GRAPH-BASED ACTIVE LEARNING: A NEW LOOK AT EXPECTED ERROR MINIMIZATION

Kwang-Sung Jun and Robert Nowak

Wisconsin Institutes for Discovery, University of Wisconsin-Madison
kjun@discovery.wisc.edu, rdnowak@wisc.edu

ABSTRACT

In graph-based active learning, algorithms based on expected error minimization (EEM) have been popular and yield good empirical performance. The exact computation of EEM optimally balances exploration and exploitation. In practice, however, EEM-based algorithms employ various approximations due to the computational hardness of exact EEM. This can result in a lack of either exploration or exploitation, which can negatively impact the effectiveness of active learning. We propose a new algorithm TSA (Two-Step Approximation) that balances between exploration and exploitation efficiently while enjoying the same computational complexity as existing approximations. Finally, we empirically show the value of balancing between exploration and exploitation in both toy and real-world datasets where our method outperforms several state-of-the-art approaches.

Index Terms— Machine learning, active learning, semi-supervised learning, graph-based learning, probabilistic model

1. INTRODUCTION

This paper studies the problem of the graph-based active learning. We are given a weighted undirected graph \( G = (N, E) \) with nodes \( N = \{1, \ldots, n\} \), edges \( E \), and weights \( w_{ij} \geq 0 \), \( \forall i \leq j \), that are 0 if there is no edge between \( i \) and \( j \). Each node \( i \in N \) has a label \( Y_i \in \{1, -1\} \). Let \( \ell_i \subseteq N \) be the initial labeled nodes. Initially, an algorithm knows the labels of \( \ell_1 \) only. At each time step \( t = 1, 2, \ldots \), an algorithm must perform

1. **PREDICT**: Make label prediction \( \hat{Y}_i \) on each unlabeled node \( i \notin \ell_t \). Let \( \hat{Y}_i := Y_i, \forall i \in \ell_t \). An algorithm suffers error rate \( \epsilon_t := \frac{1}{n} \sum_{i=1}^{n} 1 \{\hat{Y}_i \neq Y_i\} \), which is unknown to the algorithm.

2. **QUERY**: Select an unlabeled node \( q \) and query its label. Receive the label \( Y_q \). Update \( \ell_{t+1} = \ell_t \cup \{q\} \).

The goal is to achieve a low error rate while querying as few nodes as possible. The problem PREDICT is an instance of semi-supervised learning (LP) for which the seminal work of Zhu et al. [2] has been successful and de facto standard, which we call label propagation (LP). We thus focus on QUERY.

There are many examples where the data is given by or constructed as a graph. In document classification problems, two documents tend to be of the same topic when one cites the other or when they use the same keywords. A graph can be constructed based on such relations. The graph can then be used to infer a given document’s topic from the known topics of the other connected documents. More generally, a graph can be constructed based on known similarities or dissimilarities between unlabeled examples in any machine learning application. For example, hand-written digits can be recognized efficiently through graph-based learning algorithms [2]. In all these examples, the edge weights in the graph carry important information on how strongly two nodes (examples) are related, which can be used to make label predictions.

One popular approach to QUERY starts from an intuitive probabilistic model. Consider the following probabilistic model for the random variable \( Y \in \{1, -1\}^n \):

\[
P(Y = \mathbf{y}) = \frac{1}{Z} \exp \left( -\frac{\beta}{2} \sum_{i<j} w_{ij} (y_i - y_j)^2 \right),
\]

where \( Z \) is the normalization factor and \( \beta > 0 \) is a strength parameter. The model prefers labelings \( \mathbf{y} \in \{1, -1\}^n \) that vary smoothly across edges; i.e., larger weight \( w_{ij} \) implies higher likelihood of \( y_i = y_j \). We refer to the model above as binary Markov random field (BMRF). Note that BMRF would be equivalent to the Gaussian random field (GRF) if we relax the labels to belong to real values: \( Y \in \mathbb{R}^n \).

If the labels \( Y \) truly follow BMRF with known \( \beta \), given a set of observed labels of nodes \( \ell \subseteq N \), the expected error rate of a prediction strategy is well-defined; e.g., see [4]. Then, querying the node that minimizes the expected error in the next time step is a reasonable greedy strategy. We refer to the query strategy above as expected error minimization (EEM) principle. We precisely define EEM in Section 2.

