of an algorithm; and (iii) cumulative computational time, which measures the efficiency of online learning. Note that the smaller the above measures, the better the performance of an online learning algorithm.

6.3 Baselines and Parameter Settings

We compare the proposed algorithms with the Graph Perceptron Algorithm (GPA) [18]. The algorithms we studied and their parameter settings are summarized as follows.

Graph Perceptron Algorithm (GPA) [18]: This is the state-of-the-art first-order online learning algorithm on graphs. There is no required parameter for this algorithm. Note that the Perceptron algorithm is not affected by the step-size.

Online Learning with Local and Global Consistency (OLLGC): This is the proposed second-order online learning algorithm on graphs. The parameter μ is tuned by searching the grid $\{10^{-3}, 10^{-2}, \ldots, 10\}$ on a held-out random shuffle.

Selective Sampling with Local and Global Consistency (SSLGC): This is the proposed selective sampling algorithm on graphs. The parameter μ is tuned according to the grid $\{10^{-3}, 10^{-2}, \ldots, 10\}$ on a held-out random shuffle. In our experiments, we fix $\kappa = 0.4$ for all the data sets. We also study the impact of κ by setting it to $\{0.1, 0.2, \ldots, 1\}$.

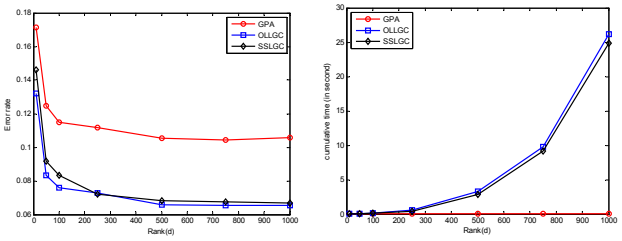
In order to compare these algorithms fairly, we randomly shuffle the ordering of nodes for each dataset. We repeat each experiment 20 times and calculate the average results.

The above algorithms are naturally designed for binary classification, while the data sets have more than two classes. In order to apply the algorithms to those data sets, we use one-vs-rest scheme, which is a standard technique for adapting binary classifiers to the multi-class scenario.

6.4 Study on Low-rank Approximation

We first study the impact of low-rank approximation on the performance of online learning algorithms. We try different ranks for approximation, and run all the algorithms. Because of the space limit, we used the *Cora* data set as a case study, because similar observations are obtained for the other data sets. Specifically, we changed the rank of the approximation using the grid $\{10, 50, 100, 250, 500, 750, 1000\}$. The results are shown in Figure 1.

It is evident that the higher the rank, the better the prediction performance because of a lower error rate. However, higher rank incurs higher computational cost, especially for second-order algorithms (OLLGC and SSLGC), because the time complexity of second-order algorithms is $O(d^2)$, where d is the rank. It implies that we need to obtain a trade-off



(a) Error Rate (b) Time

Figure 1: A case study of the impact of rank on the prediction performance (a) and time cost (b) in the *Cora* dataset.

between the predictive performance and the computational cost. Therefore, in the rest of our experiments, we chose $d = 100$, because the corresponding performance is good while the computational time is short. In fact, under different values of d, our algorithms are always better than GPA. Therefore, choosing $d = 100$ does not affect the fairness of the comparison in the rest of our experiments.

6.5 Results of Online Learning and Selective Sampling

The experimental results are shown in Table 2. For each data set, we executed paired t-tests of the error rate between the proposed algorithms and GPA at a 95% confidence interval. We found that the improvements of our algorithms over GPA are always significant. We also show the results with respect to the round of online learning in Figure 2. In all subfigures, the horizontal-axis represents the rounds of online learning, while the vertical-axis is the cumulative number of mistakes, queried nodes or cumulative time, averaged over 20 runs. Because of space limitations, we only show results on the *IMDB* and *PubMed* datasets.

