Online Prediction on Large Diameter Graphs

Authored by:

Mark Herbster
Massimiliano Pontil
Guy Lever

Abstract

Current on-line learning algorithms for predicting the labelling of a graph have an important limitation in the case of large diameter graphs; the number of mistakes made by such algorithms may be proportional to the square root of the number of vertices, even when tackling simple problems. We overcome this problem with an efficient algorithm which achieves a logarithmic mistake bound. Furthermore, current algorithms are optimised for data which exhibits cluster-structure; we give an additional algorithm which performs well locally in the presence of cluster structure and on large diameter graphs.

1 Paper Body

We continue our study of online prediction of the labelling of a graph. We show a fundamental limitation of Laplacian-based algorithms: if the graph has a large diameter then the number of mistakes made by such algorithms may be proportional to the square root of the number of vertices, even when tackling simple problems. We overcome this drawback by means of an efficient algorithm which achieves a logarithmic mistake bound. It is based on the notion of a spine, a path graph which provides a linear embedding of the original graph. In practice, graphs may exhibit cluster structure; thus in the last part, we present a modified algorithm which achieves the ?best of both worlds?: it performs well locally in the presence of cluster structure, and globally on large diameter graphs.

1

Introduction

We study the problem of predicting the labelling of a graph in the online learning framework. Consider the following game for predicting the labelling of a graph: Nature presents a graph; nature queries a vertex vi1 ; the learner predicts y?1 ? {?1, 1}, the label of the vertex; nature presents a label y1 ; nature queries a vertex vi2 ; the learner predicts y?2 ; and so forth. The learner?s goal is to minimise the total number of mistakes M = —{t : y?t 6= yt }—. 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 labelling. In [9, 8, 7], the cut size (the number of edges between disagreeing labels) was used as a measure of the complexity of a graph?s labelling, and mistake bounds relative to this and the graph diameter were derived. The strength of the methods in [8, 7] is in the case when the graph exhibits ?cluster structure?. The apparent deficiency of these methods is that they have poor bounds when the graph diameter is large relative to the number of vertices. We observe that this weakness is not due to insufficiently tight bounds, but is a problem in their performance. In particular, we discuss an example of a n-vertex labelled graph with a single edge between disagreeing label sets. On this graph, sequential prediction using the common method based upon minimising the Laplacian semi-norm of a labelling, subject to ? constraints, incurs ?(n) mistakes (see Theorem 3). The expectation is that the number of mistakes incurred by an optimal online algorithm is bounded by O(ln n). We solve this problem by observing that there exists an approximate structure-preserving embedding of any graph into a path graph. In particular the cut-size of any labelling is increased by no more than a factor of two. We call this embedding a spine of the graph. The spine is the foundation on which we build two algorithms. Firstly we predict directly on the spine with the 1-nearest-neighbor algorithm. We demonstrate that this equivalent to the Bayes-optimal classifier for a particular Markov random field. A logarithmic mistake bound for learning on a path graph follows by the Halving algorithm analysis. Secondly, we use the spine of the graph as a foundation to add a binary support tree to the original graph. This enables us to prove a bound which is the ?best of both worlds? ? if the predicted set of vertices has cluster-structure we will obtain a bound appropriate for that case, but if instead, the predicted set exhibits a large diameter we will obtain a polylogarithmic bound.

Previous work. The seminal approach to semi-supervised learning over graphs in [3] is to predict with a labelling which is consistent with a minimum label-separating cut. More recently, the graph Laplacian has emerged as a key object in semi-supervised learning, for example the semi-norm induced by the Laplacian is commonly either directly minimised subject to constraints, or used as a regulariser [14, 2]. In [8, 7] the online graph labelling problem was studied. An aim of those papers was to provide a natural interpretation of the bound on the cumulative mistakes of the kernel perceptron when the kernel is the pseudoinverse of the graph Laplacian ? bounds in this case being relative to the cut and (resistance) diameter of the graph. In this paper we necessarily build directly on the very recent results in [7] as those results depend on the resistance diameter of the predicted vertex set as opposed to the whole graph [8]. The online graph labelling problem is also studied in [13], and here the graph structure is not given initially. A slightly weaker logarithmic bound for the online graph labelling problem has also been independently derived via a connection to an online routing problem in the very recent [5].