EEM has been the main idea of many studies [2, 4, 5]. Define \( Y_{\ell} := \{Y_i\}_{i \in \ell} \). The challenge in EEM is to compute the posterior marginal of a node \( i \) given labeled nodes \( \ell \subseteq N \):

\[
P(Y_i \mid Y_{\ell} = \mathbf{y}_\ell).
\]

This is combinatorial; there is no known polynomial time algorithm for computing it, to our knowledge. Resolving

\footnote{Multi-class generalization is straightforward via the one-vs-the-rest reduction; see Section 4 for detail.}
such a computational issue in EEM has been an active area of research. Zhu et al. [3] apply a simple approximation to (2) by posterior mean of GRF, which we call ZLG. V-optimality (VOpt) [4] considers EEM under GRF instead of BMRF, which results in a closed-form solution. Σ-optimality (SOpt) [5] takes the same approach as VOpt, but based on a different error notion called survey error.

Each EEM-based algorithm has an undesirable behavior. Consider a linear chain of length 18 with edges between $i$ and $i + 1$ for all $1 \leq i \leq 17$ with weight 1; see Figure 1. Labels for node 1 and 11 are given as initial labels. We denote labeled nodes by ✓ where initial labels are in gray, the first two queries are in black, and the last two are in red. Symbols ✓/– indicate the predicted labels by LP after 4 queries. For the first query, an algorithm sees that there is at least one cut (edge connecting different labels) between node 1 and 11. ZLG drills into this region and spends its next four queries in nailing down the cut. Consequently, it does not query any node to the right side of node 11 and incurs large error; i.e., ZLG lacks exploration queries. In SOpt, the first two queries do include exploration query (node 16). Then, the next two queries include node 3 that does not reduce the error rate; node 8 would have reduced error. SOpt selects queries by which nodes have been labeled, ignoring what labels they have. In fact, this is the common characteristic of many graph-based active learning algorithms [6][7][8]. This is why SOpt is not able to optimize exploitation queries, which results in higher error than other methods as we show in toy experiments in Section 4. VOpt shares the same issue, so we omit it here. In contrast, the exact computation of EEM (row BMRF) balances between exploration and exploitation.

In this work, we propose a new algorithm TSA whose name comes from a two-step approximation to the posterior marginal (2). TSA improves upon both ZLG and SOpt without added computational complexity. The time complexity of TSA per query is $O(n^2)$, which is the same as ZLG and SOpt. Unlike ZLG and SOpt, TSA balances between exploration and exploitation. In a linear chain example in Figure 1, TSA finds the same queries as BMRF. We present TSA in Section 3 and empirical results in Section 4 where we observe that TSA outperforms baseline methods on several toy and real-world datasets.

### 2. EXPECTED ERROR MINIMIZATION (EEM)

Consider a probabilistic model over a $\mathbf{Y} \in \{1, -1\}^n$ such as (1). Given a set of labeled nodes $\ell \subseteq N$ with label $\mathbf{y}_\ell$, the optimal prediction is the Bayes decision rule

$$\hat{Y}_i(\mathbf{Y}_\ell = \mathbf{y}_\ell) := \arg\max_{y \in \{1, -1\}} P(Y_i = y | \mathbf{Y}_\ell = \mathbf{y}_\ell).$$

Note $\hat{Y}_i(\mathbf{Y}_\ell = \mathbf{y}_\ell) = Y_i$ for $i \in \ell$ trivially. We hereafter use $\hat{Y}_i$ and omit $(\mathbf{Y}_\ell = \mathbf{y}_\ell)$ when it is clear from the context.

Define the unlabeled nodes $\mathbf{u} := N \setminus \ell$. We are interested in measuring the expected error rate of the Bayes decision rule after querying $q \in \mathbf{u}$. Since we do not know $Y_q$ yet, we take expectation over $Y_q \in \{1, -1\}$ as well as $\{Y_i\}_{i \in \{u\} \cup \{q\}}$.

We define the expected error after knowing the label $Y_q$ as follows, which we call lookahead zero-one risk of node $q$:

$$R^{+q}(\mathbf{Y}_\ell = \mathbf{y}_\ell) := E_{\mathbf{Y}_q \mid Y_q}(\sum_{i \in \{u\} \cup \{q\}} 1\{\hat{Y}_i \neq Y_i\} | Y_q, \mathbf{Y}_\ell = \mathbf{y}_\ell) \quad (4)$$

where $\hat{Y}_i$ depends on $Y_q$ as well as $\mathbf{Y}_\ell$. We use $R^{+q}(\mathbf{y}_\ell)$ as a shortcut for $R^{+q}(\mathbf{Y}_\ell = \mathbf{y}_\ell)$.