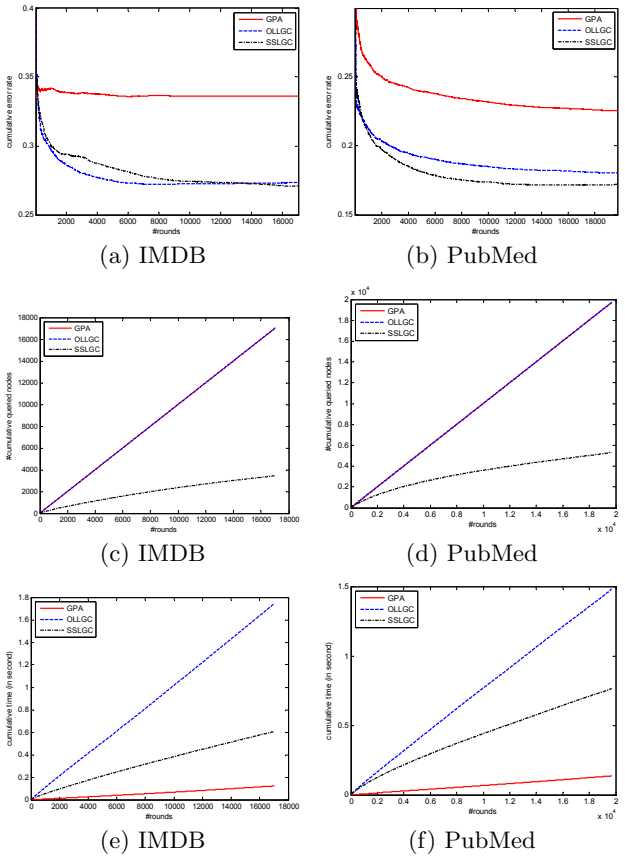
We can see that OLLGC outperforms GPA significantly on every data set. This is consistent with previous observations in vector-based online learning: second-order algorithms are generally better than first-order algorithms [20]. However, OLLGC requires more time than GPA. The reason is that the time complexity of GPA is $O(d)$, while the time complexity of OLLGC is $O(d^2)$. However, given the significant performance improvement of OLLGC over GPA, OLLGC is still very appealing.

SSLGC is better than GPA as well. Moreover, SSLGC achieves comparable results to OLLGC. Intuitively, SSLGC uses fewer labeled nodes than OLLGC, so that its performance should be no better than OLLGC. However, we can see that on *PubMed* dataset, SSLGC is even better than OLLGC. The reason is that the class distribution of *PubMed* is unbalanced. And when the data are unbalanced, passively querying the labels may be harmful, because the weight vector of the learner tends to be over-updated to fit the data from the majority class. That is why SSLGC could be better than OLLGC on the *PubMed* dataset.

Furthermore, it can be seen that SSLGC queried substantially fewer nodes while GPA and OLLGC queried every node. Although SSLGC queried much fewer nodes than OLLGC, their performances are comparable. This indicates that SSLGC is more label-efficient. Another advantage of label-efficiency is that SSLGC costs less time than OLLGC. The reason is obvious: once a node is queried, the model will

Table 2: A comparison of online learning and selective sampling algorithms on graphs in the four data sets. The smaller the value of the measure, the better the performance.

Algorithm	*Coauthor*			*Cora*		
	Error rate	#Queried nodes	Time (s)	Error rate	#Queried nodes	Time (s)
GPA	0.2326±0.0048	1711	0.0104±0.0012	0.1169±0.0022	2485	0.0135±0.0009
OLLGC	0.1838±0.0032	1711	0.1273±0.0087	0.0758±0.0013	2485	0.0929±0.0035
SSLGC	0.1854±0.0031	1275.30±21.91	0.1215±0.0181	0.0832±0.0019	1525.48±19.32	0.0821±0.0061
Algorithm	*IMDB*			*PubMed*		
	Error rate	#Queried nodes	Time (s)	Error rate	#Queried nodes	Time (s)
GPA	0.3362±0.0025	17046	0.1228±0.0048	0.2256±0.0025	19717	0.1363±0.0128
OLLGC	0.2735±0.0038	17046	1.7451±0.1141	0.1804±0.0014	19717	1.4813±0.1104
SSLGC	0.2709±0.0064	3453.55±91.32	0.6072±0.0153	0.1720±0.0050	5298.55±186.91	0.7646±0.0197