2

Preliminaries

We study the process of predicting a labelling defined on the vertices of a graph. Following the classical online learning framework, a sequence of labelled vertices {(vi1 , y1), (vi2 , y2), . . . }, the trial sequence, is presented to a learning algorithm such that, on sight of each vertex vit , the learner makes a prediction y?t for the label value, after which the correct label is revealed. This feedback information is then used by the learning algorithm to improve its performance on further examples. We analyse the performance of a learning algorithm in the mistake bound framework [12] ? the aim is to minimise the maximum possible cumulative number of mistakes made on the training sequence. A graph G = (V, E) is a collection of vertices V = {v1 , . . . , vn } joined by connecting (possibly weighted) edges. Denote i ? j whenever vi and vj are connected so that E = {(i, j) : i ? j} is the set of unordered pairs of connected vertex indices. Associated with each edge (i, j) ? E is a weight Aij , so that A is the n ? n symmetric adjacency matrix. We say that G is unweighted if Aij = 1 for every (i, j) ? E and is 0 otherwise. In this paper, we consider only connected graphs ? that is, graphs such that there exists a path between any two vertices. The Laplacian G of a graph P G is the n ? n matrix G = D ? A, where D is the diagonal degree matrix such that Dii = j Aij . The quadratic form associated with the Laplacian relates to the cut size of graph labellings. Definition 1. Given a labelling u ? IRn of G = (V, E) we define the cut size of u by 1 T 1 X ?G (u) = u Gu = Aij (ui ? uj)2 . (1) 4 4 (i,j)?E

n

In particular, if u ? {?1, 1} we say that a cut occurs on edge (i, j) if ui 6= uj and ?G (u) measures the number of cuts. We evaluate the performance of prediction algorithms in terms of the cut size and the resistance diameter of the graph. There is an established natural connection between graphs and resistive networks where each edge (i, j) ? E is viewed as a resistor with resistance 1/Aij [4]. Thus the effective resistance rG (vi , vj) between vertex vi and vj is the potential difference needed to induce a unit current flow between vi and vj . The effective resistance may be computed by the formula [11] rG (vi , vj) = (ei ? ej)T G+ (ei ? ej), (2) + n where ? ? denotes the pseudoinverse and e1 , . . . , en are the canonical basis vectors of IR . The resistance diameter of a graph RG := maxvi ,vj ?V rG (vi , vj) is the maximum effective resistance between any pair of vertices on the graph.

3

Limitations of online minimum semi-norm interpolation

As we will show, it is possible to develop online algorithms for predicting the labelling of a graph which have a mistake bound that is a logarithmic function of the number of vertices. Conversely, we first highlight a deficiency in a standard Laplacian based method for predicting a graph labelling. Given a partially labelled graph G = (V, E) with —V — = n ? that is, such that for some ‘ ? n, y‘ ? {?1, 1}‘ is a labelling defined on the ‘ vertices V‘ = {vi1 , vi2 , . . . , vi‘ } ? the minimum semi-norm interpolant is defined by y? = argmin{uT Gu : u ? IRn , uik = yk , k = 1, . . . , ‘}.