The expected error minimization (EEM) principle is to choose the query that minimizes the lookahead zero-one risk:

$$\arg\min_{q \in N \setminus \ell} R^{+q}(\mathbf{y}_\ell).$$

Define $P_{\mathbf{y}_\ell}(\cdot) := P(\cdot | \mathbf{Y}_\ell = \mathbf{y}_\ell)$ and the zero-one risk

$$R(Y_q = y, \mathbf{y}_\ell) := E_{\mathbf{Y}_q \mid Y_q}(\sum_{i \in \{u\} \cup \{q\}} 1\{\hat{Y}_i \neq Y_i\} | Y_q = y, \mathbf{Y}_\ell = \mathbf{y}_\ell)
= \frac{1}{n} \sum_{i = 1}^n \left(1 - \max_{y' \in \{1, -1\}} P_{\mathbf{y}_\ell}(Y_i = y')\right).$$

Then,

$$R^{+q}(\mathbf{y}_\ell) = \sum_{y \in \{1, -1\}} R(Y_q = y, \mathbf{y}_\ell)P_{\mathbf{y}_\ell}(Y_q = y).$$

Notice that the key quantity is the posterior marginal distribution $P_{\mathbf{y}_\ell}(Y_q = y')$ in computing (6) and $P_{\mathbf{y}_\ell}(Y_q = y)$ in (7). An efficient computation of the posterior marginal would lead to an algorithm for Predict due to (1), and also to an algorithm for Query due to (3).

### 3. TWO-STEP APPROXIMATION OF MARGINAL

Consider BMRF defined in (1). Let $\mathbf{L}$ be the graph Laplacian defined by $L_{ij} := 1\{i = j\}(\sum_{k=1}^n w_{ik}) - w_{ij}$. We rewrite (1) compactly:
For ease of exposition, we let $\beta = 1$; one can obtain results for $\beta \neq 1$ by replacing $L$ with $\beta L$.

Suppose we have observed the labels of nodes $\ell$ as $y_\ell$. We propose a two-step approximation (TSA) to the posterior marginal distribution $P_{Y_{\ell}}(y_\ell)$, which leads to a new QUERY algorithm. The key lies in the following log probability ratio approximation: $\log P(Y_{\ell}=1|Y_{\ell}=y_\ell) \approx \log \mu(Y_{\ell}=1, Y_{\ell}=y_\ell)$, for some $\mu(\cdot)$. Define the sigmoid function $\sigma(z) := (1 + \exp(-z))^{-1}$. Then, it follows that

$$P(Y_k = 1 | Y_\ell = y_\ell) = \frac{P(Y_k = 1, Y_\ell = y_\ell)}{P(Y_k = 1, Y_\ell = y_\ell) + P(Y_k = -1, Y_\ell = y_\ell)} = \sigma(\log P(Y_k = 1, Y_\ell = y_\ell) - \log P(Y_k = -1, Y_\ell = y_\ell)) \approx \sigma(\log \mu(Y_k = 1, Y_\ell = y_\ell) - \log \mu(Y_k = -1, Y_\ell = y_\ell)).$$

We construct $\mu(Y_k = y_k, Y_\ell = y_\ell)$ as a two-step upper-bound on $P(Y_k = y_k, Y_\ell = y_\ell)$. Define $\pi := \cup \{\ell\}$, the set of unlabeled nodes except $k$. Let $L_{kk} := \sum L_{\ell\ell}$ and $g(\pi) := -\frac{1}{2} \sum L_{\ell\ell} \pi \pi + y_k L_{\ell\ell} \pi \pi + y_\ell L_{\ell\ell} \pi \pi$. We simplify $\log P(Y_k = y_k, Y_\ell = y_\ell) = -\log(Z) - \frac{1}{2} A - y_k L_{kk} y_\ell + \log \left( \sum_{\pi} \exp(g(\pi)) \right).$

Note that the last term is the log-sum-exp function that is similar to the max operator. This leads to our first upperbound:

$$\log \left( \sum_{\pi} \exp(g(\pi)) \right) \leq \max_{y_\ell \in \{1, -1\}} g(\pi) + |\pi| \log 2.$$

