Journal of Machine Learning Research 16 (2015) 2003-2022 Submitted 7/15; Published 9/15

Predicting a Switching Sequence of Graph Labelings

Mark Herbster M.HERBSTER@CS.UCL.AC.UK
Stephen Pasteris S.PASTERIS@CS.UCL.AC.UK
Massimiliano Pontil M.PONTIL@CS.UCL.AC.UK
Department of Computer Science
University College London
Gower Street, London WC1E 6BT, UK

Editor: Alex Gammerman and Vladimir Vovk

Abstract

We study the problem of predicting online the labeling of a graph. We consider a novel setting for this problem in which, in addition to observing vertices and labels on the graph, we also observe a sequence of just vertices on a second graph. A latent labeling of the second graph selects one of K labelings to be active on the first graph. We propose a polynomial time algorithm for online prediction in this setting and derive a mistake bound for the algorithm. The bound is controlled by the geometric cut of the observed and latent labelings, as well as the resistance diameters of the graphs. When specialized to multitask prediction and online switching problems the bound gives new and sharper results under certain conditions.

Keywords: online learning over graphs, kernel methods, matrix winnow, switching

1. Introduction

We consider the problem of learning online a set of K binary labelings of a graph. In a simple scenario this set of labelings corresponds to a switching sequence of labelings. Initially we focus on this setting before introducing our more general model. Consider the following game for predicting the labeling of a graph: *Nature* presents a graph; *nature* queries a vertex i_1; the *learner* predicts $\hat{y}_1 \in \{-1, 1\}$ as the label of the vertex; *nature* presents a label y_1; *nature* queries a vertex i_2; the *learner* predicts $\hat{y}_2$; and so forth. The learner's goal is to minimize the total number of mistakes $M = |\{t : \hat{y}_t \neq y_t\}|$. If nature is adversarial, the learner will always mispredict, but if nature is regular or simple, there is hope that a learner may make only a few mispredictions. Thus, a central goal of online learning is to design algorithms whose total mispredictions can be bounded relative to the complexity of nature's labeling.

To predict a *single* labeling of a graph, one may employ a kernel perceptron algorithm based on the graph Laplacian (Herbster and Pontil, 2006). This method achieves a bound of $M \leq \mathcal{O}(R\phi)$, where ϕ is the *cut* (the number of edges joining disagreeing labels) and R is the (resistance) diameter of the graph. Thus ϕ measures the complexity of the labeling and R is a structural parameter of the graph independent of the labeling. Such a bound is particularly appealing when the parameters are mildly dependent or independent of the number of vertices in the graph (see Herbster and Pontil, 2006, for a discussion).

In the switching setting, we now consider a sequence *colored* by K graph labelings with $S \geq K$ switches. We illustrate a switching sequence in Figure 1. In this color illustration

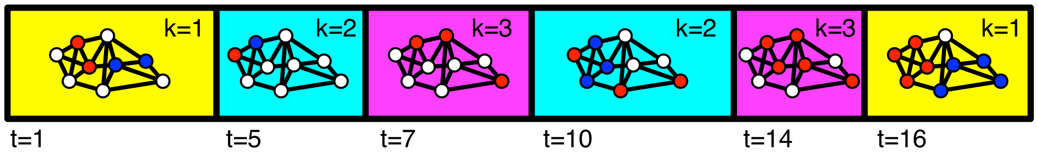

Figure 1: A switching sequence over 20 trials with K=3 graph labelings and S=5 switches.

there are $S = 5$ switches between $K = 3$ graph labelings. At each trial, a vertex of the graph is labeled according to one of the K binary functions. There are at most S trials at which the binary function currently in use is changed. In the specific example, the labeling 1 is used in trials 1–4 and 16–20, labeling 2 is used in trials 5–6 and 10–13, and labeling 3 is used in trials 7–9 and 14–15.

We will give an algorithm that achieves

$$M \leq \tilde{\mathcal{O}}\left(\left(S + R\sum_{k=1}^{K}\phi_k\right)K\log(n)\right), \tag{1}$$

where ϕ_k is the cut of the k-th binary labeling, n is the number of vertices in the graph, and the $\tilde{\mathcal{O}}(x)$ notation absorbs a polylogarithmic factor in x. Note that the term $R\sum_{k=1}^{K}\phi_k$ is the cost of learning the K binary labelings, given the information of which labeling is active on each trial. Since this information is not available to the learner, we pay a multiplicative term $K\log(n)$ and an additive term for the number of switches S. The particularly salient feature of this bound is that we pay the cost $R\sum_{k=1}^{K}\phi_k$ of learning all the binary labelings only once. This, and the fact that $S \geq K$, implies that the algorithm is maintaining an implicit *memory* of past graph labelings learned.

In the more general setting, the learner is given two graphs: an *observed* n-vertex graph $\mathcal{G}$ and a p-vertex *latent graph* $\mathcal{H}$. Hidden from the learner is a set $\{\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_K\}$ of K binary labelings of $\mathcal{G}$. On each trial one of these labelings is *active*, the learner receives a pair of vertices, $i \in \mathcal{G}$ and $j \in \mathcal{H}$, and the learner's aim is to predict the currently active binary label of vertex i. It is the unknown K-ary label of j that determines the active labeling of $\mathcal{G}$ and hence the current label of i. After making its prediction the learner receives only the current label of i. The learner never receives the label of j. Note that if the learner did in fact receive the label of j, the learning problem would separate into K independent graph labeling tasks. Thus the graph $\mathcal{H}$ is called latent because the vertex labels of this graph are never observed, although it controls which of the K labelings of $\mathcal{G}$ is active at each given trial.

We propose a polynomial time algorithm for predicting the labelings of the observed graph and we derive a mistake bound for this algorithm. The bound involves two additive terms, which measure the complexity of the K binary labelings, and the complexity of the latent labeling, respectively; as well as a multiplicative term of the order of $K\log(K(n+p))$. Returning to the switching example, the latent graph can be thought of as a "line" graph, where the sequence of vertices corresponds to the sequence of trials (although as we shall see in Section 6, for technical reasons we will need instead a binary support tree). The latent K-labeling function will then have a cut equal to S, the number of switches; and the bound (1) will be obtained as a special case of the general result described in this paper.

The paper is organized in the following manner. In Section 2, we comment about related work. In Section 3, we introduce the learning problem. In Section 4, we discuss the proposed learning algorithm. Section 5 presents our main result and details its proof. In Section 6, we illustrate our result in two specific examples and make final remarks.

2. Related Work

The problem of learning a labeling of a graph is a natural one in the online learning setting (Herbster et al., 2005; Herbster and Pontil, 2006), as well as a foundational technique for a variety of semi-supervised learning methods (Blum and Chawla, 2001; Kondor and Lafferty, 2002; Zhu et al., 2003). In the online setting, fast algorithms have been developed that operate on trees and path graphs (Herbster et al., 2008, 2009; Cesa-Bianchi et al., 2009, 2010; Vitale et al., 2011).

Our main application is to learning a switching sequence of graph labelings. Switching has been studied extensively in the online learning literature. The results divide largely into two directions: switching in the "experts" model (Herbster and Warmuth, 1998; Vovk, 1999; Bousquet and Warmuth, 2003; Gyorfi et al., 2005; Koolen and Rooij, 2008; Hazan and Seshadhri, 2009; Adamskiy et al., 2012; Cesa-Bianchi et al., 2012); and switching in online linear prediction model, see e.g. (Herbster and Warmuth, 2001; Kivinen et al., 2004; Cesa-Bianchi and Gentile, 2006). As we may view learning a graph labeling as learning a linear classifier based on a Laplacian kernel, our algorithm is directly comparable to these previous results. The implicit assumption of those switching techniques is that they learn a sequence of linear classifiers $w_1, w_2, \ldots$ and that this sequence is slowly changing over time, i.e, they are interested in predicting well when a *drifting* cost $\mathcal{O}(\sum_t \|w_t - w_{t+1}\|)$ is small. Our assumption is different: we consider that there exists a *small* set of K distinct classifiers, and we switch repeatedly between classifiers within this set. This setting is analogous to the setting proposed in an open problem by Freund (2000). Freund's challenge was to give an efficient algorithm in the expert advice model for the problem of switching repeatedly between a small set of experts within a larger set of experts. The problem was solved by Bousquet and Warmuth (2003) (see also Adamskiy et al., 2012). Those results, however, do not directly transfer to the graph labeling setting as the number of needed experts is 2^n for an n-vertex graph, and computing the marginal probabilities with a natural prior (i.e., an Ising distribution) on a graph even without switching is a well-known #P-complete problem (Goldberg and Jerrum, 2007).

