Computer Science and Artificial Intelligence Laboratory

Technical Report

MIT-CSAIL-TR-2008-051 August 7, 2008

Transductive Ranking on Graphs

Shivani Agarwal

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

Transductive Ranking on Graphs*

Shivani Agarwal
Massachusetts Institute of Technology
shivani@mit.edu

August 3, 2008

Abstract

In ranking, one is given examples of order relationships among objects, and the goal is to learn from these examples a real-valued ranking function that induces a ranking or ordering over the object space. We consider the problem of learning such a ranking function in a transductive, graph-based setting, where the object space is finite and is represented as a graph in which vertices correspond to objects and edges encode similarities between objects. Building on recent developments in regularization theory for graphs and corresponding Laplacian-based learning methods, we develop an algorithmic framework for learning ranking functions on graphs. We derive generalization bounds for our algorithms in transductive models similar to those used to study other transductive learning problems, and give experimental evidence of the potential benefits of our framework.

1 Introduction

The problem of ranking, in which the goal is to learn a real-valued ranking function that induces a ranking or ordering over an instance space, has gained much attention in machine learning in recent years (Cohen et al, 1999; Herbrich et al, 2000; Crammer and Singer, 2002; Joachims, 2002; Freund et al, 2003; Agarwal et al, 2005; Clemencon et al, 2005; Rudin et al, 2005; Burges et al, 2005; Cossock and Zhang, 2006; Cortes et al, 2007). In developing algorithms for ranking, the main setting that has been considered so far is an inductive setting with vector-valued data, where the algorithm receives as input a finite number of objects in some Euclidean space $\mathbb{R}^n$, together with examples of order relationships or preferences among them, and the goal is to learn from these examples a ranking function $f : \mathbb{R}^n \to \mathbb{R}$ that orders future objects accurately. (A real-valued function $f : X \to \mathbb{R}$ is considered to order/rank $x \in X$ higher than $x' \in X$ if $f(x) > f(x')$, and vice-versa.)

In this paper, we consider the problem of learning a ranking function in a transductive, graph-based setting, where the instance space is finite and is represented in the form of a graph. Formally, we wish to develop ranking algorithms which can take as input a weighted graph $G = (V, E, w)$ (where each vertex in V corresponds to an object, an edge in E connects two similar objects, and a weight $w(i, j)$ denotes the similarity between objects i and j), together with examples of order relationships among a small number of elements in V, and can learn from these examples a good ranking function $f : V \to \mathbb{R}$ over V.

*A preliminary version of this paper appeared in the Proceedings of the 23rd International Conference on Machine Learning (ICML) in 2006.

Graph representations of data are important for many applications of machine learning. For example, such representations have been shown to be useful for data that lies in a high-dimensional space but actually comes from an underlying low-dimensional manifold (Roweis and Saul, 2000; Tenenbaum et al, 2000; Belkin and Niyogi, 2004). More importantly, graphs form the most natural data representation for an increasing number of application domains in which pair-wise similarities among objects matter and/or are easily characterized; for example, similarities between biological sequences play an important role in computational biology. Furthermore, as has been observed in other studies on transductive graph-based learning (see, for example, (Johnson and Zhang, 2008)), and as our experimental results show in the context of ranking, when the instance space is finite and known in advance, exploiting this knowledge in the form of an appropriate similarity graph over the instances can improve prediction over standard inductive learning.

There have been several developments in theory and algorithms for learning over graphs, in the context of classification and regression. In our work we build on some of these recent developments – in particular, developments in regularization theory for graphs and corresponding Laplacian-based learning methods (Smola and Kondor, 2003; Belkin and Niyogi, 2004; Belkin et al, 2004; Zhou and Schölkopf, 2004; Zhou et al, 2004; Herbster et al, 2005) – to develop an algorithmic framework for learning ranking functions on graphs.[1]

After some preliminaries in Section 2, we describe our basic algorithmic framework in Section 3. Our basic algorithm is derived for undirected graphs and can be viewed as performing regularization within a reproducing kernel Hilbert space (RKHS) whose associated kernel is derived from the graph Laplacian. In Section 4 we discuss various extensions of the basic algorithm, including the use of other kernels and the case of directed graphs. In Section 5 we derive generalization bounds for our algorithms; our bounds are derived in transductive models similar to those used to study other transductive learning problems, and make use of some recent results on the stability of kernel-based ranking algorithms (Agarwal and Niyogi, 2008). We give experimental evidence of the potential benefits of our framework in Section 6, and conclude with a discussion in Section 7.

2 Preliminaries

Consider a setting in which there is a finite instance space that is represented as a weighted, undirected graph $G = (V, E, w)$, where $V = \{1, \ldots, n\}$ is a set of vertices corresponding to objects (instances), $E \subseteq V \times V$ is a set of edges connecting similar objects with $(i, j) \in E \Rightarrow (j, i) \in E$, and $w : E \rightarrow \mathbb{R}^+$ is a symmetric weight function such that for any $(i, j) \in E$, $w(i, j) = w(j, i)$ denotes the similarity between objects i and j. The learner is given the graph G together with a small number of examples of order relationships among vertices in V, and the goal is to learn a ranking function $f : V \rightarrow \mathbb{R}$ that ranks accurately all the vertices in V.

There are many different ways to describe order relationships among objects, corresponding to different settings of the ranking problem. For example, in the bipartite ranking problem (Freund et al, 2003; Agarwal et al, 2005), the learner is given examples of objects labeled as positive or negative, and the goal is to learn a ranking in which positive objects are ranked higher than negative ones. As in (Cortes et al, 2007; Agarwal and Niyogi, 2008), we consider a setting in which the learner is given examples of objects labeled by real numbers, and the goal is to learn a ranking in which objects labeled by larger numbers are ranked higher than objects labeled by smaller numbers. Such problems arise, for example, in information retrieval, where one is interested in retrieving documents from some database that are 'relevant' to some topic; in this case,

[1] Note that Zhou et al (2004) also consider a ranking problem on graphs; however, the form of the ranking problem they consider is very different from that considered in this paper. In particular, in the ranking problem considered in (Zhou et al, 2004), the input does not involve order relationships among objects.

one is given examples of documents with real-valued relevance scores with respect to the topic of interest, and the goal is to produce a ranking of the documents such that more relevant documents are ranked higher than less relevant ones.