We then predict using y?i = sgn(? yi), for i = 1, . . . , n. The common

justification behind the above learning paradigm [14, 2] is that minimizing the cut (1) encourages neighbouring vertices to be similarly labelled. However, we now demonstrate that in the online setting such a regime will perform poorly on ? certain graph constructions ? there exists a trial sequence on which the method will make at least ?(n) mistakes. Definition 2. An octopus graph of size d is defined to be d path graphs (the tentacles) of length d (that is, with d + 1 vertices) all adjoined at a common end vertex, to which a further single head vertex is attached, so that n = —V — = d2 + 2. This corresponds to the graph O1,d,d discussed in [8]. Theorem 3. Let G = (V, E) be an octopus graph of size d and y = (y1 , . . . , y—V —) the labelling such that yi = 1 if vi is the head vertex and yi = ?1 otherwise. There exists a trial sequence for p which online minimum semi-norm interpolation makes ?(—V —) mistakes. Proof. Let the first query vertex be the head vertex, and let the end vertex of a tentacle be queried at each subsequent trial. We show that this strategy forces at least d mistakes. The solution to the minimum semi-norm interpolation with boundary Pn values problem is precisely the harmonic solution [4] y? (that is, for every unlabeled vertex vj , i=1 Aij (? yi ? y?j) = 0). If the graph is connected y? is unique and the graph labelling problem is identical to that of identifying the potential at each vertex of a resistive network defined on the graph where each edge corresponds to a resistor of 1 unit; the harmonic principle corresponds to Kirchoff?s current law in this case. Using this analogy, suppose that the end points of k ¡ d tentacles are labelled and that the end vertex vq of an unlabelled tentacle is queried. Suppose a current of k? flows from the head to the body of the graph. By Kirchoff?s law, a current of ? flows along each labelled tentacle (in order to obey the harmonic principle at 2 every vertex it is clear that no current flows along the unlabelled tentacles). By Ohm?s law ? = d+k . Minimum semi-norm interpolation therefore results in the solution 2k y?q = 1 ? ? 0 iff k ? d. d+k Hence the minimum semi-norm solution predicts incorrectly whenever k ¡ d and the algorithm makes at least d mistakes. The above demonstrates a limitation in the method of online Laplacian minimum semi-norm interpolation for predicting a graph labelling ? the mistake bound can be proportional to the square root of the number of data points. We solve these problems in the following section.

4

A linear graph embedding

We demonstrate a method of embedding data represented as a connected graph G into a path graph, we call it a spine of G, which partially preserves the structure of G. Let Pn be the set of path graphs with n vertices. We would like to find a path graph with the same vertex set as G, which solves ?P (u) . min max P?Pn u?{?1,1}n ?G (u) If a Hamiltonian path H of G (a path on G which visits each vertex precisely once) exists, then (u) the approximation ratio is ??H ? 1. The problem of finding a Hamiltonian path is NP-complete G (u) however, and such a path is not guaranteed to exist. As we shall see, a spine S of G may be found S (u) efficiently and satisfies ? ?G (u) ? 2. We now detail the construction of a spine of a graph G = (V, E), with —V — = n. Starting from any node, G is traversed in the manner of a depth-first search (that is, each

vertex is fully explored before backtracking to the last unexplored vertex), and an ordered list VL = {vl1 , vl2 , . . . , vl2m+1 } of the vertices (m ? —E—) in the order that they are visited is formed, allowing repetitions when a vertex is visited more than once. Note that each edge in EG is traversed no more than twice when forming VL . Define an edge multiset EL = {(l1 , l2), (l2 , l3), . . . , (l2m , l2m+1)} ? the set of pairs of consecutive vertices in VL . Let u be an P arbitrary labelling of G and denote, as usual, P ?G (u) = 41 (i,j)?EG (ui ? uj)2 and ?L (u) = 14 (i,j)?EL (ui ? uj)2 . Since the multiset EL contains every element of EG no more than twice, ?L (u) ? 2?G (u). We then take any subsequence VL0 of VL containing every vertex in V exactly once. A spine S = (V, ES) is a graph formed by connecting each vertex in V to its immediate neighbours in

the subsequence VL0 with an edge. Since a cut occurs between connected vertices vi and vj in S only if a cut occurs on some edge in EL located between the corresponding vertices in the list VL we have ?S (u) ? ?L (u) ? 2?G (u).