(a) IMDB (b) PubMed

(c) IMDB (d) PubMed

(e) IMDB (f) PubMed

Figure 2: Cumulative error rate (first row), cumulative number of queried nodes (second row) and cumulative time (third row) with respect to the online learning rounds on *IMDB* (first column); and *PubMed* (second column) datasets. The lower the curve, the better the performance.

be updated as long as a mistake is incurred. Since SSLGC queried fewer nodes, it has lower chance than OLLGC to update the model, which turns out to be computationally more efficient.

6.6 Study on the Impact of κ

Now we will study the impact of κ in our selective sampling algorithm. We will also compare it with random sampling. Generally speaking, the smaller the value of κ, the fewer the number of queried nodes. Specifically, we set κ to $\{0.1, 0.2, \ldots, 1\}$, and run SSLGC 20 times under each κ. We calculate the average ratio of queried nodes for different values of κ. Then, we test random sampling which is built on GPA. Rather than querying every node, the random sampling will query a node with probability $0 < p < 1$. In other words, for each node, the learner draw a value from a standard uniform distribution $U(0, 1)$. If the value is smaller than p, it queries the label. Otherwise, it does not. For fair comparison, we set p equal to the ratio of queried nodes in SSLGC. The comparison is shown in Figure 3.

We can observe that SSLGC is better than random sampling consistently under different ratio of queried nodes. This strengthens the advantage of SSLGC over random sampling. This is also why selective sampling is demanded for label effectiveness. It will actively query those nodes whose labels are uncertain. In contrast, random sampling just passively queries the nodes, without considering the informativeness of each node.

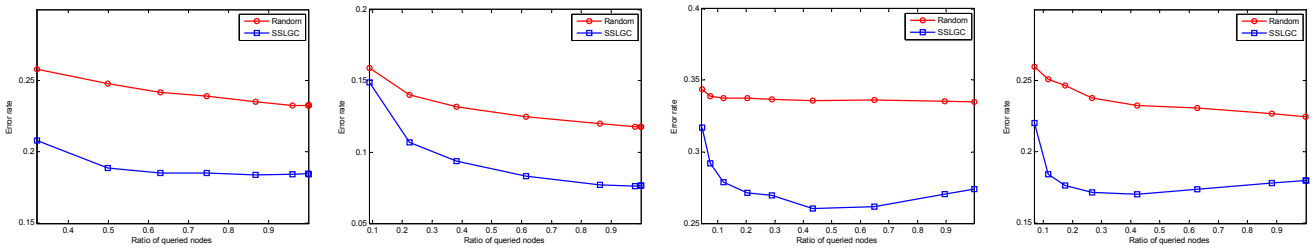
7. CONCLUSIONS

In this paper, we presented an online version of the well-known Learning with Local and Global Consistency method (OLLGC), and proved its regret bound in terms of the structural properties of a graph. Based on OLLGC, we presented Selective Sampling with Local and Global Consistency (SSLGC). We also derived a bound on the label complexity of SSLGC. Experiments show that OLLGC outperforms the state-of-the-art first-order algorithm substantially, and the selective sampling algorithm outperforms random sampling overwhelmingly given the same number of queried labels.

Note that in this paper, we studied transductive online learning and selective sampling on graphs. In our future work, we will study inductive online learning and selective sampling on graphs.

8. ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their helpful comments. This work was supported in part by U.S. National Science Foundation grants IIS-0905215, CNS-0931975, the U.S. Army Research Laboratory under Cooperative Agreement No. W911NF-09-2-0053 (NS-CTA), the U.S. Air Force Office of Scientific Research MURI award FA9550-08-1-0265, and MIAS, a DHS-IDS Center for Multimodal Information Access and Synthesis at UIUC.