More formally, in the setting we consider, each vertex $i \in V$ is associated with a real-valued label y_i in some bounded set $Y \subset \mathbb{R}$, which we take without loss of generality to be $Y = [0, M]$ for some $M > 0$; for simplicity, we assume that y_i is fixed for each i (not random). The learner is given as training examples the labels $y_{i_1}, \ldots, y_{i_m}$ for a small set of vertices $S = \{i_1, \ldots, i_m\} \subset V$, and the goal is to learn from these examples a ranking function $f : V \to \mathbb{R}$ that ranks vertices with larger labels higher than those with smaller labels; the penalty for mis-ranking a pair of vertices is proportional to the absolute difference between their real-valued labels. The quality of a ranking function $f : V \to \mathbb{R}$ (or equivalently, $\mathbf{f} \in \mathbb{R}^n$, with ith element $f_i = f(i)$; we shall use these two representations interchangeably) can then be measured by its ranking error with respect to V, which we denote by $R_V(f)$ and define as

$$R_V(f) \quad = \quad \frac{1}{\binom{n}{2}} \sum_{i<j} |y_i - y_j| \left(\mathbf{I}_{\{(y_i - y_j)(f_i - f_j) < 0\}} + \frac{1}{2} \mathbf{I}_{\{f_i = f_j\}} \right), \tag{1}$$

where $\mathbf{I}_{\{\phi\}}$ is 1 if ϕ is true and 0 otherwise. The ranking error $R_V(f)$ is the expected mis-ranking penalty of f on a pair of vertices drawn uniformly at random (without replacement) from V, assuming that ties are broken uniformly at random.[2]

The transductive, graph-based ranking problem we consider can thus be summarized as follows: given a graph $G = (V, E, w)$ and real-valued labels $y_{i_1}, \ldots, y_{i_m} \in [0, M]$ for a small set of vertices $S = \{i_1, \ldots, i_m\} \subset V$, the goal is to learn a ranking function $f : V \to \mathbb{R}$ that minimizes $R_V(f)$. Since the labels for vertices in $V \setminus S$ are unknown, the quantity $R_V(f)$ cannot be computed directly by an algorithm; instead, it must be estimated from an empirical quantity such as the ranking error of f with respect to the training set S, which we denote by $R_S(f)$ and which can be defined analogously to (1):

$$R_S(f) \quad = \quad \frac{1}{\binom{m}{2}} \sum_{k<l} |y_{i_k} - y_{i_l}| \left(\mathbf{I}_{\{(y_{i_k} - y_{i_l})(f_{i_k} - f_{i_l}) < 0\}} + \frac{1}{2} \mathbf{I}_{\{f_{i_k} = f_{i_l}\}} \right). \tag{2}$$

In the following, we develop a regularization-based algorithmic framework for learning a ranking function f that approximately minimizes $R_V(f)$. Our algorithms minimize regularized versions of a convex upper bound on the training error $R_S(f)$; the regularizers we use encourage smoothness of the learned function with respect to the graph G.

3 Basic Algorithm

Our goal is to find a function $f : V \to \mathbb{R}$ that minimizes a suitably regularized version of the training error $R_S(f)$, *i.e.*, that minimizes a suitable combination of the training error and a regularization term that penalizes complex functions. However, minimizing an objective function that involves $R_S(f)$ is an NP-hard problem, since $R_S(f)$ is a sum of 'discrete' step-function losses of the form

$$\ell_{\text{disc}}(f, i, j) \quad = \quad |y_i - y_j| \left(\mathbf{I}_{\{(y_i - y_j)(f_i - f_j) < 0\}} + \frac{1}{2} \mathbf{I}_{\{f_i = f_j\}} \right). \tag{3}$$

[2]Note that, unlike transductive settings for classification and regression, we choose to measure the performance of a learned ranking function on the complete vertex set V, not just on the set of vertices $V \setminus S$ that do not appear in the training set S. This is because, unlike a classification or regression algorithm that can choose to return the training labels for vertices in the training set S, a ranking algorithm cannot 'rank' the vertices in the training set S correctly just from the given real-valued labels for those vertices; instead, it must use the learned ranking function to rank all the vertices in V relative to each other. (Of course, if desired, one could choose to measure performance with respect to $V \setminus S$; the algorithms we develop would still be applicable.)

Instead, we shall minimize (a regularized version of) a convex upper bound on $R_S(f)$. Several different convex loss functions can be used for this purpose, leading to different algorithmic formulations. We focus on the following ranking loss, which we refer to as the *hinge ranking loss* due to its similarity to the hinge loss used in classification:

$$\ell_{\mathrm{h}}(f,i,j) \;=\; \Big(|y_i - y_j| - (f_i - f_j)\cdot \mathrm{sgn}(y_i - y_j)\Big)_+, \tag{4}$$

where $\mathrm{sgn}(u)$ is 1 if $u>0$, 0 if $u=0$ and -1 if $u<0$, and where a_+ is a if $a>0$ and 0 otherwise. Clearly, $\ell_{\mathrm{h}}(f,i,j)$ is convex in f and upper bounds $\ell_{\mathrm{disc}}(f,i,j)$. We therefore consider minimizing a regularized version of the training ℓ_{h}-error $R_S^{\ell_{\mathrm{h}}}(f)$, which is convex in f and upper bounds $R_S(f)$:

$$R_S^{\ell_{\mathrm{h}}}(f) \;=\; \frac{1}{\binom{m}{2}} \sum_{k<l} \ell_h(f,i_k,i_l)\,. \tag{5}$$

Thus, we want to find a function $f_S : V \to \mathbb{R}$ that solves the following optimization problem for some suitable regularizer $\mathscr{S}(f)$ (and an appropriate regularization parameter $\lambda > 0$):

$$\min_{f:V\to\mathbb{R}} \left\{ R_S^{\ell_{\mathrm{h}}}(f) + \lambda \mathscr{S}(f) \right\}. \tag{6}$$

What would make a good regularizer for real-valued functions defined on the vertices of an undirected graph? It turns out this question has been studied in considerable depth in recent years, and some answers are readily available (Smola and Kondor, 2003; Belkin and Niyogi, 2004; Belkin et al, 2004; Zhou and Schölkopf, 2004; Zhou et al, 2004; Herbster et al, 2005).