(3)

Thus we have reduced the problem of learning the cut on a generic graph to that of learning the cut on a path graph. In the following we see that 1-nearest neighbour (1-NN) algorithm is a Bayes optimal algorithm for this problem. Note that the 1-NN algorithm does not perform well ? on general graphs; on the octopus graph discussed above, for example, it can make at least ?(n) mistakes, and even ?(n) mistakes on a related graph construction [8].

5

Predicting with a spine

We consider implementing the 1-NN algorithm on a path graph and demonstrate that it achieves a mistake bound which is logarithmic in the length of the line. Let G = (V, E) be a path graph, where V = {v1 , v2 , . . . , vn } is the set of vertices and E = {(1, 2), (2, 3), . . . , (n ? 1, n)}. The nearest neighbour algorithm, in the standard online learning framework described above, attempts to predict a graph labelling by producing, for each query vertex vit , the prediction y?t which is consistent with the label of the closest labelled vertex (and predicts randomly in the case of a tie). Theorem 4. Given the task of predicting the labelling of any unweighted, n-vertex path graph P in the online framework, the number of mistakes, M , incurred by the 1-NN algorithm satisfies

n?1 ?P (u) M ? ?P (u) log2 + + 1, (4) ?P (u) ln 2 where u ? {?1, 1}n is any labelling consistent with the trial sequence. Proof. We shall prove the result by noting that the Halving algorithm [1] (under certain conditions on the probabilities assigned to each hypothesis) implements the nearest neighbour algorithm on a path graph. Given any input space X and finite binary concept class C ? {?1, 1}—X— , the Halving algorithm learns any target concept c? ? C as follows. Each hypothesis c ? C is given an associated probability p(c). A sequence of labelled examples {(x1 , y1), . . . , (xt?1 , yt?1)} ? X ? {?1, 1}, is revealed in accordance with the usual online framework. Let Ft be the set of feasible hypotheses at trial t; Ft = {c : c(xs) = ys ?s ¡ t}. Given an unlabelled example xtP? X at trial t the predicted label y?t is that which agrees with the majority vote ? that is, such that it predicts randomly if this is equal to most

MH mistakes with
1 2).
c?Ft ,c(xt)=y ?t
P
c?Ft
p(c)
p(c)
¿
1 2
(and
It is well known [1] that the Halving algorithm makes at
MH ? log2
1 p(c?)
.
(5)

We now define a probability distribution over the space of all labellings u ? {?1, 1}n of P such that the Halving algorithm with these probabilities implements the nearest neighbour algorithm. Let a cut occur on any given edge with probability ?, independently of all other cuts; Prob(ui+1 6= ui) = ? ?i ¡ n. The position of all cuts fixes the labelling up to flipping every label, and each of these two resulting possible arrangements are equally likely. This recipe associates with each possible labelling u ? {?1, 1}n a probability p(u) which is a function of the labelling?s cut size 1 ?P (u) ? (1 ? ?)n?1??P (u) . (6) 2 This induces a full joint probability distribution on the space of vertex labels. In fact (6) is a Gibbs measure and as such defines a Markov random field over the space of vertex labels [10]. The mass function p therefore satisfies the Markov property p(u) =

p(ui = ? — uj = ?j ?j 6= i) = p(ui = ? — uj = ?j ?j ? Ni),
(7)

where here Ni is the set of vertices neighbouring vi ? those connected to vi by an edge. We will give an equivalent Markov property which allows a more general conditioning to reduce to that over boundary vertices.