An example of predicting in our more general setting applies to online multitask learning and is inspired by Cavallanti et al. (2010, Corollary 3). We adapt their model to our graph labeling set-up. Further related work includes (Dekel et al., 2007), which considered learning multiple tasks related through a joint loss function; and (Avishek et al., 2011), which generalized the usual setting to include negatively correlated tasks as well as positively correlated tasks. Rather than learning a group of interelated linear classifiers it is also natural to consider multi-task learning with expert advice. Two prominent results include those of Abernethy et al. (2007) and Adamskiy et al. (2012).

Our main technical debt is to the following four papers. Firstly, the mistake bound analysis of matrix winnow (Warmuth, 2007), which strongly informs the proof of our main result. Secondly, our analysis of using matrix winnow on graphs is inspired by the graph Laplacian construction in (Gentile et al., 2013). Thirdly, our first two techniques require a modification of the Laplacian to ensure strict positive definiteness, and here we used the simple construction from (Herbster and Pontil, 2006). Finally we use the *binary support tree* construction (Herbster et al., 2008) to model the trial sequence in the switching setting.

3. Problem

In this section, we present the problem under study. We begin by introducing some graph terminology.

We are given two undirected graphs, an n-vertex graph $\mathcal{G}$ and a p-vertex graph $\mathcal{H}$. We let $\mathcal{V}(\mathcal{G})$ and $\mathcal{V}(\mathcal{H})$ be the set of vertices in $\mathcal{G}$ and $\mathcal{H}$, respectively, and let $\mathbf{L}_{\mathcal{G}}$ and $\mathbf{L}_{\mathcal{H}}$ be the corresponding graph Laplacians. For every positive integer d, we define $\mathbb{N}_d = \{1, \ldots, d\}$, the set of integers from 1 and up to including d. Unless confusion arises, for simplicity we identify vertices by their indices. Indices $i, i', i_t \in \mathbb{N}_n$ will always be associated with vertices in $\mathcal{G}$, and indices $j, j', j_t \in \mathbb{N}_p$ will be associated with vertices in $\mathcal{H}$.

A *labeling* of a graph is a function which maps vertices on the graph to a set of labels. We define the *cut* induced by a labeling of a graph as the number of edges whose end vertices have different labels. Note that this definition is independent of the number of labels used. We will use the notation $\mathrm{cut}_{\mathcal{G}}(\mathbf{u})$ to denote the cut associated with the labeling $\mathbf{u}$ of graph $\mathcal{G}$. In particular if $\mathbf{u}$ is a binary labelling with label set $\{-1, 1\}$ then $\mathrm{cut}_{\mathcal{G}}(\mathbf{u}) = \frac{1}{4}\mathbf{u}^{\mathrm{T}}\mathbf{L}_{\mathcal{G}}\mathbf{u}$.

In the paper we refer to $\mathcal{G}$ as the *observed graph* since during the learning process we will observe *both* a vertex of $\mathcal{G}$ and a corresponding label, whereas we refer to $\mathcal{H}$ as the *latent graph* because we will *only* observe a vertex of $\mathcal{H}$ but *never* observe the corresponding label. As we will see, the latent graph provides side information which can guide the prediction tasks on the observed graph. The goal is to predict well the binary labels associated to vertices in $\mathcal{G}$ using sequential information of the form $(i_t, j_t, y_t) \in \mathcal{V}(\mathcal{G}) \times \mathcal{V}(\mathcal{H}) \times \{-1, 1\}$ for $t = 1, 2, \ldots, T$; the true label y_t is determined by using one of the K binary classifiers, and which of these is active at each trial is determined by a K-class classifier which acts on the latent graph $\mathcal{H}$. Specifically, we let $\boldsymbol{\omega}_1, \ldots \boldsymbol{\omega}_K$ be the binary classifiers (labelings) on graph $\mathcal{G}$. Each $\boldsymbol{\omega}_k$ is a function from $\mathcal{V}(\mathcal{G})$ to $\{-1, 1\}$. The latent labeling controls which of the K labelings of $\mathcal{G}$ is currently active and it is given by a function $\boldsymbol{\mu} : \mathcal{V}(\mathcal{H}) \to \mathbb{N}_K$. In the paper, when confusion does not arise, we simply regard the functions $\boldsymbol{\omega}_k$ as vectors in $\{-1, 1\}^n$ and $\boldsymbol{\mu}$ as a vector in $\{1, \ldots, K\}^p$.

We consider the following online learning game between nature and learner. The learner knows the graphs $\mathcal{G}$ and $\mathcal{H}$ from the outset but does not initially know the labelings $\boldsymbol{\omega}_1, \ldots, \boldsymbol{\omega}_K$, and as we already noted never observes the latent labeling $\boldsymbol{\mu}$. On trial t nature presents the learner with vertices $(i_t, j_t) \in \mathbb{N}_n \times \mathbb{N}_p$, the learner predicts a value $\hat{y}_t \in \{-1, 1\}$ and then the true label y_t is revealed to the learner. This label is computed by nature as $y_t = \omega_{\mu_{j_t}, i_t}$, that is the i_t-th component of the binary vector $\omega_{\mu_{j_t}}$. We define

Algorithm 1

```
Input: An $n$-vertex graph $\mathcal{G}$ and $p$-vertex graph $\mathcal{H}$.
Parameters: $K$, $\hat{\theta}$, $\eta$.
Initialization: $\boldsymbol{W}_0 \leftarrow \frac{\boldsymbol{I}}{K(n+p)}$, where $\boldsymbol{I}$ is the $(n+p)\times(n+p)$ identity matrix.
For $t = 1, \ldots, T$
  • Get pair of vertices $i_t, j_t \in \mathcal{V}(\mathcal{G}) \times \mathcal{V}(\mathcal{H})$.
  • Define the matrix $\boldsymbol{X}_t := \frac{1}{2}\boldsymbol{x}_t\boldsymbol{x}_t^T$, with $\boldsymbol{x}_t$ given by Equation (4).
  • Predict
        $\hat{y}_t = \begin{cases} 1 & \text{if } \operatorname{Tr}(\boldsymbol{W}_{t-1}\boldsymbol{X}_t) \geq \frac{K+1}{2K\hat{\theta}}, \\ -1 & \text{otherwise.} \end{cases}$
  • Receive label $y_t \in \{-1, 1\}$ and if $\hat{y}_t \neq y_t$ update
        $\boldsymbol{W}_t \leftarrow \exp\left(\log\left(\boldsymbol{W}_{t-1}\right) + \eta(y_t - \hat{y}_t)\boldsymbol{X}_t\right). \quad (2)$
```

the set of mistakes as $\mathcal{M} := \{t : \hat{y}_t \neq y_t\}$ and the number of mistakes $M := |\mathcal{M}|$. The aim of the learner is for M to be small.

Before presenting the learning algorithm we require some more notation. Given a matrix $\boldsymbol{A}$ we define $\boldsymbol{A}^+$, $\boldsymbol{A}^T$ and $\operatorname{Tr}(\boldsymbol{A})$ to be its pseudoinverse, transpose and trace respectively. We let $\mathbf{S}^d$ be the set of $d\times d$ symmetric matrices and let $\mathbf{S}^d_+$ and $\mathbf{S}^d_{++}$ be the subset of positive semidefinite and strictly positive definite matrices. Recall that the set of symmetric matrices $\mathbf{S}^d_+$ has the following partial ordering: for every $\boldsymbol{A}, \boldsymbol{B} \in \mathbf{S}^d_+$ we say that $\boldsymbol{A} \preceq \boldsymbol{B}$ if and only if $\boldsymbol{B} - \boldsymbol{A} \in \mathbf{S}^d_+$. Every real valued function f induces a spectral function $f : \mathbf{S}^d \to \mathbf{S}^d$ which is obtained by applying f to the eigenvalues of A. Specifically, if $\{\lambda_i, \boldsymbol{u}_i\}_{i=1}^d$ is an eigensystem of A, that is, $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_d$ are orthonormal vectors and λ_i are real numbers such that $\boldsymbol{A} = \sum_{i=1}^d \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T$, then we define $f(A) = \sum_{i=1}^d f(\lambda_i)\boldsymbol{u}_i\boldsymbol{u}_i^T$. Examples of spectral functions which we will use are $\exp(t)$, $\log(t)$ and $t \log t$. Note that the last two functions are well defined only on $\mathbf{S}^d_{++}$ and the last function can be extended to $\mathbf{S}^d_+$ as a limiting process. Finally, for vectors $\boldsymbol{\alpha} \in \mathbb{R}^n$ and $\boldsymbol{\beta} \in \mathbb{R}^p$ we define $[\boldsymbol{\alpha}, \boldsymbol{\beta}] \in \mathbb{R}^{n+p}$ to be the concatenation of $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, which we regard as a column vector. Hence $[\boldsymbol{\alpha}, \boldsymbol{\beta}]^T \left[\bar{\boldsymbol{\alpha}}, \bar{\boldsymbol{\beta}}\right] = \boldsymbol{\alpha}^T\bar{\boldsymbol{\alpha}} + \boldsymbol{\beta}^T\bar{\boldsymbol{\beta}}$.