A suitable measure of regularization on functions $f : V \to \mathbb{R}$ would be a measure of smoothness with respect to the graph G; in other words, a good function f would be one whose value does not vary rapidly across vertices that are highly similar. It turns out that a regularizer that captures this notion can be derived from the graph Laplacian. The (normalized) Laplacian matrix $\mathbf{L}$ of the graph G is defined as follows: if $\mathbf{W}$ is defined to be the $n \times n$ matrix with (i,j)th entry W_{ij} given by

$$W_{ij} \;=\; \begin{cases} w(i,j) & \text{if } (i,j) \in E \\ 0 & \text{otherwise}, \end{cases} \tag{7}$$

and $\mathbf{D}$ is a diagonal matrix with ith diagonal entry d_i given by

$$d_i \;=\; \sum_{j:(i,j)\in E} w(i,j)\,, \tag{8}$$

then (assuming $d_i > 0\ \forall i$)

$$\mathbf{L} \;=\; \mathbf{D}^{-1/2}(\mathbf{D}-\mathbf{W})\mathbf{D}^{-1/2}\,. \tag{9}$$

The smoothness of a function $f : V \to \mathbb{R}$ with respect to G can then be measured by the following regularizer (recall from Section 2 that we also represent $f : V \to \mathbb{R}$ as $\mathbf{f} \in \mathbb{R}^n$):

$$\mathscr{S}(f) \;=\; \mathbf{f}^T \mathbf{L} \mathbf{f}. \tag{10}$$

To see how the above regularizer measures smoothness, consider first the unnormalized Laplacian $\widetilde{\mathbf{L}}$, which has been used, for example, by Belkin et al (2004); this is defined simply as

$$\widetilde{\mathbf{L}} \;=\; \mathbf{D} - \mathbf{W}\,. \tag{11}$$

If we define $\widetilde{\mathscr{S}}(f)$ analogously to (10) but using $\widetilde{\mathbf{L}}$ instead of $\mathbf{L}$, so that

$$\widetilde{\mathscr{S}}(f) \;=\; \mathbf{f}^T \widetilde{\mathbf{L}} \mathbf{f}, \tag{12}$$

then it is easy to show that

$$\widetilde{\mathscr{S}}(f) \;=\; \frac{1}{2} \sum_{(i,j)\in E} w(i,j)\,(f_i - f_j)^2\,. \tag{13}$$

Thus $\widetilde{\mathscr{S}}(f)$ measures the smoothness of f with respect to the graph G in the following sense: a function f that does not vary rapidly across similar vertices, so that $(f_i - f_j)^2$ is small for $(i,j) \in E$ with large $w(i,j)$, would receive lower values of $\widetilde{\mathscr{S}}(f)$, and would thus be preferred by an algorithm using this quantity as a regularizer. The regularizer $\mathscr{S}(f)$ based on the normalized Laplacian $\mathbf{L}$ plays a similar role, but uses a degree-normalized measure of smoothness; in particular, it can be shown in this case that

$$\mathscr{S}(f) \;=\; \frac{1}{2} \sum_{(i,j)\in E} w(i,j) \left(\frac{f_i}{\sqrt{d_i}} - \frac{f_j}{\sqrt{d_j}} \right)^2\,. \tag{14}$$

Other forms of normalization are also possible; see for example (Johnson and Zhang, 2007) for a detailed analysis.

Putting everything together, our basic algorithm for learning from S a ranking function $f_S : V \to \mathbb{R}$ thus consists of solving the following optimization problem:

$$\min_{f:V\to\mathbb{R}} \left\{ R_S^{\ell_{\mathrm{h}}}(f) + \lambda\, \mathbf{f}^T \mathbf{L} \mathbf{f} \right\}. \tag{15}$$

In practice, the above optimization problem can be solved by reduction to a convex quadratic program, much as is done in support vector machines (SVMs). In particular, introducing a slack variable ξ_{kl} for each pair $1 \le k < l \le m$, we can re-write the above optimization problem as follows:

$$\begin{aligned} &\min_{\mathbf{f}\in\mathbb{R}^n} \left\{ \frac{1}{2} \mathbf{f}^T \mathbf{L} \mathbf{f} \;+\; C \sum_{k<l} \xi_{kl} \right\} \\ &\text{subject to} \\ &\quad \xi_{kl} \;\ge\; |y_{i_k} - y_{i_l}| - (f_{i_k} - f_{i_l}) \cdot \operatorname{sgn}(y_{i_k} - y_{i_l}) \qquad (1 \le k < l \le m) \\ &\quad \xi_{kl} \;\ge\; 0 \qquad\qquad\qquad\qquad\qquad\qquad\quad (1 \le k < l \le m)\,, \end{aligned} \tag{16}$$

where $C = 1/(\lambda m(m-1))$. On introducing Lagrange multipliers α_{kl} and β_{kl} for the above inequalities and formulating the Lagrangian dual (see for example (Boyd and Vandenberghe, 2004) or (Burges, 1998) for a detailed description of the use of this standard technique in SVMs), the above problem further reduces to the following (convex) quadratic program in the $\binom{m}{2}$ variables $\{\alpha_{kl}\}$:

$$\begin{aligned} &\min_{\{\alpha_{kl}\}} \left\{ \frac{1}{2} \sum_{k<l} \sum_{k'<l'} \alpha_{kl} \alpha_{k'l'} \cdot \operatorname{sgn}\left((y_{i_k} - y_{i_l})(y_{i_{k'}} - y_{i_{l'}}) \right) \cdot \phi(k,l,k',l') \;-\; \sum_{k<l} \alpha_{kl} \cdot |y_{i_k} - y_{i_l}| \right\} \\ &\text{subject to} \\ &\quad 0 \le \alpha_{kl} \le C \qquad (1 \le k < l \le m)\,, \end{aligned} \tag{17}$$

where

$$\phi(k,l,k',l') \;=\; L^+_{i_k i_{k'}} - L^+_{i_l i_{k'}} - L^+_{i_k i_{l'}} + L^+_{i_l i_{l'}}\,. \tag{18}$$

Here L^+_{ij} denotes the (i,j)th element of $\mathbf{L}^+$, the pseudo-inverse of $\mathbf{L}$. Note that the Laplacian $\mathbf{L}$ is known to be positive semi-definite, and to not be positive definite (Chung, 1997); this means it has a zero eigenvalue, and is therefore singular (Strang, 1988) (hence the need for the pseudo-inverse). It is also easy to verify from the definition that $\mathbf{D}-\mathbf{W}$ (and therefore $\mathbf{L}$) has rank smaller than n.