Definition 5. Given a path graph P = (V, E), a set of vertices V 0 ? V and a vertex vi ? V , we define the boundary vertices v‘ , vr (either of which may be vacuous) to be the two vertices in V 0 that are closest to vi in each direction along the path; its nearest neighbours in each direction. The distribution induced by (6) satisfies the following Markov property; given a partial labelling of P defined on a subset V 0 ? V , the label of any vertex vi is independent of all labels on V 0 except those on the vertices v‘ , vr (either of which could be vacuous) p(ui = ? — uj = ?j , ?j : vj ? V 0)

= p(ui = ? — u‘ = ¿ , ur = ?r).
(8)

Given the construction of the probability distribution formed by independent cuts on graph edges, we can evaluate conditional probabilities. For example, p(uj = ? — uk = ?) is the probability of an even number of cuts between vertex vj and vertex vk . Since cuts occur with probability ? and there

are —k?j— possible arrangements of s cuts we have s
p(uj = ? — uk = ?) =
X —k ? j— 1 ?s (1 ? ?)—k?j—?s = (1 + (1 ? 2?)—k?j—). s 2 s even
(9)
X —k ? j— 1 ?s (1 ? ?)—k?j—?s = (1 ? (1 ? 2?)—k?j—). s 2
(10)
Likewise we have that p(uj 6= ? — uk = ?) =
s odd

Note also that for any single vertex we have p(ui = ?) = 12 for ? ? {?1, 1}.
Lemma 6. Given the task of predicting the labelling of an n-vertex path graph online, the Halving algorithm, with a probability distribution over the labellings defined as in (6) and such that 0 ¡ ? ¡ 12 , implements the nearest neighbour algorithm. Proof. Suppose that t ? 1 trials have been performed so that we have a partial labelling of a subset V 0 ? V , {(vi1 , y1), (vi2 , y2), . . . , (vit?1 , yt?1)}. Suppose the label of vertex vit is queried so that the Halving algorithm makes the following prediction y?t for vertex vit : y?t = y if p(uit = y — uij = yj ? 1 ? j ¡ t) ¿ 21 , y?t = ?y if p(uit = y — uij = yj ? 1 ? j ¡ t) ¡ 21 (and predicts randomly if this probability is equal to 12). We first consider the case where the conditional labelling includes vertices on both sides of vit . We have, by (8), that p(uit = y — uij = yj ? 1 ? j ¡ t)
= p(uit = y — u‘ = y? (‘) , ur = y? (r)) =
p(u‘ = y? (‘) — ur = y? (r) , uit = y)p(ur = y? (r) , uit = y) p(u‘ = y? (‘) , ur = y? (r))
=
p(u‘ = y? (‘) — uit = y)p(ur = y? (r) — uit = y) p(u‘ = y? (‘) — ur = y? (r))
(11)

where v‘ and vr are the boundary vertices and ? (‘) and ? (r) are trials at which vertices v‘ and vr are queried, respectively. We can evaluate the right hand side of this expression using (9, 10). To show equivalence with the nearest neighbour method whenever ? ¡ 12 , we have from (9, 10, 11) p(uit = y — u‘ = y, ur 6= y)
=
(1 + (1 ? 2?)—‘?it —)(1 ? (1 ? 2?)—r?it —) 2(1 ? (1 ? 2?)—‘?r—)

which is greater than 12 if —‘ ? it — ¡ —r ? it — and less than 21 if —‘ ? it — ¿ —r ? it —. Hence, this produces predictions exactly in accordance with the nearest neighbour scheme. We also have more simply that for all it , ‘ and r and ? ¡ 12 p(uit = y — u‘ = y, ur = y) ¿
1 1 , and p(uit = y — u‘ = y) ¿ . 2 2

This proves the lemma for all cases. A direct application of the Halving algorithm mistake bound (5) now gives
1 2 M ? log2 = log2 p(u) ??P (u) (1 ? ?)n?1??P (u)
P (u) 1 where u is any labelling consistent with the trial sequence. We choose ? = min(?n?1 , 2) (note P (u) that the bound is vacuous when ?n?1 ¿ 12 since M is necessarily upper bounded by n) giving