We now have an integer optimization problem, which is hard in general. We relax the domain of $y_\pi$ to real, which leads to our second upperbound:

$$\max_{y_\ell \in \{1, -1\}} g(\pi) \leq \max_{y_\ell \in \mathbb{R}} g(\pi).$$

We now have a concave quadratic maximization problem. Find the closed form solution (see the supplementary material [X]). Then, altogether,

$$\log P(Y_k = y_k, Y_\ell = y_\ell) \leq -\log(Z) - \frac{1}{2} A - y_k L_{kk} y_\ell +$$

$$\frac{1}{2} (y_k L_{kk} + y_\ell L_{\ell\ell}) L^{-1}_{\ell\ell}(L_{\ell\ell} y_k + L_{\ell\ell} y_\ell) + |\pi| \log 2$$

$$= \log \mu(Y_k = y_k, Y_\ell = y_\ell),$$

where $L_{\ell\ell}^{-1} = (L_{\ell\ell})^{-1}$. Let $f_k := \log \mu(Y_k = 1, Y_\ell = y_\ell) - \log \mu(Y_k = -1, Y_\ell = y_\ell)$ be the decision value of node $k$. We simplify $f_k$:

$$f_k = -2L_{kk} y_\ell + 2L_{kk} y_k L_{\ell\ell} L_{\ell\ell} y_\ell$$

for which we present a natural interpretation in our appendix. Finally, compute $P(Y_k = 1 | Y_\ell = y_\ell) \approx \sigma(f_k)$ for all $k \not\in \ell$ and perform EEM (5).

### Computing Marginals Altogether

Note that we need to compute $P(Y_k = 1 | Y_\ell = y_\ell)$ for every node $k \in u$, and the matrix inversion in (9) is costly. Denote by $\circ$ the Hadamard product and $[L^{-1}_{\ell\ell} k \ell] k \in u$ a vector whose $k$-th component has value $f_k$. Note that the one-step covariance update rule says that $L^{-1}_{\ell\ell} u u = L^{-1}_{\ell\ell} - (L^{-1}_{\ell\ell})_{:k} (L^{-1}_{\ell\ell})_{:k}$, where we assume that node $k$ is the largest index among $\ell$, without loss of generality. Using one-step covariance update rule, one can compute the marginals all at once with one matrix inversion (see the supplementary material [A, 2]):

$$[f_k] k \in u = -2 \left( \frac{1}{(L^{-1}_{\ell\ell})_{:k}} \right) \circ (L^{-1}_{\ell\ell} L_{\ell\ell} y_\ell).$$

Evaluating EEM (5) involves computing (10) $O(n)$ times. Since the matrix inversion in (10) can be performed in $O(n^2)$ using the one-step covariance update, the time complexity per query would be $O(n^3)$. However, one can use the “dangling node” trick presented in Appendix A of [3] to improve it to $O(n^2)$; see our Appendix for detail.

### Comparison to ZLG

Let $\sigma^{\text{Linear}}(z) := \frac{1}{2} (z + 1)$ that is valid over $z \in [-1, 1]$ only. ZLG performs a simple approximation: $[P(Y_k = 1 | Y_\ell = y_\ell)] k \approx \sigma^{\text{Linear}}(-L^{-1}_{\ell\ell} L_{\ell\ell} y_\ell),$

where we apply $\sigma^{\text{Linear}}$ elementwise. Input to $\sigma^{\text{Linear}}$ is always in $[-1, 1]$ due to the property of the harmonic function [2]. In TSA, $P(Y_k = 1 | Y_\ell = y_\ell) k \approx \sigma \left( 2 \cdot \frac{1}{(L^{-1}_{\ell\ell})_{:k}} \right) \circ (-L^{-1}_{\ell\ell} L_{\ell\ell} y_\ell).$

Both methods utilize $h := (-L^{-1}_{\ell\ell} L_{\ell\ell} y_\ell)$, which is the decision value of LP that is thresholded at 0 to make predictions (and notice both methods lead to the same prediction). Beside using a different sigmoid function, TSA further weights $h_k$ by $1/(L^{-1}_{\ell\ell})_{:k}$ where $(L^{-1}_{\ell\ell})_{:k}$ is always positive. $(L^{-1}_{\ell\ell})_{:k}$ can be interpreted as the variance of node $k$ in GRF context. The larger the variance of a node is, the closer its decision value to 0, and the closer the marginal probability to 1/2. Such a variance information is not utilized in ZLG.