It can be shown that, on solving the above quadratic program for $\{\alpha_{kl}\}$, the solution $\mathbf{f}_S \in \mathbb{R}^n$ to the original problem is found as

$$\mathbf{f}_S \;=\; \mathbf{L}^+\mathbf{a}\,, \tag{19}$$

where $\mathbf{a} \in \mathbb{R}^n$ has ith element a_i given by

$$a_i \;=\; \begin{cases} \sum\limits_{l:k<l} \alpha_{kl}\cdot \mathrm{sgn}(y_{i_k}-y_{i_l}) - \sum\limits_{j:j<k} \alpha_{jk}\cdot \mathrm{sgn}(y_{i_j}-y_{i_k}) & \text{if } i=i_k \in S \\ 0 & \text{otherwise}\,. \end{cases} \tag{20}$$

RKHS View The above algorithm can in fact be viewed as performing regularization in a reproducing kernel Hilbert space (RKHS). In particular, let $\mathscr{F}$ be the column-space of $\mathbf{L}^+$, *i.e.*, $\mathscr{F}$ is the set of all vectors in $\mathbb{R}^n$ that can be expressed as a linear combination of the columns of $\mathbf{L}^+$. Recall that the column-space of any symmetric positive semi-definite (PSD) matrix $\mathbf{K} \in \mathbb{R}^{n\times n}$ is an RKHS with $\mathbf{K}$ as its kernel. Since the Laplacian $\mathbf{L}$ is symmetric PSD (Chung, 1997), and since the pseudo-inverse of a symmetric PSD matrix is also symmetric PSD (Strang, 1988), we have that $\mathbf{L}^+$ is symmetric PSD. Consequently, $\mathscr{F}$ is an RKHS with $\mathbf{L}^+$ as its kernel. We shall show now that the algorithm derived above can be viewed as performing regularization within the RKHS $\mathscr{F}$. In order to establish this, we need to show two things: first, that the algorithm always returns a function in $\mathscr{F}$, and second, that the regularizer $\mathscr{S}(f) = \mathbf{f}^T\mathbf{L}\mathbf{f}$ used by the algorithm is equivalent to the (squared) norm of $\mathbf{f}$ in the RKHS $\mathscr{F}$. The first of these follows simply from the form of the solution to the optimization problem in (16); in particular, it is clear from (19) that the solution always belongs to the column-space of $\mathbf{L}^+$. To see the second of these, *i.e.*, the equivalence of the algorithmic regularizer and the RKHS norm, let $f \in \mathscr{F}$; by definition, this means there exists a coefficient vector $\mathbf{c} \in \mathbb{R}^n$ such that $\mathbf{f} = \sum_{i=1}^n c_i \mathbf{L}^+_i$, where $\mathbf{L}^+_i$ denotes the ith column of $\mathbf{L}^+$. Then we have

$$\|\mathbf{f}\|^2_{\mathscr{F}} = \langle \mathbf{f},\mathbf{f}\rangle_{\mathscr{F}} = \sum_{i=1}^n c_i \langle \mathbf{f}, \mathbf{L}^+_i\rangle_{\mathscr{F}} = \sum_{i=1}^n c_i f_i = \mathbf{c}^T\mathbf{f},$$

where the third equality follows from the reproducing property. Furthermore, we have

$$\mathscr{S}(f) = \mathbf{f}^T\mathbf{L}\mathbf{f} = (\mathbf{c}^T\mathbf{L}^+)\mathbf{L}(\mathbf{L}^+\mathbf{c}) = \mathbf{c}^T\mathbf{L}^+\mathbf{c} = \mathbf{c}^T\mathbf{f}.$$

Thus we see that $\mathscr{S}(f) = \|\mathbf{f}\|^2_{\mathscr{F}}$, and therefore our algorithm can be viewed as performing regularization within the RKHS $\mathscr{F}$.

4 Extensions of Basic Algorithm

The RKHS view of the algorithm described in Section 3 raises the natural possibility of using other kernels derived from the graph G in place of the Laplacian-based kernel $\mathbf{L}^+$. We discuss some of these possibilities in Section 4.1; we consider both the case when the weights $w(i,j)$ are derived from a kernel function on the object space, and the case when the weights are simply similarities between objects (that do not necessarily come from a kernel function). In some cases, the similarities may be asymmetric, in which case the graph G must be directed; we discuss this setting in Section 4.2.

4.1 Other Graph Kernels[3]

Consider first the special case when the weights $w(i,j)$ are derived from a kernel function, *i.e.*, when each vertex $i \in V$ is associated with an object x_i in some space X, and there is a kernel function (*i.e.*, a symmetric, positive semi-definite function) $\kappa : X \times X \to \mathbb{R}$ such that for all $i, j \in V$, $(i,j) \in E$ and $w(i,j) = \kappa(x_i, x_j)$. In this case, the weight matrix $\mathbf{W}$, with (i,j)th element $W_{ij} = \kappa(x_i, x_j)$, is symmetric positive semi-definite, and one can simply use $\mathbf{W}$ as the kernel matrix; the resulting optimization problem is equivalent to

$$\min_{f:V\to\mathbb{R}} \left\{ R_S^{\ell_{\mathrm{h}}}(f) + \lambda\, \mathbf{f}^T \mathbf{W}^{-1} \mathbf{f} \right\}, \tag{21}$$

where $\mathbf{W}^{-1}$ denotes the inverse of $\mathbf{W}$ if it exists and the pseudo-inverse otherwise. However, as Johnson and Zhang (2008) show in the context of regression, it can be shown that from the point of view of ranking the objects $\{x_i : i \in V\}$, using the kernel matrix $\mathbf{W}$ as above is equivalent to learning a ranking function $g : X \to \mathbb{R}$ in the standard inductive setting using the kernel function κ, which involves solving the following optimization problem:

$$\min_{g\in\mathscr{F}_\kappa} \left\{ \frac{1}{\binom{m}{2}} \sum_{k<l} \ell_{\mathrm{h}}(g, x_{i_k}, x_{i_l}) + \lambda \left\| g \right\|^2_{\mathscr{F}_\kappa} \right\}, \tag{22}$$

where $\mathscr{F}_\kappa$ denotes the RKHS corresponding to κ, and where (admittedly overloading notation) we use

$$\ell_{\mathrm{h}}(g, x_i, x_j) \;\; = \;\; \Big(|y_i - y_j| - (g(x_i) - g(x_j)) \cdot \mathrm{sgn}(y_i - y_j) \Big)_+ . \tag{23}$$