(a) Coauthor (b) Cora (c) IMDB (d) PubMed

Figure 3: A comparison between selective sampling and random sampling with respect to different ratios of queried nodes on (a) *Coauthor*; (b) *Cora*; (c) *IMDB*; and (d) *PubMed* data sets. The lower the curve, the better the performance.

9. REFERENCES

[1] M. Bilgic and L. Getoor. Effective label acquisition for collective classification. In *KDD*, pages 43–51, 2008.

[2] M. Bilgic, L. Mihalkova, and L. Getoor. Active learning for networked data. In *International Conference on Machine Learning*, pages 79–86, 2010.

[3] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, 2004.

[4] L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *Ussr Computational Mathematics and Mathematical Physics*, 7(3):200–217, 1967.

[5] N. Cesa-Bianchi, A. Conconi, and C. Gentile. A second-order perceptron algorithm. *SIAM J. Comput.*, 34(3):640–668, 2005.

[6] N. Cesa-Bianchi, C. Gentile, and F. Orabona. Robust bounds for classification via selective sampling. In *ICML*, page 16, 2009.

[7] N. Cesa-Bianchi, C. Gentile, F. Vitale, and G. Zappella. Active learning on trees and graphs. In *Conference on Learning Theory*, pages 320–332, 2010.

[8] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Worst-case analysis of selective sampling for linear classification. *Journal of Machine Learning Research*, 7:1205–1230, 2006.

[9] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

[10] F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, February 1997.

[11] D. A. Cohn, L. E. Atlas, and R. E. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

[12] K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weight vectors. In *NIPS*, pages 414–422, 2009.

[13] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28(2-3):133–168, 1997.

[14] G. H. Golub and C. F. V. Loan. *Matrix computations (3rd ed.)*. Johns Hopkins University Press, Baltimore, MD, USA, 1996.

[15] Q. Gu and J. Han. Towards active learning on graphs: An error bound minimization approach. In *IEEE International Coference on Data Mining*, pages 882–887, 2012.

[16] M. Herbster and G. Lever. Predicting the labelling of a graph via minimum p-seminorm interpolation. In *COLT*, 2009.

[17] M. Herbster, G. Lever, and M. Pontil. Online prediction on large diameter graphs. In *NIPS*, pages 649–656, 2008.

[18] M. Herbster and M. Pontil. Prediction on a graph with a perceptron. In *NIPS*, pages 577–584, 2006.

[19] M. Herbster, M. Pontil, and L. Wainer. Online learning over graphs. In *ICML*, pages 305–312, 2005.

[20] S. C. H. Hoi, J. Wang, and P. Zhao. Exact soft confidence-weighted learning. In *ICML*, 2012.

[21] M. Ji and J. Han. A variance minimization criterion to active learning on graphs. *AISTATS*, pages 556–564, 2012.

[22] M. Ji, Y. Sun, M. Danilevsky, J. Han, and J. Gao. Graph regularized transductive classification on heterogeneous information networks. In *ECML/PKDD (1)*, pages 570–586, 2010.

[23] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1987.

[24] F. Orabona and N. Cesa-Bianchi. Better algorithms for selective sampling. In *ICML*, pages 433–440, 2011.

[25] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Reviews*, 65(6):386–408, November 1958.

[26] S. Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

[27] A. J. Smola and R. I. Kondor. Kernels and regularization on graphs. In *COLT*, pages 144–158, 2003.

[28] S. Tong and D. Koller. Support vector machine active learning with application sto text classification. In *International Conference on Machine Learning*, pages 999–1006, 2000.

[29] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf. Learning with local and global consistency. In *NIPS*, 2003.

[30] X. Zhu, Z. Ghahramani, and J. D. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *ICML*, pages 912–919, 2003.