A striking example is our introductory example in Figure[1] When the initial labels are given for node 1 and 11, the posterior marginal $P(Y_k = -1 | Y_\ell = y_\ell)$ for $k = 12, \ldots, 18$ under BMRF is (0.88, 0.79, 0.72, 0.67, 0.63, 0.60, 0.57) and under TSA is (0.88, 0.73, 0.66, 0.62, 0.60, 0.58, 0.57). Among node 12 to 18, node 16 has the smallest lookahead risk under both methods. However, under ZLG the marginals are (1.1, \ldots) for node 12 to 18, which results in all 0 lookahead risk. Similarly, after querying node 6 the segment from
node 2 to 5 have marginals (0,0, . . . 0) and 0 lookahead risk values. This explains why ZLG lacks exploration queries.

Another difference is that replacing occurrences of $L$ with $\beta L$ in ZLG results in no contribution of $\beta$ whereas in TSA there exists contribution of $\beta$; the smaller the $\beta$ is the closer the marginal probabilities to 1/2. We observed that $\beta$ changes the balance between exploration and exploitation. However, parameter tuning in active learning is hard in general; we leave it as a future work and use $\beta = 1$ in experiments.

4. EXPERIMENTS

Throughout the experiments, all methods start from one labeled node that is chosen uniformly at random. For every method, we break ties uniformly at random. Let $C$ be the number of classes. We handle multi-class case by instantiating one algorithm for each one-vs-the-rest (total $C$ runs). After computing each one-vs-the-rest marginal (binary), we compute the multi-class marginal distribution (now multinomial) by normalizing the binary marginals. Finally, the multi-class zero-one risk is a trivial extension of (6) from which we compute the EEM query.

Toy Data The first toy dataset is a linear chain with 15 nodes where each edge has weight 1. We choose an edge uniformly at random and assign positive label on one side and negative on the other side. We repeat the experiment 50 times where we assign new labels before each trial. We show the result in Fig. 2(a). There is no absolute winner. For very early time period, both VOpt and SOpt perform slightly better than the rest since they explore only — rough locations of the two positive boxes are discovered fast. On the other hand, ZLG incurs very low accuracy in the first half for the following two reasons: (i) before discovering a positive node, every node has the same lookahead risk and ZLG resorts to tie-breaking uniformly at random and (ii) after discovering the first positive node, ZLG drills down the exact boundary of it while completely not knowing the existence of the other positive box. In the end, however, ZLG becomes the best since it does not waste queries on exploration. TSA, our method, balances between exploration and exploitation and perform well on average.

Real-World Data We use exactly the same dataset

| Name    | $|N|$ | $|E|$ | The number of classes |
|---------|-----|-----|----------------------|
| DBLP    | 1711 | 2898 | 4                     |
| CORA    | 2485 | 5069 | 7                     |
| CITESEER | 2109 | 3665 | 6                     |

Table 1. Real-world dataset summary

Fig. 2. Experiment Results. Plots show accuracy vs. the number of queries. Error bars are in gray.
as \[5\] which is summarized in Table 1. DBLP is a coauthorship network, and both CORA and CITESEER are citation networks; see [5] for detail. We repeat the experiment 50 times and plot the results in Fig. [2(d-f)]. Overall, SOpt is better than ZLG for earlier time period, but ZLG is better for later time period (except in CITESEER), which we believe is due to the fact that ZLG lacks exploration queries and SOpt lacks exploitation queries, respectively. In contrast, TSA is as good as SOpt for earlier time period and as good as or even better than ZLG for later time period in all three datasets as TSA is able to balance between exploration and exploitation.

5. APPENDIX

Interpretation of the TSA Marginal

Define \( \hat{y}_\pi := -L^{-1}_{kk}L_{\pi\pi y} \) and \( \hat{y}_k := -L^{-1}_{kk} (L_{kk} L_{k\pi}) (y_k \hat{y}_\pi) \). Then, we can rewrite \( f_k \):

\[
 f_k = -2L_{kk} y_\pi - 2L_{k\pi} \pi y_\pi = 2L_{kk} \hat{y}_k .
\]

Recall \( P(Y_k = 1 \mid Y_\pi = y_\pi) \approx \sigma(f_k) \). The TSA marginal has the following “imputation” interpretation:

1. Given labels \( y_\pi \), compute the posterior mean of GRF \[2\] to impute the (soft) labels of \( \pi \): \( \hat{y}_\pi = -L^{-1}_{\pi\pi} L_{\pi\pi y} e_\pi \).