4. The Algorithm

The learning algorithm we propose fits into the broad category of online matrix learning. At the core of the algorithm is an implicit spectral regularization, and we use a modification of matrix winnow (Warmuth, 2007) as our base algorithm.

As input the algorithm is given the graphs $\mathcal{G}$ and $\mathcal{H}$. The algorithm then depends on two input parameters, $K > 1$ and $\hat{\theta}$. The first parameter is the number labelings of the observed graph, which then determines the learning rate η. The second parameter $\hat{\theta}$ is a scaled threshold for the linear classifier. The parameter $\hat{\theta}$ is an upper bound on a measure of the complexity of the underlying learning problem, which is denoted by θ (cf. (6)).

We map a pair of vertices on the observed and latent graphs to a rank one positive semidefinite matrix, and use a linear classifier in the embedded space. Specifically, we map $(i_t, j_t) \in \mathcal{V}(\mathcal{G}) \times \mathcal{V}(\mathcal{H})$ to $\boldsymbol{X}_t \in \mathbf{S}_+^{n+p}$ given by the equation

$$\boldsymbol{X}_t := \frac{1}{2}\boldsymbol{x}_t\boldsymbol{x}_t^T \tag{3}$$

where

$$\boldsymbol{x}_t := \left[\frac{1}{\sqrt{\rho(\mathbf{G})}}(\boldsymbol{G}^{\frac{1}{2}})_{i_t}, \frac{1}{\sqrt{\rho(\mathbf{H})}}(\boldsymbol{H}^{\frac{1}{2}})_{j_t}\right], \tag{4}$$

matrices $\boldsymbol{G} \in \mathbf{S}_{++}^n$ and $\boldsymbol{H} \in \mathbf{S}_{++}^p$ are prescribed and we defined $\rho(\mathbf{G}) := \max_{i=1}^n \mathbf{G}_{ii}$ and $\rho(\mathbf{H}) := \max_{j=1}^p \mathbf{H}_{jj}$. The algorithm works for any such embeddings but the mistake bound presented in Theorem 1 below is obtained by choosing

$$\mathbf{G} = \mathbf{L}_{\mathcal{G}}^+ + R_{\mathcal{G}}\mathbf{1}\mathbf{1}^T \quad \text{and} \quad \mathbf{H} = \mathbf{L}_{\mathcal{H}}^+ + R_{\mathcal{H}}\mathbf{1}\mathbf{1}^T \tag{5}$$

where $\mathbf{1}$ denotes the vector $(1, \ldots, 1)^T$ and $R_{\mathcal{G}} = \max_{i=1}^n (\mathbf{L}_{\mathcal{G}}^+)_{ii}$ and $R_{\mathcal{H}} = \max_{j=1}^p (\mathbf{L}_{\mathcal{H}}^+)_{jj}$ are (essentially) the resistance diameters[1] of $\mathcal{G}$ and $\mathcal{H}$, respectively.

At each trial we predict by a linear threshold function in the embedded space, namely we predict positive if $\mathrm{Tr}\,(\boldsymbol{W}_{t-1}\boldsymbol{X}_t) > \frac{K+1}{2K\hat{\theta}}$ and negative otherwise, where $\boldsymbol{W}_t \in \mathbf{S}_+^{n+p}$ is a parameter matrix which is updated by the algorithm after each trial and initially set to a positive multiple of the identity matrix. Specifically, $\boldsymbol{W}_t$ is updated via Equation (2) only when a mistake is made. The worst case cost of an update is in the order of $(n+p)^3$ since this requires computing an eigensystem of an $(n+p) \times (n+p)$ matrix. However if the number of mistakes is much smaller than $n+p$ then the computation per trial can be substantially reduced because the weight matrix can be decomposed as the sum of a multiple of the identity matrix plus a low rank matrix (specifically the rank at trial t is equal to the current number of mistakes plus one). In this paper we are primarily concerned with the mistake bound and postpone further discussions on large scale implementations of the algorithm to a future occasion.

5. Main Result

In this section, we present our main result and give a detailed proof.

Theorem 1 *Let*

$$\theta = 8R_{\mathcal{G}}\sum_{k=1}^K \mathrm{cut}_{\mathcal{G}}(\boldsymbol{\omega}_k) + 4R_{\mathcal{H}}\mathrm{cut}_{\mathcal{H}}(\boldsymbol{\mu}) + 2\sum_{k=1}^K\left(\frac{1}{n}\sum_{i=1}^n \omega_{k,i}\right)^2 + 2\sum_{k=1}^K \frac{1}{p}\sum_{j=1}^p \mathcal{I}(\mu_j = k),$$

and let $\bar{c} := (5\log(5/3) - 2)^{-1} \leq 1.81$. The number of mistakes made by Algorithm 1 with $\theta \leq \hat{\theta}$ and learning rate $\eta := \frac{1}{2}\log\left(\frac{K+3}{K+1}\right)$ is upper bounded by

$$4K\bar{c}\left(2R_{\mathcal{G}}\sum_{k=1}^K \mathrm{cut}_{\mathcal{G}}(\boldsymbol{\omega}_k) + R_{\mathcal{H}}\mathrm{cut}_{\mathcal{H}}(\boldsymbol{\mu}) + K\right)\left(\log(K(n+p)) + \frac{\hat{\theta}}{\theta} - 1\right).$$

1. Specifically, $\max_{i=1}^n (\mathbf{L}_{\mathcal{G}}^+)_{ii}$ is a lower bound on the resistance diameter of $\mathcal{G}$, see (Herbster and Pontil, 2006, Eq. (9)).

To prepare for the proof we introduce some notation. The K-class labeling $\boldsymbol{\mu}$ induces K boolean labelings on $\mathcal{H}$, denoted by $\boldsymbol{\mu}_k \in \{0,1\}^p$, $k \in \mathbb{N}_K$, and is defined componentwise as $\mu_{k,j} = 1$ if $\mu_j = k$ and $\mu_{k,j} = 0$ otherwise. We also define, for every $k \in \mathbb{N}_K$,

$$\Phi_k := \boldsymbol{\mu}_k^T \boldsymbol{H}^{-1} \boldsymbol{\mu}_k, \text{ and } \Phi'_k := \boldsymbol{\omega}_k^T \boldsymbol{G}^{-1} \boldsymbol{\omega}_k.$$

For $i \in \mathbb{N}_n$, we let $\boldsymbol{e}_i$ be the i-th unit basis vector, that is, $e_{i,i'} = 0$ if $i \neq i'$ and $e_{i,i} = 1$.

We let

$$\boldsymbol{z}_k := \left[\sqrt{\rho(\mathbf{G})}\boldsymbol{G}^{-\frac{1}{2}}\boldsymbol{\omega}_k, \sqrt{\rho(\mathbf{H})}\boldsymbol{H}^{-\frac{1}{2}}\boldsymbol{\mu}_k\right]$$

and define the k-th embedded classifier associated with the k-th labelings as

$$\boldsymbol{Z}_k := \frac{\boldsymbol{z}_k \boldsymbol{z}_k^T}{\hat{\theta}},$$

with $\hat{\theta} \geq \theta$ where

$$\theta := \sum_{k=1}^{K} \|\boldsymbol{z}_k\|^2 = \rho(\mathbf{G}) \sum_{k=1}^{K} \boldsymbol{\omega}_k^T \boldsymbol{G}^{-1} \boldsymbol{\omega}_k + \rho(\mathbf{H}) \sum_{k=1}^{K} \boldsymbol{\mu}_k^T \boldsymbol{H}^{-1} \boldsymbol{\mu}_k . \tag{6}$$

Note that the representation of the k-th embedded classifier depends on the k-th labeling of the observed graph and the k-th "one versus all" labeling of the latent graph.

We have the following proposition.