In particular, we have the following result, which can be proved in exactly the same manner as the corresponding result for regression in (Johnson and Zhang, 2008):

Theorem 1 *Let $\mathbf{W} \in \mathbb{R}^{n\times n}$ be a matrix with (i,j)th entry $W_{ij} = \kappa(x_i, x_j)$ for some kernel function $\kappa : X \times X \to \mathbb{R}$, where for each $i \in V$, $x_i \in X$ is some fixed object associated with i. If $f_S : V \to \mathbb{R}$ is the solution of the transductive ranking method in (21) and $g_S : X \to \mathbb{R}$ is the solution of the inductive ranking method in (22), then for all $i \in V$, we have*

$$f_S(i) \;=\; g_S(x_i)\,.$$

Thus, when the weights $w(i,j)$ are derived from a kernel function κ as above, using the matrix $\mathbf{W}$ as the kernel matrix in a transductive setting does not give any advantage over simply using the kernel function κ in an inductive setting (provided of course that appropriate descriptions of the objects $x_{i_1}, \ldots, x_{i_m} \in X$ corresponding to the training set $S = \{i_1, \ldots, i_m\}$ are available for use in an inductive algorithm). However, one can consider using other kernel matrices derived from $\mathbf{W}$, such as $\mathbf{W}^p$ for $p > 1$, or $\mathbf{W}^{(d)} = \sum_{i=1}^d \mu_i \mathbf{v}_i \mathbf{v}_i^T$ for $d < n$, where $\{(\mu_i, \mathbf{v}_i)\}$ is the eigen-system of $\mathbf{W}$. Johnson and Zhang (2008) give a detailed comparison of these different kernel matrices in the context of transductive methods for regression, and discuss why these kernel choices can give better results in practice than $\mathbf{W}$ itself.

In the more general case, when the weights $w(i,j)$ represent similarities among objects but are not necessarily derived from a kernel function, the weight matrix $\mathbf{W}$ is not necessarily positive semi-definite, and we need to construct from $\mathbf{W}$ a symmetric, positive semi-definite matrix that can be used as a kernel;

[3]Note that in this paper, a graph kernel refers not to a kernel function defined on pairs of objects represented individually as graphs (as considered, for example, by Gärtner et al (2003)), but rather to a kernel function (or kernel matrix) defined on pairs of vertices within a single graph.

indeed, this is exactly what the Laplacian kernel $\mathbf{L}^+$ achieves. In this case also, it is possible to construct other kernel matrices. For example, as is done above with $\mathbf{W}$, one can start with $\mathbf{L}^+$ and use the matrix $(\mathbf{L}^+)^p$ for $p > 1$, which corresponds to using as regularizer $\mathbf{f}^T \mathbf{L}^p \mathbf{f}$ (the case $p = 2$ is discussed in (Belkin et al, 2004)). Similarly, one can use $(\mathbf{L}^+)^{(d)} = \sum_{i=1}^{d} \mu_i \mathbf{v}_i \mathbf{v}_i^T$ for $d < n$, where $\{(\mu_i, \mathbf{v}_i)\}$ is the eigen-system of $\mathbf{L}^+$. Another example of a graph kernel that can be used is the diffusion kernel (Kondor and Lafferty, 2002), defined as

$$e^{-\beta \mathbf{L}} = \lim_{k \to \infty} \left(\mathbf{I}_n - \frac{\beta \mathbf{L}}{k} \right)^k , \tag{24}$$

where $\mathbf{I}_n$ denotes the $n \times n$ identity matrix and $\beta > 0$ is a parameter. For further examples of graph kernels that can be used in the above setting, we refer the reader to (Smola and Kondor, 2003), where several other kernels derived from the graph Laplacian are discussed. Smola and Kondor (2003) also show that any graph-based regularizer that is invariant to permutations of the vertices of the graph must necessarily (up to a constant factor and some trivial additive components) be a function of the Laplacian.

4.2 Directed Graphs

While most similarity measures among objects are symmetric, in some cases, it is possible for similarities to be asymmetric. This can happen, for example, when an asymmetric definition of similarity is used, such as when object i is considered to be similar to object j if i is one of the r objects that are closest to j, for some fixed $r \in \mathbb{N}$ and some distance measure among objects (it is possible that i is one of the r closest objects to j, but j is not among the r objects closest to i). This situation can also arise when the actual definition of similarity used is symmetric, but for computational or other reasons, an asymmetric approximation is used; this was the case, for example, with the similarity scores available for a protein ranking task considered in (Agarwal, 2006). In such cases, the graph $G = (V, E, w)$ must be directed: $(i, j) \in E$ no longer implies $(j, i) \in E$, and even if (i, j) and (j, i) are both in E, $w(i, j)$ is not necessarily equal to $w(j, i)$, so that the weight matrix $\mathbf{W}$ is no longer symmetric.

The case of directed graphs can be treated similarly to the undirected case. In particular, the goal is the same: to find a function $f : V \to \mathbb{R}$ that minimizes a suitably regularized convex upper bound on the training error $R_S(f)$. The convex upper bound on $R_S(f)$ can be chosen to be the same as before, *i.e.*, to be the ℓ_{h}-error $R_S^{\ell_{\mathrm{h}}}(f)$. The goal is then again to solve the optimization problem given in (6), for some suitable regularizer $\mathscr{S}(f)$. This is where the technical difference lies: in the form described so far, the regularizers discussed above apply only to undirected graphs. Indeed, until very recently, the notion of a Laplacian matrix has been associated only with undirected graphs.

Recently, however, an analogue of the Laplacian has been proposed for directed graphs (Chung, 2005). This shares many nice properties with the Laplacian for undirected graphs, and in fact can also be derived via discrete analysis on directed graphs (Zhou et al, 2005). It is defined in terms of a random walk on the given directed graph.