2. Based on the given labels \( y_\pi \) and the imputed labels \( \hat{y}_\pi \), compute the posterior mean of GRF of node \( k \): \( \hat{y}_k = -L^{-1}_{kk} (L_{kk} L_{k\pi}) (y_k \hat{y}_\pi) \).

3. Compute the TSA marginal \( P(Y_k = 1 \mid Y_\pi = y_\pi) \approx \sigma(2L_{kk} \hat{y}_k) \).

This reveals the close connection of TSA marginal to the posterior mean of GRF.

Fast Computation of the Lookahead Risk

The methodology here uses the same technique presented in Appendix A of [3]. We summarize the result here; see the supplementary material [A.3] for detail. Denote by \( f_k^{+}(q,y) \) the decision value of node \( k \) after labeling node \( q \) as \( y \) so that we compute the approximation \( P(Y_k = 1 \mid Y_q = y, Y_\pi = y_\pi) \approx (1 + \exp(-f_k^{+}(q,y)))^{-1} \). Assume for now that \( G := L_{uu}^{-1} \) and \( [f_k]_{k \in u} \) are computed from the previous iteration. The idea is to add in a “dongle” node that is attached to node \( q \) with weight \( w_q \) with label \( y \). Then, let \( w_q \to \infty \) to arrive at

\[
 [f_k^{+}(q,y)]_{k \in u} = 2 \left[ G_{kk} - \frac{G_{kq} G_{qk}}{G_{qq}} \right]^{-1} \left[ \frac{1}{2} \text{diag}(G) \right]_{k} [f_k]_{k} + \left( y - \frac{f_q}{2} \right) G_{qk} .
\]

One can verify that \( f_k^{+}(q,y) = \infty \) for \( y = 1 \) and \( -\infty \) for \( y = -1 \), correctly. Once we find the solution \( q' \) of EEM \( 5 \) for the current time step, then we can prepare \( L_{uu}^{-1} \left( q', y \right) \) for the next iteration using the one-step covariance update. This implies that the time complexity per query is \( O(n^2) \) for performing EEM. Note that the full matrix inversion with cost \( O(n^3) \) has to be performed for the first query.

Acknowledgements

This work was partially supported by the National Science Foundation grants CCF-1218189 and IIS-1447449 and by MURI grant ARMY W911NF-15-1-0479.

6. REFERENCES

[1] Xiaojin Zhu, “Semi-Supervised Learning Literature Survey,” Tech. Rep. 1530, Computer Sciences, University of Wisconsin-Madison, 2005.

[2] Xiaojin Zhu, Zoubin Ghahramani, and John Lafferty, “Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions,” in Proceedings of the International Conference on Machine Learning (ICML), 2003, pp. 912–919.

[3] Xiaojin Zhu, John Lafferty, and Zoubin Ghahramani, “Combining Active Learning and Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions,” in ICML workshop on The Continuum from Labeled to Unlabeled Data in Machine Learning and Data Mining, 2003, pp. 58–65.

[4] Ming Ji and Jiawei Han, “A Variance Minimization Criterion to Active Learning on Graphs,” in Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS), 2012, pp. 556–564.

[5] Yifei Ma, Roman Garnett, and Jeff Schneider, “Sigma-Optimality in Active Learning on Gaussian Random Fields,” in Advances in Neural Information Processing Systems (NIPS), 2013.

[6] Quanquan Gu and Jiawei Han, “Towards active learning on graphs: An error bound minimization approach,” in Proceedings - IEEE International Conference on Data Mining (ICDM), 2012, pp. 882–887.

[7] Akshay Gadde, Aamir Anis, and Antonio Ortega, “Active Semi-supervised Learning Using Sampling Theory for Graph Signals,” in Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 492–501.

[8] Andrew Guillory and Jeff A Bilmes, “Label Selection on Graphs,” in Advances in Neural Information Processing Systems (NIPS), pp. 691–699. 2009.
Supplementary Material

A. DERIVATIONS

A.1. The Solution of the Second Upper Bound

Recall that our second upperbound involves the following optimization problem:

\[
\max_{y_\pi \in \mathbb{R}^{|\tau|}} g(u) = \max_{y_\pi \in \mathbb{R}^{|\tau|}} \left( \frac{1}{2} y_\pi^T L_{m\pi} y_\pi + (y_k L_{k\pi} + y_\ell^T L_{\ell\pi}) y_\pi \right). 
\]