?P (u) n?1 + (n ? 1 ? ?P (u)) log2 1 + +1 M ? ?P (u) log2 ?P (u) n ? 1 ? ?P (u)

n?1 ?P (u) ? ?P (u) log2 + + 1. ?P (u) ln 2

This proves the theorem. The nearest neighbour algorithm can predict the labelling of any graph G = (V, E), by first transferring the data representation to that of a spine S of G, as presented in Section 4. We now apply the above argument to this method and immediately deduce our first main result. Theorem 7. Given the task of predicting the labelling of any unweighted, connected, n-vertex graph G = (V, E) in the online framework, the number of mistakes, M , incurred by the nearest neighbour algorithm operating on a spine S of G satisfies

n?1 2?G (u) M ? 2?G (u) max 0, log2 + 1, (12) + 2?G (u) ln 2 where u ? {?1, 1}n is any labelling consistent with the trial sequence. Proof. Theorem 4 gives bound (4) for predicting on any path, hence M ? ?S (u) log2 ?S (u) ln 2 + 1. Since this is an increasing function of ?S (u) for ?S (u) ? n ?S (u) ? n ? 1 (M is necessarily upper bounded by n) we upper bound

n?1 ?S (u)

+

? 1 and is vacuous at substituting ?S (u) ?

2?G (u) (equation (3)). We observe that predicting with the spine is a minimax improvement over Laplacian minimal seminorm interpolation. Recall Theorem 3, there we showed ? that there exists a trial sequence such that Laplacian p minimal semi-norm interpolation incurs ?(n) mistakes. In fact this trivially generalizes to ?(?G (u)n) mistakes by creating a colony of ?G (u) octopi then identifying each previously separate head vertex as a single central vertex. The upper bound (12) is smaller than the prior lower bound. The computational complexity for this algorithm is O(—E— + —V — ln —V —) time. We compute the spine in O(—E—) time by simply listing vertices in the order in which they are first visited during a depthfirst search traversal of G. Using online 1-NN requires O(—V — ln —V —) time to predict an arbitrary vertex sequence using a self-balancing binary search tree (e.g., a red-black tree) as the insertion of each vertex into the tree and determination of the nearest left and right neighbour is O(ln —V —).

6

Prediction with a binary support tree

The Pounce online label prediction algorithm [7] is designed to exploit cluster structure of a graph G = (V, E) and achieves the following mistake bound M ? N (X, ?, rG) + 4?G (u)? + 1,

(13)

for any ? ¿ 0. Here, u ? IRn is any labelling consistent with the trial sequence, X = {vi1 , vi2 , . . . } ? V is the set of inputs and N (X, ?, rG) is a covering number ? the minimum number of balls of resistance diameter ? (see Section 2) required to cover X. The mistake bound (13) can be preferable to (12) whenever the inputs are sufficiently clustered and so has a cover of small diameter sets. For example, consider two (m + 1)-cliques, one labeled ?+1?, one ??1? with cm arbitrary interconnecting edges (c ? 1) here the bound (12)

is vacuous while (13) is M ? 8c + 3 2 (with ? = m , N (X, ?, rG) = 2, and ?G (u) = cm). An input space V may have both local cluster structure yet have a large diameter. Imagine a ?universe? such that points are distributed into many dense clusters such that some sets of clusters are tightly packed but overall the distribution is quite diffuse. A given ?problem? X ? V may then be centered on a few clusters or alternatively encompass the entire space. Thus, for practical purposes, we would like a prediction algorithm