Given a weighted, directed graph $G = (V, E, w)$ with $V = \{1, \ldots, n\}$ as before, let d_i^+ be the out-degree of vertex i:

$$d_i^+ = \sum_{j : (i,j) \in E} w(i, j) . \tag{25}$$

If G is strongly connected and aperiodic, one can consider the standard random walk over G, whose transi-

tion probability matrix $\mathbf{P}$ has (i,j)th entry P_{ij} given by

$$P_{ij} = \begin{cases} \frac{w(i,j)}{d_i^+} & \text{if } (i,j) \in E \\ 0 & \text{otherwise}. \end{cases} \tag{26}$$

In this case, the above random walk has a unique stationary distribution $\pi : V \rightarrow (0,1]$, and the Laplacian $\mathbf{L}$ of G is defined as

$$\mathbf{L} = \mathbf{I}_n - \frac{\Pi^{1/2}\mathbf{P}\Pi^{-1/2} + \Pi^{-1/2}\mathbf{P}^T\Pi^{1/2}}{2}, \tag{27}$$

where Π is a diagonal matrix with $\Pi_{ii} = \pi(i)$. In the case when G is not strongly connected and aperiodic, one can use what is termed a *teleporting* random walk, which effectively allows one to jump uniformly to a random vertex with some small probability η (Zhou et al, 2005); the probability transition matrix $\mathbf{P}^{(\eta)}$ for such a walk has (i,j)th entry given by

$$P_{ij}^{(\eta)} = (1-\eta)P_{ij} + \eta\frac{1}{n-1}\mathbf{I}_{\{i \neq j\}}. \tag{28}$$

Such a teleporting random walk always converges to a unique and positive stationary distribution, and therefore for a general directed graph, one can use as Laplacian a matrix defined similarly to the matrix $\mathbf{L}$ in (27), using $\mathbf{P}^{(\eta)}$ and the corresponding stationary distribution in place of $\mathbf{P}$ and Π.

The Laplacian matrix $\mathbf{L}$ constructed as above is always symmetric and positive semi-definite, and as discussed by Zhou et al (2005), it can be used in exactly the same way as in the undirected case to define a smoothness regularizer $\mathscr{S}(f) = \mathbf{f}^T\mathbf{L}\mathbf{f}$ appropriate for functions defined on the vertices of a directed graph. Thus, the algorithmic framework developed for the undirected case applies in exactly the same manner to the directed case, except for the replacement with the appropriate Laplacian matrix.

As discussed above for the case of undirected graphs, using the above regularizer corresponds to performing regularization in an RKHS with kernel matrix $\mathbf{L}^+$, and again, it is possible to extend the basic framework by using other kernel matrices derived from the (directed) graph instead, such as the matrices $(\mathbf{L}^+)^p$ or $(\mathbf{L}^+)^{(d)}$ described above, for some $p > 1$ and $d < n$ (with $\mathbf{L}$ now corresponding to the directed Laplacian constructed above), or even a directed version of the diffusion kernel, $e^{-\beta\mathbf{L}}$. Other graph kernels defined in terms of the graph Laplacian for undirected graphs (such as those discussed in (Smola and Kondor, 2003)) can be extended to directed graphs in a similar manner.

5 Generalization Bounds

In this section we study generalization properties of our graph-based ranking algorithms. In particular, we are interested in bounding the 'generalization error' $R_V(f_S)$ (see Section 2) of a ranking function $f_S : V \rightarrow \mathbb{R}$ learned from (the labels corresponding to) a training set $S = \{i_1, \ldots, i_m\} \subset V$, assumed to be drawn randomly according to some probability distribution. In transductive models used to study graph-based classification and regression, where one is similarly given labels corresponding to a training set $S = \{i_1, \ldots, i_m\} \subset V$ and the goal is to predict the labels of the remaining vertices, it is common to assume that the vertices in S are selected uniformly at random from V, either with replacement (Blum et al, 2004) or without replacement (Hanneke, 2006; El-Yaniv and Pechyony, 2006; Cortes et al, 2008; Johnson and Zhang, 2008). We consider similar models here for the graph-based ranking problem.

We first consider in Section 5.1 a model in which the vertices in S are selected uniformly at random *with* replacement from V; we make use of some recent results on the stability of kernel-based ranking algorithms

(Agarwal and Niyogi, 2008) to obtain a generalization bound for our algorithms under this model. We then consider in Section 5.2 a model in which the vertices in S are selected uniformly at random *without* replacement from V. Building on recent results of (El-Yaniv and Pechyony, 2006; Cortes et al, 2008) on stability of transductive learning algorithms, we show that stability-based generalization bounds for our ranking algorithms can be obtained under this model too.

5.1 Uniform Sampling With Replacement

Let $\mathscr{U}$ denote the uniform distribution over V, and consider a model in which each of the m vertices in $S = \{i_1, \ldots, i_m\}$ is drawn randomly and independently from V according to $\mathscr{U}$; in other words, S is drawn randomly according to $\mathscr{U}^m$ (note that S in this case may be a multi-set). We derive a generalization bound that holds with high probability under this model. Our bound is derived for the case of a general graph kernel (see Section 4); specific consequences for the Laplacian-based kernel matrix $\mathbf{L}^+$ (as in Section 3) are discussed after giving the general bound. Specifically, let $\mathbf{K} \in \mathbb{R}^{n \times n}$ be any symmetric, positive semi-definite kernel matrix derived from the graph $G = (V, E, w)$ (which could be undirected or directed), and for any (multi-set) $S = \{i_1, \ldots, i_m\} \subset V$, let $f_S : V \to \mathbb{R}$ be the ranking function learned by solving the optimization problem

$$\min_{f: V \to \mathbb{R}} \left\{ R_S^{\ell_h}(f) + \lambda \, \mathbf{f}^T \mathbf{K}^{-1} \mathbf{f} \right\}, \tag{29}$$

where $\mathbf{K}^{-1}$ denotes the inverse of $\mathbf{K}$ if it exists and the pseudo-inverse otherwise. Then we wish to obtain a high-probability bound on the generalization error $R_V(f_S)$.

As discussed in Section 3 for the specific case of the Laplacian kernel, learning a ranking function f_S according to (29) corresponds to performing regularization in the RKHS $\mathscr{F}_{\mathbf{K}}$ comprising of the column-space of $\mathbf{K}$ (in particular, the regularizer $\mathbf{f}^T \mathbf{K}^{-1} \mathbf{f}$ is equivalent to the squared RKHS norm $\|f\|^2_{\mathscr{F}_{\mathbf{K}}}$). Using the notion of algorithmic stability (Bousquet and Elisseeff, 2002), Agarwal and Niyogi (2008) have shown recently that ranking algorithms that perform regularization in an RKHS (subject to some conditions) have good generalization properties. We use these results to obtain a generalization bound for our graph-based ranking algorithm (29) under the model discussed above.