By equating its derivative to zero, \(-L_{m\pi} y_\pi - y_k L_{k\pi} - y_\ell^T L_{\ell\pi} = 0\). This leads to the solution \(y_\pi^* := -L_{m\pi}^{-1}(L_{m\pi} y_k - L_{m\pi} y_\ell) = -L_{m\pi}^{-1}(L_{m\pi} y_k - L_{m\pi} y_\ell)(y_k - y_\ell)\). Note the solution here is equivalent to the posterior mean of the GRF given labels \(y_\ell\) and \(y_k\). By plugging in \(y_\pi^*\) into the objective, we have

\[
\max_{y_\pi \in \mathbb{R}^{|\tau|}} - \left( \frac{1}{2} y_\pi^T L_{m\pi} y_\pi + (y_k L_{k\pi} + y_\ell^T L_{\ell\pi}) y_\pi \right) \\
= \max_{y_\pi \in \mathbb{R}^{|\tau|}} - \left( \frac{1}{2} y_\pi^T L_{m\pi} y_\pi + y_k L_{k\pi} + y_\ell^T L_{\ell\pi} \right) y_\pi \\
= \left(-\frac{1}{2} (y_k L_{k\pi} + y_\ell^T L_{\ell\pi}) + y_k L_{k\pi} + y_\ell^T L_{\ell\pi} \right) L_{m\pi}^{-1}(L_{m\pi} y_k + L_{m\pi} y_\ell) \\
= \frac{1}{2} (y_k L_{k\pi} + y_\ell^T L_{\ell\pi}) L_{m\pi}^{-1}(L_{m\pi} y_k + L_{m\pi} y_\ell). 
\]

A.2. Computing Marginals Altogether

Note that one needs to compute \(P(Y_k = 1 \mid Y_{\ell} = y_\ell)\) for every \(k \neq \ell\), which can be expensive. Recall that the set of the whole unlabeled nodes is \(u = \tau \cup \{k\}\). Let \(G := L_{u1}^{-1}\). By one-step covariance update rule, if \(k\) is the largest index among \(u\)

\[
\begin{pmatrix}
L_{m1}^{-1} \\
0 \\
0
\end{pmatrix} = G - \frac{G_k G_k}{G_{kk}} .
\]

If \(k\) is not the largest, the LHS has zeros for all elements of the column and row corresponding to node \(k\), and removing these zeros results in \(L_{m1}^{-1}\). Then,

\[
[f_k]_{k \in u} = \left[ -2L_{k\ell} y_\ell + 2L_{k\pi} L_{m\pi}^{-1} L_{m\ell} y_\ell \right]_k \\
\text{(a)} = \left[ -2L_{k\ell} y_\ell + 2L_{ku} \left( G - \frac{G_k G_k}{G_{kk}} \right) L_{u\ell} y_\ell \right]_k \\
\text{(b)} = \left[ -2I + 2L_{uu} G \right] L_{u\ell} y_\ell - 2 \left[ L_{ku} \frac{G_k G_k}{G_{kk}} L_{u\ell} y_\ell \right]_k \\
\text{(c)} = -2 \left[ \frac{L_{ku} G_k}{G_{kk}} \right] \circ \left[ G_k L_{u\ell} y_\ell \right]_k \\
\text{(d)} = -2 \left[ \frac{1}{G_{kk}} \right] \circ (G L_{u\ell} y_\ell) \\
\text{(e)} = -2 \left[ \frac{1}{(L_{uu}^{-1})_{kk}} \right] \circ (L_{uu}^{-1} L_{u\ell} y_\ell) ,
\]

where (a) is due to the one-step covariance update rule (note the change from \(\tau\) to \(u\)), (b) is due to \(L_{uu} G = I\), and (c) holds by observing that it is a product of two scalars.
A.3. Fast computation of the lookahead risk

We need to compute the lookahead risk for every unlabeled node \( i \in u = N \setminus \ell \). A naive approach requires us to explicitly compute one-step lookahead inversion for each \( i \in u \), which takes \( O(n^3) \) time for choosing a query. Let \( i \in u \) be the node for which we like to compute the lookahead risk and define \( \overline{u} := u \setminus \{i\} \). Let \( [f_k]_{k \in u} \) be the decision value vector after observing \( Y_\ell = y_\ell \). Let \( [f^{\ell+1}_k]_{k \in u} \) be the decision vector after observing \( Y_{\ell+1}(i) = y_{\ell+1}(i) \). Given \( [f_k]_k \), we wish to compute \( [f^{\ell+1}_k]_k \) without explicitly computing the one-step lookahead inversion \( L^{-1}_{uu} \).