Proposition 2 *For all $k \in \mathbb{N}_K$ and trials t it holds that*

$$\text{(i)} \qquad \operatorname{Tr}\left(\boldsymbol{Z}_k^T \boldsymbol{X}_t\right) = \frac{(\omega_{k,i_t} + \mu_{k,j_t})^2}{2\hat{\theta}}$$

$$\text{(ii)} \qquad \sum_{k=1}^{K} \operatorname{Tr}\left(\boldsymbol{Z}_k^T \boldsymbol{X}_t\right) = \frac{(K + 1 + 2y_t)}{2\hat{\theta}}$$

(iii) $\qquad \boldsymbol{X}_t$ *has eigenvalues in* $[0,1]$

$$\text{(iv)} \qquad \|\boldsymbol{z}_k\|^2 = \rho(\mathbf{H})\Phi_k + \rho(\mathbf{G})\Phi'_k$$

$$\text{(v)} \qquad \operatorname{Tr}\left(\boldsymbol{Z}_k \log\left(\boldsymbol{Z}_k\right)\right) < 0.$$

Proof (i): Note that $\operatorname{Tr}\left(\boldsymbol{Z}_k^T \boldsymbol{X}_t\right) = \frac{\operatorname{Tr}\left(\boldsymbol{z}_k \boldsymbol{z}_k^T (\boldsymbol{x}_t \boldsymbol{x}_t^T)\right)}{2\hat{\theta}} = \frac{(\boldsymbol{x}_t^T \boldsymbol{z}_k)^2}{2\hat{\theta}}$. The result then follows since $\boldsymbol{x}_t^T \boldsymbol{z}_k = \boldsymbol{e}_{i_t}^T \boldsymbol{\omega}_k + \boldsymbol{e}_{j_t}^T \boldsymbol{\mu}_k = \omega_{k,i_t} + \mu_{k,j_t}$.

(ii): If $k \neq \mu_{j_t}$ then $\mu_{k,j_t} = 0$ and by part (i) we have

$$\operatorname{Tr}\left(\boldsymbol{Z}_k^T \boldsymbol{X}_t\right) = \frac{1}{2\hat{\theta}}(\omega_{k,i_t} + \mu_{k,j_t})^2 = \frac{1}{2\hat{\theta}}(\omega_{k,i_t})^2 = \frac{1}{2\hat{\theta}}.$$

Suppose now that $k = \mu_{j_t}$. By the definition of y_t we have $y_t = \omega_{\mu_{j_t},i_t} = \omega_{k,i_t}$ so since $\mu_{k,j_t} = 1$ when $k = \mu_{j_t}$ we have, by part (i) that

$$\operatorname{Tr}\left(\boldsymbol{Z}_k^T \boldsymbol{X}_t\right) = \frac{1}{2\hat{\theta}}(\omega_{k,i_t} + \mu_{k,j_t})^2 = \frac{1}{2\hat{\theta}}(1 + y_t)^2 = \frac{1}{2\hat{\theta}}(1 + 2y_t + y_t^2) = \frac{(1 + y_t)}{\hat{\theta}}$$

as $y_t^2 = 1$. By summing the above over k we get the result.

(iii): Note that

$$\boldsymbol{X}_t := \frac{1}{2}\boldsymbol{x}_t\boldsymbol{x}_t^T = \frac{\|\boldsymbol{x}_t\|^2}{2}\frac{\boldsymbol{x}_t}{\|\boldsymbol{x}_t\|}\frac{\boldsymbol{x}_t^T}{\|\boldsymbol{x}_t\|}.$$

Thus $\boldsymbol{X}_t$ is a rank one positive semidefinite matrix and its only nonzero eigenvalue is $\|\boldsymbol{x}_t\|^2/2$. A direct computation then gives that $\|\boldsymbol{x}_t\|^2 \leq 2$. The result follows.

(iv): $\|\boldsymbol{z}_k\|^2 = \boldsymbol{z}_k^T\boldsymbol{z}_k = \rho(\mathbf{G})\boldsymbol{\omega}_k^T\boldsymbol{G}^{-1}\boldsymbol{\omega}_k + \rho(\mathbf{H})\boldsymbol{\mu}_k^T\boldsymbol{H}^{-1}\boldsymbol{\mu}_k = \rho(\mathbf{G})\Phi'_k + \rho(\mathbf{H})\Phi_k$.

(v): Note that $\boldsymbol{Z}_k$ is a positive semidefinite rank one matrix. Hence denoting with λ the non-trivial eigenvalue we have $\mathrm{Tr}\left(\boldsymbol{Z}_k \log\left(\boldsymbol{Z}_k\right)\right) = \lambda\log\lambda$. The result then follows if we show that $\lambda \in (0,1)$. To see this we write

$$\boldsymbol{Z}_k = \frac{\boldsymbol{z}_k\boldsymbol{z}_k^T}{\hat{\theta}} = \frac{\|\boldsymbol{z}_k\|^2}{\hat{\theta}}\frac{\boldsymbol{z}_k}{\|\boldsymbol{z}_k\|}\frac{\boldsymbol{z}_k^T}{\|\boldsymbol{z}_k\|}.$$

By definition $\theta = \sum_{k=1}^K \|\boldsymbol{z}_k\|^2$ so since $\theta \leq \hat{\theta}$ we have $\frac{\|\boldsymbol{z}_k\|^2}{\hat{\theta}} \leq 1$ as required.

■

We now define the quantum relative entropy, which plays a central role in the amortized analysis of the algorithm. We note that this technique was previously employed in online learning by Tsuda et al. (2005).

Definition 3 *The quantum relative entropy of symmetric positive semidefinite square matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ is*

$$\Delta(\boldsymbol{A},\boldsymbol{B}) := \mathrm{Tr}\left(\boldsymbol{A}\log\left(\boldsymbol{A}\right) - \boldsymbol{A}\log\left(\boldsymbol{B}\right) + \boldsymbol{B} - \boldsymbol{A}\right).$$

We will utilize the following lemmas.

Lemma 4 *For $t \in \mathcal{M}$ we have that*

$$\sum_{k=1}^K(\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_{t-1}) - \Delta(\boldsymbol{Z}_k,\boldsymbol{W}_t)) \geq \frac{c}{K\hat{\theta}}.$$

where $c := 5\log(5/3) - 2$.

Proof When $t \in \mathcal{M}$ we have, for all $k \in \mathbb{N}_K$, that

$$\begin{aligned}
&\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_{t-1}) - \Delta(\boldsymbol{Z}_k,\boldsymbol{W}_t)\\
&\quad= \mathrm{Tr}\left(\boldsymbol{Z}_k\log\left(\boldsymbol{W}_t\right) - \boldsymbol{Z}_k\log\left(\boldsymbol{W}_{t-1}\right)\right) + \mathrm{Tr}\left(\boldsymbol{W}_{t-1}\right) - \mathrm{Tr}\left(\boldsymbol{W}_t\right)\\
&\quad=\eta(y_t-\hat{y}_t)\,\mathrm{Tr}\left(\boldsymbol{Z}_k\boldsymbol{X}_t\right) + \mathrm{Tr}\left(\boldsymbol{W}_{t-1}\right) - \mathrm{Tr}\left(\exp\left(\log\left(\boldsymbol{W}_{t-1}\right) + \eta(y_t-\hat{y}_t)\boldsymbol{X}_t\right)\right) &(7)\\
&\quad\geq\eta(y_t-\hat{y}_t)\,\mathrm{Tr}\left(\boldsymbol{Z}_k\boldsymbol{X}_t\right) + \mathrm{Tr}\left(\boldsymbol{W}_{t-1}\right) - \mathrm{Tr}\left(\exp\left(\log\left(\boldsymbol{W}_{t-1}\right)\right)\exp\left(\eta(y_t-\hat{y}_t)\boldsymbol{X}_t\right)\right) &(8)\\
&\quad=\eta(y_t-\hat{y}_t)\,\mathrm{Tr}\left(\boldsymbol{Z}_k\boldsymbol{X}_t\right) + \mathrm{Tr}\left(\boldsymbol{W}_{t-1}(\boldsymbol{I} - \exp\left(\eta(y_t-\hat{y}_t)\boldsymbol{X}_t\right))\right)\\
&\quad\geq\eta(y_t-\hat{y}_t)\,\mathrm{Tr}\left(\boldsymbol{Z}_k\boldsymbol{X}_t\right) + (1-e^{\eta(y_t-\hat{y}_t)})\,\mathrm{Tr}\left(\boldsymbol{W}_{t-1}\boldsymbol{X}_t\right) &(9)
\end{aligned}$$

where Equation (7) comes from the algorithm's update of $\boldsymbol{W}_{t-1}$ (see Equation (2)), Equation (8) comes from Lemma A.8 with $\boldsymbol{A} := \log(\boldsymbol{W}_{t-1})$ and $\boldsymbol{B} := \eta(y_t - \hat{y}_t)\boldsymbol{X}_t$, and Equation (9) comes by first applying Lemma A.9 with $a := \eta(y_t - \hat{y}_t)$ and $\boldsymbol{A} := \boldsymbol{X}_t$ (using Proposition 2-(iii)), and then applying Lemma A.10 with $\boldsymbol{A} := \boldsymbol{W}_{t-1}$.
We hence have