Before describing the results of (Agarwal and Niyogi, 2008) that we use, we introduce some notation. Let X be any domain, and for each $x \in X$, let there be a fixed label $y_x \in [0, M]$ associated with x. Let $f : X \to \mathbb{R}$ a ranking function on X, and let $\ell(f, x, x')$ be a ranking loss. Then for any distribution $\mathscr{D}$ on X, define the expected ℓ-error of f with respect to $\mathscr{D}$ as

$$R^{\ell}_{\mathscr{D}}(f) \;=\; \mathbf{E}_{(x,x') \sim \mathscr{D} \times \mathscr{D}} \left[\ell(f, x, x') \right]. \tag{30}$$

Similarly, for any (multi-set) $S = \{x_1, \ldots, x_m\} \subset X$, define the empirical ℓ-error of f with respect to S as

$$R^{\ell}_{S}(f) \;=\; \frac{1}{\binom{m}{2}} \sum_{k<l} \ell(f, x_k, x_l). \tag{31}$$

Also, define the following ranking losses (again overloading notation):

$$\ell_{\text{disc}}(f,x,x') = |y_x - y_{x'}| \left(\mathbf{I}_{\{(y_x - y_{x'})(f(x)-f(x'))<0\}} + \frac{1}{2}\mathbf{I}_{\{f(x)=f(x')\}} \right). \tag{32}$$

$$\ell_{\text{h}}(f,x,x') = \left(|y_x - y_{x'}| - (f(x)-f(x')) \cdot \text{sgn}(y_x - y_{x'}) \right)_+. \tag{33}$$

$$\ell_1(f,x,x') = \begin{cases} |y_x - y_{x'}|, & \text{if } (f(x)-f(x')) \cdot \text{sgn}(y_x - y_{x'}) \leq 0 \\ 0, & \text{if } (f(x)-f(x')) \cdot \text{sgn}(y_x - y_{x'}) \geq |y_x - y_{x'}| \\ |y_x - y_{x'}| - (f(x)-f(x')) \cdot \text{sgn}(y_x - y_{x'}), & \text{otherwise}. \end{cases} \tag{34}$$

Note that the loss ℓ_1 defined above, while not convex, forms an upper bound on ℓ_{disc}. Finally, define the expected ranking error of f with respect to $\mathscr{D}$ as

$$R_{\mathscr{D}}(f) \equiv R_{\mathscr{D}}^{\ell_{\text{disc}}}(f),$$

and the empirical ranking error of f with respect to S as

$$R_S(f) \equiv R_S^{\ell_{\text{disc}}}(f).$$

In what follows, for any (multi-set) $S = \{x_1, \ldots, x_m\} \subset X$ and any $x_k \in S$, $x'_k \in X$, we shall use $S^{(x_k, x'_k)}$ to denote the (multi-)set obtained from S by replacing x_k with x'_k. The following definition and result are adapted from (Agarwal and Niyogi, 2008):[4]

Definition 1 (Uniform loss stability) *Let $\mathscr{A}$ be a ranking algorithm whose output on a training sample $S \subset X$ we denote by f_S, and let ℓ be a ranking loss function. Let $\beta : \mathbb{N} \to \mathbb{R}$. We say that $\mathscr{A}$ has* uniform loss stability β *with respect to ℓ if for all $m \in \mathbb{N}$, all (multi-sets) $S = \{x_1, \ldots, x_m\} \subset X$ and all $x_k \in S$, $x'_k \in X$, we have for all $x, x' \in X$,*

$$\left| \ell(f_S, x, x') - \ell(f_{S^{(x_k, x'_k)}}, x, x') \right| \leq \beta(m).$$

Theorem 2 (Agarwal and Niyogi (2008)) *Let $\mathscr{A}$ be a ranking algorithm whose output on a training sample $S \subset X$ we denote by f_S, and let ℓ be a bounded ranking loss function such that $0 \leq \ell(f,x,x') \leq B$ for all $f : X \to \mathbb{R}$ and $x, x' \in X$. Let $\beta : \mathbb{N} \to \mathbb{R}$ be such that $\mathscr{A}$ has uniform loss stability β with respect to ℓ. Then for any distribution $\mathscr{D}$ over X and any $0 < \delta < 1$, with probability at least $1 - \delta$ over the draw of S according to $\mathscr{D}^m$, the expected ℓ-error of the learned function f_S is bounded by*

$$R_{\mathscr{D}}^{\ell}(f_S) < R_S^{\ell}(f_S) + 2\beta(m) + (m\beta(m) + B)\sqrt{\frac{2}{m} \ln\left(\frac{1}{\delta}\right)}.$$

The above result shows that ranking algorithms with good stability properties have good generalization behaviour. Agarwal and Niyogi (2008) further show that ranking algorithms that perform regularization in an RKHS have good stability with respect to the loss ℓ_1:[5]

[4] Agarwal and Niyogi (2008) consider a more general setting where the label y_x associated with an instance $x \in X$ may be random; the definitions and results given here are stated for the special case of fixed labels.

[5] The result stated here is a special case of the original result, stated for the hinge ranking loss.

Theorem 3 (Agarwal and Niyogi (2008)) *Let $\mathscr{F}$ be an RKHS consisting of real-valued functions on a domain X, with kernel $\kappa : X \times X \to \mathbb{R}$ such that $\kappa(x,x) \leq \kappa_{\max} < \infty \ \forall\, x \in X$. Let $\lambda > 0$, and let $\mathscr{A}$ be a ranking algorithm that, given a training sample $S \subset X$, learns a ranking function $f_S \in \mathscr{F}$ by solving the optimization problem*

$$\min_{f \in \mathscr{F}} \left\{ R_S^{\ell_h}(f) + \lambda \|f\|_{\mathscr{F}}^2 \right\}.$$