Suppose we like to set \( Y_i \) to be \( y_i \in \{+1, -1\} \). The solution starts from adding a node named 0 with label \( y_0 \leftarrow y_i \) to the graph and add an edge between node 0 and \( i \) with weight \( w_0 \) while leaving \( Y_i \) unobserved. The new node is a “dougie” attached to node \( i \). Let \( L^+ := D^+ - W^+ \) be the graph Laplacian of the augmented graph. Denote by \( e_i \) the indicator vector with \( i \)-th component being 1. Then, we express the decision vector \( [f^{\ell+1}_k]_{k \in u} \) from the augmented graph as follows (then we will later take \( w_0 \) to infinity to get \( [f^{\ell+1}_k]_k \)):

\[
[f^{\ell+0}_k]_k = \left( \frac{1}{(L^+_{uu})^{-1}} \right)_{kk_k} \cdot \left( (L^+_{uu})^{-1} W^+_{uv} y_{\ell+1} \right)
\]

\[
= 2 \left( \frac{1}{(L^+_{uu})^{-1}} \right)_{kk_k} \cdot \left( (w_0 e_i e_i^T + D_{uu} - W_{uu})^{-1} (w_0 y_0 e_i + W_{uv} y_\ell) \right)
\]

\[
= 2 \left( \frac{1}{(L^+_{uu})^{-1}} \right)_{kk_k} \cdot \left( (w_0 e_i e_i^T + L_{uu})^{-1} (w_0 y_0 e_i + W_{uv} y_\ell) \right).
\]

Let \( G := L^{-1}_{uu} \). Applying the matrix inversion lemma, \( (w_0 e_i e_i^T + L_{uu})^{-1} = G - \frac{G_{ii} e_i e_i^T}{w_0 + G_{ii}} \). Then,

\[
\left( \frac{1}{(L^+_{uu})^{-1}} \right)_{kk_k} = \left[ \frac{1}{G_{kk} - \frac{G_{ii} e_i e_i^T}{w_0 + G_{ii}}} \right]_{kk_k} \rightarrow \left[ \frac{1}{G_{kk} - \frac{G_{ii} e_i e_i^T}{w_0 + G_{ii}}} \right]_{kk_k}
\]

and

\[
(w_0 e_i e_i^T + L_{uu})^{-1} (w_0 y_0 e_i + W_{uv} y_\ell) = \left( G - \frac{G_{ii} e_i e_i^T}{w_0 + G_{ii}} \right) (w_0 y_0 e_i + W_{uv} y_\ell)
\]

\[
= w_0 y_0 G_{ii} + G W_{uv} y_\ell - \frac{w_0 y_0 G_{ii} e_i e_i^T}{w_0 + G_{ii}} - \frac{G_{ii} e_i e_i^T}{w_0 + G_{ii}}
\]

\[
= \frac{y_0 G_{ii} + w_0 y_0 G_{ii} e_i e_i^T}{w_0 + G_{ii}} + G W_{uv} y_\ell - \frac{w_0 y_0 G_{ii} e_i e_i^T}{w_0 + G_{ii}} - \frac{G_{ii} e_i e_i^T}{w_0 + G_{ii}}
\]

\[
= \frac{y_0 G_{ii} e_i e_i^T}{w_0 + G_{ii}} + G W_{uv} y_\ell - \frac{G_{ii} e_i e_i^T}{w_0 + G_{ii}}
\]

\[
\rightarrow \frac{y_0 G_{ii} e_i e_i^T}{G_{ii}} + G W_{uv} y_\ell - \frac{G_{ii} e_i e_i^T}{G_{ii}}
\]

\[
= \frac{y_0 G_{ii} e_i e_i^T}{G_{ii}} + \frac{1}{2} \text{diag}(G) \circ [f_k]_k - \frac{1}{2} G_{ii} f_i.
\]

Therefore,

\[
[f^{\ell+0}_k]_k \rightarrow 2 \left( G_{kk} - \frac{G_{ii}^2}{G_{ii}} \right)^{-1} \circ \left( \frac{1}{2} \text{diag}(G) \circ [f_k]_k + \frac{y_0}{G_{ii}} - \frac{G_{ii} f_i}{2} \right) = [f^{\ell+1}_k]_k \in u,
\]

where one can show that \( f^{\ell+1}_i = +\infty \) if \( y_0 = 1 \) and \( -\infty \) if \( y_0 = -1 \).