$$\begin{aligned}
&\sum_{k=1}^{K}(\Delta(\boldsymbol{Z}_k, \boldsymbol{W}_{t-1}) - \Delta(\boldsymbol{Z}_k, \boldsymbol{W}_t)) \\
\geq& \eta(y_t - \hat{y}_t)\sum_{k=1}^{K}\operatorname{Tr}(\boldsymbol{Z}_k\boldsymbol{X}_t) + K(1 - e^{\eta(y_t-\hat{y}_t)})\operatorname{Tr}(\boldsymbol{W}_{t-1}\boldsymbol{X}_t) \\
=& \eta(y_t - \hat{y}_t)\frac{(K+1+2y_t)}{2\hat{\theta}} + K(1 - e^{\eta(y_t-\hat{y}_t)})\operatorname{Tr}(\boldsymbol{W}_{t-1}\boldsymbol{X}_t) \qquad (10)
\end{aligned}$$

where Equation (10) comes from Proposition 2-(ii).
Let ρ be the right hand side of Equation (10). Noting that $\eta := \frac{1}{2}\log\left(\frac{K+3}{K+1}\right)$ we have the following. When $y_t = 1$ and $\hat{y}_t = -1$ then $(1-e^{\eta(y_t-\hat{y}_t)})$ is negative and $\operatorname{Tr}(\boldsymbol{W}_{t-1}\boldsymbol{X}_t) < \frac{K+1}{2K\hat{\theta}}$ and thus

$$\begin{aligned}
\rho &\geq \eta(K+3)(\hat{\theta})^{-1} + \frac{K+1}{2}(1-e^{2\eta})(\hat{\theta})^{-1} \\
&= \frac{1}{2}\log\left(\frac{K+3}{K+1}\right)(K+3)(\hat{\theta})^{-1} + \frac{K+1}{2}\left(1 - \frac{K+3}{K+1}\right)(\hat{\theta})^{-1} \\
&= \left(\frac{1}{2}\log\left(\frac{K+3}{K+1}\right)(K+3) - 1\right)(\hat{\theta})^{-1} \\
&\geq \frac{c}{K\hat{\theta}}. \qquad (11)
\end{aligned}$$

Alternately, when $y_t = -1$ and $\hat{y}_t = 1$ then $(1 - e^{\eta(y_t-\hat{y}_t)})$ is positive and $\operatorname{Tr}(\boldsymbol{W}_{t-1}\boldsymbol{X}_t) \geq \frac{K+1}{2K\hat{\theta}}$ and thus

$$\begin{aligned}
\rho &\geq -\eta(K-1)(\hat{\theta})^{-1} + \frac{K+1}{2}(1-e^{-2\eta})(\hat{\theta})^{-1} \\
&= -\frac{1}{2}\log\left(\frac{K+3}{K+1}\right)(K-1)(\hat{\theta})^{-1} + \frac{K+1}{2}\left(1 - \frac{K+1}{K+3}\right)(\hat{\theta})^{-1} \\
&= \left(\frac{K+1}{K+3} - \frac{1}{2}\log\left(\frac{K+3}{K+1}\right)(K-1)\right)(\hat{\theta})^{-1} \\
&\geq \frac{c}{K\hat{\theta}}. \qquad (12)
\end{aligned}$$

The constant c in Equations (11) and (12) is derived from the following argument. For $K \geq 2$ the functions

$$K\left(\frac{1}{2}\log\left(\frac{K+3}{K+1}\right)(K+3) - 1\right) \qquad (13)$$

and

$$K\left(\frac{K+1}{K+3} - \frac{1}{2}\log\left(\frac{K+3}{K+1}\right)(K-1)\right) \qquad (14)$$

are monotonic increasing (see Lemmas A.12 and A.13) so

$$\begin{aligned} K\left[\frac{1}{2}\log\left(\frac{K+3}{K+1}\right)(K+3)-1\right] &\geq 2\left[\frac{1}{2}\log\left(\frac{2+3}{2+1}\right)(2+3)-1\right] \\ &= 5\log\left(\frac{5}{3}\right)-2=c \end{aligned}$$

and

$$\begin{aligned} K\left[\frac{K+1}{K+3}-\frac{1}{2}\log\left(\frac{K+3}{K+1}\right)(K-1)\right] &\geq 2\left[\frac{2+1}{2+3}-\frac{1}{2}\log\left(\frac{2+3}{2+1}\right)(2-1)\right] \\ &= \frac{6}{5}-\log\left(\frac{5}{3}\right)>c. \end{aligned}$$

Hence $\frac{1}{2}\log\left(\frac{K+3}{K+1}\right)(K+3)-1\geq\frac{c}{K}$ and $\frac{K+1}{K+3}-\frac{1}{2}\log\left(\frac{K+3}{K+1}\right)(K-1)\geq\frac{c}{K}$. ■

Lemma 5 *It holds that* $\sum_{k=1}^{K}\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_0)\geq|\mathcal{M}|\frac{c}{K\hat{\theta}}$.

Proof We have

$$\begin{aligned} \sum_{k=1}^{K}\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_0) &\geq \sum_{k=1}^{K}(\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_0)-\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_T)) \\ &= \sum_{k=1}^{K}\sum_{t=1}^{T}(\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_{t-1})-\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_t)) \\ &= \sum_{t=1}^{T}\sum_{k=1}^{K}(\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_{t-1})-\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_t)) \\ &= \sum_{t\in\mathcal{M}}\sum_{k=1}^{K}(\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_{t-1})-\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_t))+\sum_{t\notin\mathcal{M}}\sum_{k=1}^{K}(\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_{t-1})-\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_t)) \\ &\geq |\mathcal{M}|\frac{c}{K\hat{\theta}} \qquad (15) \end{aligned}$$

where Equation (15) comes from Lemma 4 and the fact that on a trial $t\notin\mathcal{M}$ we have $\boldsymbol{W}_t=\boldsymbol{W}_{t-1}$ and hence $\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_{t-1})=\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_t)$ for all $k\in\mathbb{N}_K$. ■

Lemma 6 *It holds that* $\sum_{k=1}^{K}\Delta(\boldsymbol{Z}_k,\boldsymbol{W}_0)\leq\frac{\theta}{\hat{\theta}}\log(K(n+p))+\left(1-\frac{\theta}{\hat{\theta}}\right)$.

Proof of Lemma 6 Recall that $\boldsymbol{W}_0 = \frac{\boldsymbol{I}}{K(n+p)}$, where $\boldsymbol{I}$ is the $(n+p) \times (n+p)$ identity matrix. We observe that

$$\begin{aligned}
\sum_{k=1}^{K} \Delta(\boldsymbol{Z}_k, \boldsymbol{W}_0) &= \sum_{k=1}^{K} (\mathrm{Tr}\,(\boldsymbol{Z}_k \log(\boldsymbol{Z}_k)) - \mathrm{Tr}\,(\boldsymbol{Z}_k \log(\boldsymbol{W}_0)) + \mathrm{Tr}\,(\boldsymbol{W}_0) - \mathrm{Tr}\,(\boldsymbol{Z}_k)) \\
&\leq -\sum_{k=1}^{K} \mathrm{Tr}\,(\boldsymbol{Z}_k \log(\boldsymbol{W}_0)) + \sum_{k=1}^{K} \mathrm{Tr}\,(\boldsymbol{W}_0) - \sum_{k=1}^{K} \mathrm{Tr}\,(\boldsymbol{Z}_k) \qquad (16) \\
&= -\sum_{k=1}^{K} \mathrm{Tr}\left(\boldsymbol{Z}_k \log\left(\frac{\boldsymbol{I}}{K(n+p)}\right)\right) + \sum_{k=1}^{K} \mathrm{Tr}\left(\frac{\boldsymbol{I}}{K(n+p)}\right) - \sum_{k=1}^{K} \mathrm{Tr}\,(\boldsymbol{Z}_k) \\
&= \log(K(n+p)) \sum_{k=1}^{K} \mathrm{Tr}\,(\boldsymbol{Z}_k) + \frac{1}{K(n+p)} \sum_{k=1}^{K} \mathrm{Tr}\,(\boldsymbol{I}) - \sum_{k=1}^{K} \mathrm{Tr}\,(\boldsymbol{Z}_k) \\
&= \log(K(n+p)) \sum_{k=1}^{K} \mathrm{Tr}\,(\boldsymbol{Z}_k) + 1 - \sum_{k=1}^{K} \mathrm{Tr}\,(\boldsymbol{Z}_k) \\
&= 1 + (\log(K(n+p) - 1)) \sum_{k=1}^{K} \mathrm{Tr}\,(\boldsymbol{Z}_k) \\
&= 1 + (\log(K(n+p) - 1)) \sum_{k=1}^{K} \mathrm{Tr}\left(\frac{\boldsymbol{z}_k \boldsymbol{z}_k^T}{\hat{\theta}}\right) \\
&= 1 + (\log(K(n+p) - 1)) \frac{1}{\hat{\theta}} \sum_{k=1}^{K} \boldsymbol{z}_k^T \boldsymbol{z}_k \\
&= 1 + \left(\log(K(n+p) - 1)\,\frac{\theta}{\hat{\theta}}\right) \qquad (17) \\
&= \frac{\theta}{\hat{\theta}} \log(K(n+p)) + \left(1 - \frac{\theta}{\hat{\theta}}\right)
\end{aligned}$$

where Equation (16) comes from Proposition 2-(v) and Equation (17) comes from the definition of θ. ■

We are now ready to prove our main result.