which achieves the ?best of both worlds?, that is a mistake bound which is no greater, in order of magnitude, than the maximum of (12) and (13). The rest of this paper is directed toward this goal. We now introduce the notion of binary support tree, formalise the Pounce method in the support tree setting and then prove the desired result. Definition 8. Given a graph G = (V, E), with —V — = n, and spine S, we define a binary support tree of G to be any binary tree T = (VT , ET) of least possible depth, D, whose leaves are the vertices of S, in order. Note that D ¡ log2 (n) + 1. We show that there is a weighting of the support tree which ensures that the resistance diameter of the support tree is small, but also such that any labelling of the leaf vertices can be extended to the support tree such that its cut size remains small. This enables effective learning via the support tree. A related construction has been used to build preconditioners for solving linear systems [6]. Lemma 9. Given any spine graph S = (V, E) with —V — = n, and labelling u ? {?1, 1}n , with ? ? [?1, 1]—VT — support tree T = (VT , ET), there exists a weighting A of T , and a labelling u ? and u are identical on V , ?T (u) ? ¡ ?S (u) and RT ? (log2 n + 1)(log2 n + of T such that u 4)(log2 (log2 n + 2))2 . Proof. Let vr be the root vertex of T . Suppose each edge (i, j) ? ET has a weight Aij , which is a function of the edge?s depth d = max{dT (vi , vr), dT (vj , vr)}, Aij = W (d) where dT (v, v 0) ? such is the number of edges in the shortest path from v to v 0 . Consider the unique labelling u that, for 1 ? i ? n we have u ?i = ui and such that for every other vertex vp ? VT , with child u ? +? u vertices vc1 , vc2 , we have u ?p = c1 2 c2 , or u ?p = u ?c in the case where vp has only one child, vc . Suppose the edges (p, c1), (p, c2) ? ET are at some depth d in T , and let V 0 ? V correspond to the leaf vertices of T descended from vp . Define ?S (uV 0) to be the cut of u restricted to vertices in V 0 . If u ? c1 = u ?c2 then (? up ? u ?c1)2 + (? up ? u ?c2)2 = 0 ? 2?S (uV 0), and if u ?c1 6= u ?c2 then 2 2 (? up ? u ?c1) + (? up ? u ?c2) ? 2 ? 2?S (uV 0). Hence

W (d) (? up ? u ?c1)2 + (? up ? u ?c2)2 ? 2W (d)?S (uV 0) (14) (a similar inequality is trivial in the case that vp has only one child). Since the sets of leaf descendants of all vertices at depth d form a partition of V , summing (14) first over all parent nodes at a given depth and then over all integers d ? [1, D] gives ? ?2 4?T (u)

D X

W (d)?S (u).

d=1

(15) We then choose 1 (d + 1)(log2 (d + 1))2 R? ? 21 + ln2 2 2 x ln12 x dx =

W (d) = and note that

P?
1 d=1 (d+1)(log2 (d+1))2
(16) 1 2
+ ln 2 ¡ 2.

PD Further, RT = 2 d=1 (d + 1)(log2 (d + 1))2 ? D(D + 3)(log2 (D + 1))2 and so D ? log2 n + 1 gives the resistance bound. Definition 10. Given the task of predicting the labelling of an unweighted graph G = (V, E) the ? is formed augmented Pounce algorithm proceeds as follows: An augmented graph G? = (V? , E) by attaching a binary support tree of G, with weights defined as in (16), to G; formally let T = (VT , ET) be such a binary support tree of G, then G? = (VT , E ? ET). The Pounce algorithm is ? then used to predict the (partial) labelling defined on G. Theorem 11. Given the task of predicting the labelling of any unweighted, connected, n-vertex graph G = (V, E) in the online framework, the number of mistakes, M , incurred by the augmented Pounce algorithm satisfies M ? min{N (X, ?, rG) + 12?G (u)?} + 1, (17) ?¿0

where N (X, ?, rG) is the covering number of the input set X = {vi1 , vi2 , . . . } ? V relative to the resistance distance rG of G and u ? IRn is any labelling consistent with the trial sequence. Furthermore, M ? 12?G (u)(log2 n + 1)(log2 n + 4)(log2 (log2 n + 2))2 + 2. (18)