Proof of Theorem 1 Combining Lemmas 6 and 5 we have

$$|\mathcal{M}| \frac{c}{K\hat{\theta}} \leq \sum_{k=1}^{K} \Delta(\boldsymbol{Z}_k, \boldsymbol{W}_0) \leq \frac{\theta}{\hat{\theta}} \log(K(n+p)) + \left(1 - \frac{\theta}{\hat{\theta}}\right)$$

which gives

$$\begin{aligned}|\mathcal{M}| &\leq \frac{K\hat{\theta}}{c}\frac{\theta}{\hat{\theta}}\log\left(K(n+p)\right) + \frac{K\hat{\theta}}{c}\left(1-\frac{\theta}{\hat{\theta}}\right)\\ &= \frac{K\theta}{c}\log\left(K(n+p)\right) + \frac{K\theta}{c}\frac{\hat{\theta}}{\theta}\left(1-\frac{\theta}{\hat{\theta}}\right)\\ &= \frac{K\theta}{c}\left(\log\left(K(n+p)\right) + \left(\frac{\hat{\theta}}{\theta}-1\right)\right).\end{aligned}$$

Finally we compute θ by choosing the matrices $\mathbf{H}$ and $\mathbf{G}$ as per Equation (5). A direct computation gives, for any vector $\boldsymbol{\omega} \in \mathbb{R}^n$, that

$$\boldsymbol{\omega}^{\scriptscriptstyle T}\boldsymbol{G}^{-1}\boldsymbol{\omega} = \boldsymbol{\omega}^{\scriptscriptstyle T}(\mathbf{L}_{\mathcal{G}}^{+} + R_{\mathcal{G}}\mathbf{1}\mathbf{1}^{\scriptscriptstyle T})^{-1}\boldsymbol{\omega} = \boldsymbol{\omega}^{\scriptscriptstyle T}\mathbf{L}_{\mathcal{G}}\boldsymbol{\omega} + \frac{1}{R_{\mathcal{G}}}\left(\frac{1}{n}\sum_{i=1}^{n}\omega_i\right)^2$$

and, likewise, for any vector $\boldsymbol{\mu} \in \mathbb{R}^p$

$$\boldsymbol{\mu}^{\scriptscriptstyle T}\boldsymbol{H}^{-1}\boldsymbol{\mu} = \boldsymbol{\mu}^{\scriptscriptstyle T}(\mathbf{L}_{\mathcal{H}}^{+} + R_{\mathcal{H}}\mathbf{1}\mathbf{1}^{\scriptscriptstyle T})^{-1}\boldsymbol{\mu} = \boldsymbol{\mu}^{\scriptscriptstyle T}\mathbf{L}_{\mathcal{H}}\boldsymbol{\mu} + \frac{1}{R_{\mathcal{H}}}\left(\frac{1}{p}\sum_{j=1}^{p}\mu_j\right)^2.$$

For the observed labelings we have $\boldsymbol{\omega}_k^{\scriptscriptstyle T}\mathbf{L}_{\mathcal{G}}\boldsymbol{\omega}_k = 4\mathrm{cut}(\boldsymbol{\omega}_k)$. Using this and $\rho(\mathbf{G}) = 2R_{\mathcal{G}}$, a direct computation gives

$$\begin{aligned}\rho(\mathbf{G})\sum_{k=1}^{K}\boldsymbol{\omega}_k^{\scriptscriptstyle T}\boldsymbol{G}^{-1}\boldsymbol{\omega}_k &= 2R_{\mathcal{G}}\left(4\sum_{k=1}^{K}\mathrm{cut}(\boldsymbol{\omega}_k) + \frac{1}{R_{\mathcal{G}}}\sum_{k=1}^{K}\left(\frac{1}{n}\sum_{i=1}^{n}\omega_{k,i}\right)^2\right)\\ &\leq 8R_{\mathcal{G}}\sum_{k=1}^{K}\mathrm{cut}(\boldsymbol{\omega}_k) + 2K.\end{aligned}$$

For the latent labeling we have $\sum_{k=1}^{K}\boldsymbol{\mu}_k^{\scriptscriptstyle T}\mathbf{L}_{\mathcal{H}}\boldsymbol{\mu}_k = 2\mathrm{cut}(\boldsymbol{\mu})$. Using this and $\rho(\mathbf{H}) = 2R_{\mathcal{H}}$, we obtain

$$\begin{aligned}\rho(\mathbf{H})\sum_{k=1}^{K}\boldsymbol{\mu}_k^{\scriptscriptstyle T}\boldsymbol{H}^{-1}\boldsymbol{\mu}_k &= 2R_{\mathcal{H}}\left(2\mathrm{cut}(\boldsymbol{\mu}) + \frac{1}{R_{\mathcal{H}}}\sum_{k=1}^{K}\left(\frac{1}{n}\sum_{i=1}^{n}\mu_{k,i}\right)^2\right)\\ &\leq 4R_{\mathcal{H}}\mathrm{cut}(\boldsymbol{\mu}) + 2K.\end{aligned}$$

We conclude that

$$\begin{aligned}\theta = \sum_{k=1}^{K}\|\boldsymbol{z}_k\|^2 &= \rho(\mathbf{G})\sum_{k=1}^{K}\boldsymbol{\omega}_k^{\scriptscriptstyle T}\boldsymbol{G}^{-1}\boldsymbol{\omega}_k + \rho(\mathbf{H})\sum_{k=1}^{K}\boldsymbol{\mu}_k^{\scriptscriptstyle T}\boldsymbol{H}^{-1}\boldsymbol{\mu}_k\\ &\leq 4\left(2R_{\mathcal{G}}\sum_{k=1}^{K}\mathrm{cut}(\boldsymbol{\omega}_k) + R_{\mathcal{H}}\mathrm{cut}(\boldsymbol{\mu}) + K\right).\end{aligned}$$

The result now follows by substituting the last inequality in the mistake bound. ■

6. Discussion

In this section, we consider two special cases of the problem studied in this paper and make final remarks. We tailor Theorem 1 to these cases and then compare to similar mistake bounds available in the literature.

6.1 Uniform Multitask Prediction

In the uniform multitask problem we suppose that we have p tasks corresponding to predicting the binary labeling of a graph. We assume that the tasks are interrelated so that only $K \ll p$ graph labelings are needed. To solve this problem we assume each task is given a number in $\{1, \ldots, p\}$. Each task number denotes a unique vertex in the latent graph which is a p-vertex clique. Applying the bound of Theorem 1 gives

$$M \leq \mathcal{O}\left(\left(\sum_{k=1}^{K} \mathrm{cut}_{\mathcal{G}}(\boldsymbol{\omega}_k) R_{\mathcal{G}} + p\right) K \log(K(n+p))\right).$$

This follows immediately from the fact that the clique has resistance diameter $\mathcal{O}(\frac{1}{p})$ and the cut of a K-"coloring" is $\mathcal{O}(p^2)$.

In (Cavallanti et al., 2010), a broad range of results are given for online multi-task learning in generic reproducing kernel Hilbert spaces. We apply their Corollary 3 to our problem with the kernel $\mathbf{G}^{-1} := \mathbf{L}_{\mathcal{G}}^{+} + R_{\mathcal{G}}\mathbf{1}\mathbf{1}^T$. In their setting there is no parameter K and instead they learn p distinct graph labelings, and thus obtain

$$M \leq \mathcal{O}\left(\frac{1}{p}\left(\sum_{k=1}^{p} \mathrm{cut}_{\mathcal{G}}(\boldsymbol{\omega}_k) + \sum_{i<j}^{p} (\boldsymbol{\omega}_i - \boldsymbol{\omega}_j)^T \mathbf{G} (\boldsymbol{\omega}_i - \boldsymbol{\omega}_j)\right) R_{\mathcal{G}}\right).$$

This is small when each of the p binary labelings are near one another in the norm induced by the Laplacian. This is distinct from our bound where we pay a fixed price for each of the p tasks of $K \log(K(n+p))$. Thus our bound is stronger when $K \ll p$ and the averaged squared norm between labelings of the p tasks is larger than $K \log(K(n+p))$.

6.2 Switching

We now consider the case where we have a switching sequence of graph labelings with S switches between K labelings. We sketch a proof of the bound announced in the introduction (cf. Equation (1)), namely

$$M \leq \tilde{\mathcal{O}}\left(\left(S + R_{\mathcal{G}} \sum_{k=1}^{K} \mathrm{cut}_{\mathcal{G}}(\boldsymbol{\omega}_k)\right) K \log(n)\right),$$

where the $\tilde{\mathcal{O}}(x)$ notation absorbs a polylogarithmic factor in x. Notice that the apparently natural structure for the proof would be to choose a latent graph which is a "line" with T vertices, where the linear ordering of the vertices reflects the linear trial sequence. Unfortunately, the resistance diameter of this line graph would then be equal to T which would make the bound vacuous. We overcome this difficulty by borrowing a trick from (Herbster et al., 2008) and we instead use a binary tree with T leaves and thus a resistance diameter of $2\log_2 T$. We assume that for each trial we receive a label of a leaf along the natural linear ordering of the leaves. If ϕ is the cut along the leaves such a labeling may be extended to a labeling of the complete binary tree in a way that the cut increases by no more than a factor of $\log_2 T$. This extension works by choosing the label of the parent of each vertex to be consistent with the label of either of its children. The result follows since the labeling on each successive "level" of the tree down to the root is now a subsequence of the previous labeling, and the cut of a subsequence can only decrease. Hence with $\log_2 T$ levels the cut increases by no more than a logarithmic factor. A second insight is that we do not actually need a tree with T leaves, we in fact only need M leaves corresponding to when the algorithm incurs a mistake, hence,

$$M \leq \mathcal{O}\left(\left(S(\log(M))^2 + R_{\mathcal{G}}\sum_{k=1}^{K}\mathrm{cut}_{\mathcal{G}}(\boldsymbol{\omega}_k)\right)K\log(K(n+p))\right)$$

which we upper bound by

$$M \leq \mathcal{O}\left(\log(M)^3\left(S + R_{\mathcal{G}}\sum_{k=1}^{K}\mathrm{cut}_{\mathcal{G}}(\boldsymbol{\omega}_k)\right)K\log(Kn)\right).$$

Then the following technical lemma gives the result (proof in the Appendix).

Lemma 7 *Given a function $M:\mathbb{R}\to\mathbb{R}$, a constant $e>0$ such that $M(x)\leq ex\log(M(x))^3$, then there exist constants $a,b>0$ such that $M(x)\leq ax\log(x)^3$ for all $x>b$.*

We may also apply the technique of Herbster and Warmuth (1998) to the switching problem. Here, the underlying learning algorithm would be the perceptron with the kernel $\mathbf{G}^{-1} := \mathbf{L}_{\mathcal{G}}^{+} + R_{\mathcal{G}}\mathbf{1}\mathbf{1}^{T}$. As in (Cavallanti et al., 2010) the implicit assumption is that the underlying switching process is smooth and thus there is no parameter K, just a sequence of $S+1$ graph labelings. The other ingredient needed for a tracking kernel perceptron is an upper bound $\hat{\boldsymbol{\phi}} := \max_{k\in\{1,\ldots,S+1\}}\boldsymbol{\omega}_k^T\mathbf{G}\boldsymbol{\omega}_k$. The upper bound is then used to define an additional update to the perceptron, which maintains the hypothesis vector in a ball of squared norm equal to $\hat{\boldsymbol{\phi}}$. This perceptron update and projection step then lead to the bound (Herbster and Warmuth, 1998, Theorem 10),

$$M \leq \mathcal{O}\left(\left(\sum_{k=1}^{S}\sqrt{\hat{\boldsymbol{\phi}}(\boldsymbol{\omega}_k-\boldsymbol{\omega}_{k+1})^T\mathbf{G}(\boldsymbol{\omega}_k-\boldsymbol{\omega}_{k+1})} + \mathrm{cut}_{\mathcal{G}}(\boldsymbol{\omega}_{S+1})\right)R_{\mathcal{G}}\right).$$

Thus we observe with the projection kernel perceptron we pay a cost of

$$\sqrt{\hat{\boldsymbol{\phi}}(\boldsymbol{\omega}_k-\boldsymbol{\omega}_{k+1})^T\mathbf{G}(\boldsymbol{\omega}_k-\boldsymbol{\omega}_{k+1})}R_{\mathcal{G}}$$

for each switch $k\in\{1,\ldots,S\}$. Whereas when $K \ll S$ the dominant non poly-logarithmic term we pay per switch is $\mathcal{O}(K\log n)$.

6.3 Final Remarks

In this paper we presented a novel setting for online prediction over a graph. Our model is governed by K binary labelings and a latent K-labeling (defined on a second graph) which determines which one of the binary labelings is active at each trial.

We proposed an efficient algorithm for online prediction in this setting and derived a bound on the number of mistakes made by the algorithm. An interesting feature of this bound is that it mimics the bound one would obtain having *a-priori* information about which binary labeling is active at each trial. A shortcoming of the bound is that it requires knowledge of the number of binary labelings K and the threshold θ. In practice these parameters are not known in advance and techniques based on the "doubling trick" could be employed to tune the parameters.

Finally, we note that the problem considered in this paper could also be applied to the batch learning setting and our bound may be converted to a batch bound using techniques from (Cesa-Bianchi et al., 2004). In the batch setting a natural algorithm is given by empirical error minimization (Vapnik, 1998) over a hypothesis space of binary classifiers defined on the graph. This space is obtained by a certain function composition involving the binary labelings and the latent labeling. We conjecture that the problem of performing empirical error minimization over such a hypothesis space is NP-hard. Therefore in future work our algorithm could be employed to obtain an efficient sub-optimal solution to empirical error minimization in this challenging setting.

Acknowledgments

The third author acknowledges support from the EPSRC and GENES.

Appendix A. Appendix

In this appendix, we state some auxiliary results which are used in the main body of the paper.

The first result is the famous Golden-Thompson Inequality, whose proof can be found, for example, in (Bhatia, 1997).

Lemma A.8 *For any symmetric matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ we have that*

$$\mathrm{Tr}\left(\exp\left(\boldsymbol{A}+\boldsymbol{B}\right)\right) \leq \mathrm{Tr}\left(\exp\left(\boldsymbol{A}\right)\exp\left(\boldsymbol{B}\right)\right).$$

The next two results are taken from (Tsuda et al., 2005).

Lemma A.9 *If $\boldsymbol{A} \in \mathbf{S}_+^d$ with eigenvalues in $[0,1]$ and $a \in \mathbb{R}$, then*

$$(1-e^a)\boldsymbol{A} \preceq \boldsymbol{I} - \exp\left(a\boldsymbol{A}\right).$$

Lemma A.10 *If $\boldsymbol{A} \in \mathbf{S}_+^d$ and $\boldsymbol{B}, \boldsymbol{C}$ are symmetric matrices such that $\boldsymbol{B} \preceq \boldsymbol{C}$, then*

$$\mathrm{Tr}\left(\boldsymbol{A}\boldsymbol{B}\right) \leq \mathrm{Tr}\left(\boldsymbol{A}\boldsymbol{C}\right).$$

Next we show that the functions (13) and (14) are monotonic increasing. We will use the following lemma.

Lemma A.11 *For every $x > 0$ it holds that $\frac{2x}{2+x} < \log(1+x) < \frac{x}{\sqrt{x+1}}$.*

Proof To prove the right inequality, we let

$$f(x) = \frac{x}{\sqrt{x+1}} - \log(x+1).$$

Since $f(x) = 0$ as $x \to 0$, the result follows if we show that $f'(x) > 0$ for $x > 0$. We have that

$$f'(x) = \frac{x - 2\sqrt{x+1} + 2}{2(x+1)^{3/2}}\,.$$

With a change of variable $x \to z^2 - 1$, we have

$$\frac{x - 2\sqrt{x+1} + 2}{2(x+1)^{3/2}} = \frac{(1-z)^2}{2z^3}\,,$$

which is positive for $z \in (1, \infty]$ and hence $x \in (0, \infty)$.

The proof of the left inequality follows a similar pattern. ■

Lemma A.12 *The following function*

$$f(k) = k\left(\frac{1}{2}(k+3)\log\left(\frac{k+3}{k+1}\right) - 1\right)$$

is increasing for $k \geq 2$.