Proof. Let u be some labelling consistent with the trial sequence. By (3) we have that ?S (u) ? 2?G (u) for any spine S of G. Moreover, by the arguments in Lemma 9 there exists some labelling ? of the weighted support tree T of G, consistent with u on V , such that ?T (u) ? ¡ ?S (u). We then u have ? = ?T (u) ? + ?G (u) ¡ 3?G (u). ?G?(u)
(19)

By Rayleigh?s monotonicity law the addition of the support tree does not increase the resistance between any vertices on G, hence N (X, ?, rG?) ? N (X, ?, rG).
(20)

? yields ? on G, Combining inequalities (19) and (20) with the pounce bound (13) for predicting u ? + 1 ? N (X, ?, rG) + 12?G (u)? + 1. M ? N (X, ?, rG?) + 4?G?(u)? ? G? + 2 ? which proves (17). We prove (18) by covering G? with single ball so that M ? 4?G?(u)R 12?G (u)RT + 2 and the result follows from the bound on RT in Lemma 9.

7
Conclusion

We have explored a deficiency with existing online techniques for predicting the labelling of a graph. As a solution, we have presented an approximate cut-preserving embedding of any graph G = (V, E) into a simple path graph, which we call a spine, such that an implementation of the 1nearest-neighbours algorithm is an efficient realisation of a Bayes optimal classifier. This therefore achieves a mistake bound which is logarithmic in the size of the vertex set for any graph, and the complexity of our algorithm is of O(—E— + —V — ln —V —). We further applied the insights gained to a second algorithm ? an augmentation of the Pounce algorithm, which achieves a polylogarithmic

performance guarantee, but can further take advantage of clustered data, in which case its bound is relative to any cover of the graph.

2 References

[1] J. M. Barzdin and R. V. Frievald. On the prediction of general recursive functions. Soviet Math. Doklady, 13:1224?1228, 1972. [2] M. Belkin and P. Niyogi. Semi-supervised learning on riemannian manifolds. Machine Learning, 56:209? 239, 2004. [3] A. Blum and S. Chawla. Learning from labeled and unlabeled data using graph mincuts. In Proc. 18th International Conf. on Machine Learning, pages 19?26. Morgan Kaufmann, San Francisco, CA, 2001. [4] P. Doyle and J. Snell. Random walks and electric networks. Mathematical Association of America, 1984. [5] J. Fakcharoenphol and B. Kijsirikul. Low congestion online routing and an improved mistake bound for online prediction of graph labeling. CoRR, abs/0809.2075, 2008. [6] K. Gremban, G. Miller, and M. Zagha. Performance evaluation of a new parallel preconditioner. Parallel Processing Symposium, International, 0:65, 1995. [7] M. Herbster. Exploiting cluster-structure to predict the labeling of a graph. In The 19th International Conference on Algorithmic Learning Theory, pages 54?69, 2008. [8] M. Herbster and M. Pontil. Prediction on a graph with a perceptron. In B. Sch?olkopf, J. Platt, and T. Hoffman, editors, Advances in Neural Information Processing Systems 19, pages 577?584. MIT Press, Cambridge, MA, 2007. [9] M. Herbster, M. Pontil, and L. Wainer. Online learning over graphs. In ICML ?05: Proceedings of the 22nd international conference on Machine learning, pages 305?312, New York, NY, USA, 2005. ACM. [10] R. Kinderman and J. L. Snell. Markov Random Fields and Their Applications. Amer. Math. Soc., Providence, RI, 1980. [11] D. Klein and M. Randi?c. Resistance distance. Journal of Mathematical Chemistry, 12(1):81?95, 1993. [12] N. Littlestone. Learning when irrelevant attributes abound: A new linear-threshold algorithm. Machine Learning, 2:285?318, 1988. [13] K. Pelckmans and J. A. Suykens. An online algorithm for learning a labeling of a graph. In In Proceedings of the 6th International Workshop on Mining and Learning with Graphs, 2008. [14] X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using gaussian fields and harmonic functions. In 20-th International Conference on Machine Learning (ICML-2003), pages 912?919, 2003.