Proof Differentiating, we have

$$f'(k) = \frac{\left(2k^2 + 5k + 3\right)\log\left(\frac{k+3}{k+1}\right) - 4k - 2}{2(k+1)}\,.$$

We will check to see if the numerator of the above expression is positive. Using the left inequality in Lemma A.11 we have that

$$(2k^2 + 5k + 3)\log\left(\frac{k+3}{k+1}\right) - 4k - 2 \geq \frac{2(2k^2 + 5k + 3)}{2+k} - 4k - 2 = \frac{2}{2+k} > 0.$$

■

Lemma A.13 *The following function*

$$g(k) = k\left(\frac{k+1}{k+3} - \frac{1}{2}(k-1)\log\left(\frac{k+3}{k+1}\right)\right)$$

is increasing for $k \geq 2$.

Proof Differentiating, we have

$$g'(k) = \frac{2\left(2k^3+9k^2+6k+3\right)-(k+3)^2\left(2k^2+k-1\right)\log\left(\frac{k+3}{k+1}\right)}{2(k+1)(k+3)^2}\,. \tag{18}$$

We will show that the numerator of the above expression is positive. The right inequality in Lemma A.11 gives that

$$\log\left(\frac{k+3}{k+1}\right) < \frac{2\sqrt{\frac{k+3}{k+1}}}{k+3}.$$

Using this, we lower bound the numerator in the r.h.s. of equation (18) by

$$2\left(-(k+3)\sqrt{\frac{k+3}{k+1}}\left(2k^2+k-1\right)+k(k(2k+9)+6)+3\right).$$

With a change of variable $k \to \frac{3-y^2}{y^2-1}$, we have

$$\frac{8\left(y^6-7y^3+12y^2+3y^4(y-2)-3\right)}{\left(y^2-1\right)^3}\,.$$

Note $k \in [2,\infty)$ implies $y \in (1, \sqrt{\frac{5}{3}}]$. Since we are checking for positivity we strike the term $\frac{8}{(y^2-1)^3}$ which gives

$$y^6+3(y-2)y^4-7y^3+12y^2-3\,.$$

Factoring the above gives

$$(-1+y)^3(3+y(3+y)^2)\,,$$

which is positive for $y \in (1, \sqrt{\frac{5}{3}}]$. ■

Proof of Lemma 7. Without loss of generality let $e = 1$ (else consider the function M', defined by $M'(x) := M(x/e)$, instead of M (noting that $\log(ex)^3 \in O(\log(x)^3)$)).

Note first that we have some d such that for all $y > d$ we have that the function $y \to \frac{y}{\log(y)^3}$ is increasing.

Since $\exp(x) \in \omega(x^6)$ we have $\exp\left(x^{\frac{1}{3}}\right) \in \omega(x^2)$ so $\frac{1}{x}\exp\left(x^{\frac{1}{3}}\right) \in \omega(x)$. There hence exists a c such that for all $x > c$ we have $\frac{1}{x}\exp\left(x^{\frac{1}{3}}\right) > x$.

Let $b := \max\{c, \log(d)^3\}$. Now suppose we have some $x > b$. We then prove the inequality $\log(M(x))^3 \leq x$. To show this consider the converse, that $\log(M(x))^3 > x$. Then $M(x) > \exp\left(x^{\frac{1}{3}}\right)$. Since the function $y \to y/\log(y)^3$ is increasing for $y > d$ and $\exp\left(x^{\frac{1}{3}}\right) > d$, we then have that $M(x)/\log(M(x))^3 > \frac{1}{x}\exp\left(x^{\frac{1}{3}}\right)$, which is greater than x since $x > c$. But this contradicts the fact that $M(x) \leq x\log(M(x))^3$. So we have shown that $\log(M(x))^3 \leq x$.

If we have $M(x) > 8x\log(x)^3$ then we have $8x\log(x)^3 < M(x) \leq x\log(M(x))^3$ so we must have $2\log(x) < \log(M(x))$ so we have $x^2 < M(x)$. But, by above, $\log(M(x))^3 \leq x$, and hence $M(x) \leq x\log(M(x))^3 \leq x^2$ which is a contradiction. Hence we have that $M(x) \leq 8x\log(x)^3$ as required. ■

References

J. Abernethy, P. Bartlett, and A. Rakhlin. Multitask learning with expert advice. In *Proceedings 20th Annual Conference on Learning Theory*, pages 484–498, 2007.

D. Adamskiy, M.K. Warmuth, and W.M. Koolen. Putting Bayes to sleep. In *Advances in Neural Information Processing Systems 25*, pages 135–143, 2012.

S. Avishek, R. Piyush, H. Daumé III, and S. Venkatasubramanian. Online learning of multiple tasks and their relationships. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, pages 643–651, 2011.

R. Bhatia. *Matrix Analysis.* Springer, 1997.

A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In *Proceedings of the 18th International Conference on Machine Learning*, pages 19–26, 2001.

O. Bousquet and M.K. Warmuth. Tracking a small set of experts by mixing past posteriors. *The Journal of Machine Learning Research*, 3:363–396, 2003.

G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Linear algorithms for online multitask classification. *Journal of Machine Learning Research*, 1:2901–2934, 2010.

N. Cesa-Bianchi and C. Gentile. Tracking the best hyperplane with a simple budget perceptron. In *Proceedings of the 18th Conference on Learning Theory*, pages 483–498, 2006.

N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, 2004.

N. Cesa-Bianchi, C. Gentile, and F. Vitale. Fast and optimal prediction on a labeled tree. In *Proceedings of the 22nd Annual Conference on Learning*, 2009.

N. Cesa-Bianchi, C. Gentile, F. Vitale, and G. Zappella. Random spanning trees and the prediction of weighted graphs. In *Proceedings of the 27th International Conference on Machine Learning*, pages 175–182, 2010.

N. Cesa-Bianchi, P. Gaillard, G. Lugosi, and G. Stoltz. Mirror descent meets fixed share (and feels no regret). In *Advances in Neural Information Processing Systems 24*, pages 989–997, 2012.

O. Dekel, P.M. Long, and Y. Singer. Online learning of multiple tasks with a shared loss. *Journal of Machine Learning Research*, 8(10):2233–2264, 2007.

Y. Freund. Private communication, 2000. Also posted on http://www.learning-theory.org.

C. Gentile, M. Herbster, and S. Pasteris. Online similarity prediction of networked data from known and unknown graphs. In *Proceedings of the 26th Annual Conference on Learning Theory*, 2013.

L.A. Goldberg and M. Jerrum. The complexity of ferromagnetic Ising with local fields. *Combinatorics, Probability & Computing*, 16(1):43–61, 2007.

A. Gyorfi, T. Linder, and G. Lugosi. Tracking the best of many experts. In *Proceedings 18th Annual Conference on Learning Theory*, pages 204–216, 2005.

E. Hazan and C. Seshadhri. Efficient learning algorithms for changing environments. In *Proceedings of the 26th International Conference on Machine Learning*, pages 393–400, 2009.

M. Herbster and M. Pontil. Prediction on a graph with a perceptron. In *Advances in Neural Information Processing Systems 19*, pages 577–584, 2006.

M. Herbster and M.K. Warmuth. Tracking the best expert. *Machine Learning*, 32(2): 151–178, 1998.

M. Herbster and M.K. Warmuth. Tracking the best linear predictor. *The Journal of Machine Learning Research*, 1:281–309, 2001.

M. Herbster, M. Pontil, and L. Wainer. Online learning over graphs. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 305–312, 2005.

M. Herbster, G. Lever, and M. Pontil. Online prediction on large diameter graphs. In *Advances in Neural Information Processing Systems 21*, pages 649–656, 2008.

M. Herbster, M. Pontil, and S. Rojas-Galeano. Fast prediction on a tree. In *Advances in Neural Information Processing Systems*, pages 657–664, 2009.

J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52:2165–2176, 2004.

R.I. Kondor and J.D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *Proceedings of the 19th International Conference on Machine Learning*, pages 315–322, 2002.

W. M. Koolen and S. Rooij. Combining expert advice efficiently. In *21st Annual Conference on Learning Theory - COLT 2008, Helsinki, Finland, July 9-12, 2008*, pages 275–286, 2008.

K. Tsuda, G. Rätsch, and M.K. Warmuth. Matrix exponentiated gradient updates for on-line learning and Bregman projection. *Journal of Machine Learning Research*, 6:995–1018, 2005.

V. Vapnik. *Statistical Learning Theory.* Wiley-Blackwell, 1998.

F. Vitale, N. Cesa-Bianchi, C. Gentile, and G. Zappella. See the tree through the lines: The shazoo algorithm. In *Advances in Neural Information Processing Systems 23*, pages 1584–1592, 2011.

V. Vovk. Derandomizing stochastic prediction strategies. *Machine Learning*, 35(3):247–282, 1999.

M.K. Warmuth. Winnowing subspaces. In *Proceedings of the 24th International Conference on Machine Learning*, pages 999–1006, 2007.

X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In *Proceedings of the 23rd International Conference on Machine Learning*, pages 912–919, 2003.