Then $\mathscr{A}$ has uniform loss stability β with respect to the ranking loss ℓ_1, where for all $m \in \mathbb{N}$,

$$\beta(m) \;=\; \frac{16\kappa_{\max}}{\lambda m}.$$

In order to apply the above results to our graph-based setting, where the domain X is the finite vertex set V, let us note that the ℓ_1 loss in this case becomes (for a ranking function $f : V \to \mathbb{R}$ and vertices $i, j \in V$)

$$\ell_1(f,i,j) \;=\; \begin{cases} |y_i - y_j|, & \text{if } (f_i - f_j)\cdot \operatorname{sgn}(y_i - y_j) \;\leq\; 0 \\ 0, & \text{if } (f_i - f_j)\cdot \operatorname{sgn}(y_i - y_j) \;\geq\; |y_i - y_j| \\ |y_i - y_j| - (f_i - f_j)\cdot \operatorname{sgn}(y_i - y_j), & \text{otherwise}, \end{cases} \tag{35}$$

and that the training ℓ_1-error of f with respect to $S = \{i_1, \ldots, i_m\} \subset V$ becomes

$$R_S^{\ell_1}(f) \;=\; \frac{1}{\binom{m}{2}} \sum_{k<l} \ell_1(f, i_k, i_l). \tag{36}$$

Then we have the following generalization result:

Theorem 4 *Let $\mathbf{K} \in \mathbb{R}^{n \times n}$ be a symmetric positive semi-definite matrix, and let $K_{\max} = \max_{1 \leq i \leq n} \{K_{ii}\}$. Let $\lambda > 0$, and for any (multi-set) $S = \{i_1, \ldots, i_m\} \subset V$, let f_S be the ranking function learned by solving the optimization problem (29). Then for any $0 < \delta < 1$, with probability at least $1 - \delta$ over the draw of S according to $\mathscr{U}^m$, the generalization error of the learned function f_S is bounded by*

$$R_V(f_S) \;<\; \left(1 + \frac{1}{n-1}\right)\left(R_S^{\ell_1}(f_S) + \frac{32K_{\max}}{\lambda m} + \left(\frac{16K_{\max}}{\lambda} + M\right)\sqrt{\frac{2}{m}\ln\left(\frac{1}{\delta}\right)}\right).$$

Proof By Theorem 3, the graph-based ranking algorithm that learns a ranking function by solving the optimization problem (29) has uniform loss stability β with respect to the loss ℓ_1, where

$$\beta(m) \;=\; \frac{16K_{\max}}{\lambda m}.$$

Noting that ℓ_1 is bounded as $0 \leq \ell_1(f,i,j) \leq M$ for all $f : V \to \mathbb{R}$ and $i, j \in V$, we can therefore apply Theorem 2 to the above algorithm and to the uniform distribution $\mathscr{U}$ over V to obtain that for any $0 < \delta < 1$, with probability at least $1 - \delta$ over the draw of S according to $\mathscr{U}^m$,

$$R_{\mathscr{U}}^{\ell_1}(f_S) \;<\; R_S^{\ell_1}(f_S) + \frac{32K_{\max}}{\lambda m} + \left(\frac{16K_{\max}}{\lambda} + M\right)\sqrt{\frac{2}{m}\ln\left(\frac{1}{\delta}\right)}.$$

Now, since $\ell_{\text{disc}}(f,i,j) \leq \ell_1(f,i,j)$, we have

$$R_{\mathscr{U}}(f_S) \;\leq\; R_{\mathscr{U}}^{\ell_1}(f_S),$$

which gives that with probability at least $0 < \delta < 1$ as above,

$$R_{\mathscr{U}}(f_S) \;<\; R_S^{\ell_1}(f_S) + \frac{32K_{\max}}{\lambda m} + \left(\frac{16K_{\max}}{\lambda} + M\right)\sqrt{\frac{2}{m}\ln\left(\frac{1}{\delta}\right)}.$$

The result follows by observing that since $\ell_{\mathrm{disc}}(f,i,i) = 0$ for all i,

$$R_V(f_S) \;=\; \frac{1}{\binom{n}{2}}\frac{n^2}{2}R_{\mathscr{U}}(f_S) \;=\; \left(1+\frac{1}{n-1}\right)R_{\mathscr{U}}(f_S).$$

□

Remark Note that the factor of $(1+\frac{1}{n-1})$ in the above result is necessary only because we choose to measure the generalization error by $R_V(f)$; if we chose to measure it by $R_{\mathscr{U}}(f)$, this factor would be unnecessary.

In the case of the Laplacian kernel $\mathbf{L}^+$ for a connected, undirected graph $G = (V,E,w)$, one can bound L_{ii}^+ in terms of the (unweighted) diameter of the graph and properties of the weight function w:

Theorem 5 *Let $G = (V,E,w)$ be a connected, weighted, undirected graph, and let $\mathbf{L}$ be the (normalized) Laplacian matrix of G. Let $d = \max_{1\le i\le n} d_i$ and $w_{\min} = \min_{(i,j)\in E} w(i,j)$, and let ρ be the unweighted diameter of G, i.e., the length (number of edges) of the longest path between any two vertices i and j in V. Then for all $1 \le i \le n$,*

$$L_{ii}^+ \;\le\; \frac{\rho d}{w_{\min}}.$$

The proof of the above result is based on the proof of a similar result of (Herbster et al, 2005), which was given for the unnormalized Laplacian of an unweighted graph; details are provided in Appendix A. Combining the above result with Theorem 4, we get the following generalization bound in this case:

Corollary 1 *Let $G = (V,E,w)$ be a connected, weighted, undirected graph, and let $\mathbf{L}$ be the (normalized) Laplacian matrix of G. Let $d = \max_{1\le i\le n} d_i$ and $w_{\min} = \min_{(i,j)\in E} w(i,j)$, and let ρ be the unweighted diameter of G. Let $\lambda > 0$, and for any (multi-set) $S = \{i_1,\ldots,i_m\} \subset V$, let f_S be the ranking function learned by solving the optimization problem (15). Then for any $0 < \delta < 1$, with probability at least $1-\delta$ over the draw of S according to $\mathscr{U}^m$, the generalization error of the learned function f_S is bounded by*

$$R_V(f_S) \;<\; \left(1+\frac{1}{n-1}\right)\left(R_S^{\ell_1}(f_S) + \frac{32\rho d}{\lambda m w_{\min}} + \left(\frac{16\rho d}{\lambda w_{\min}} + M\right)\sqrt{\frac{2}{m}\ln\left(\frac{1}{\delta}\right)}\right).$$

Proof Follows immediately from Theorems 4 and 5.

□

5.2 Uniform Sampling Without Replacement

Consider now a model in which the vertices in $S = \{i_1,\ldots,i_m\}$ are drawn uniformly at random from V but without replacement; in other words, S is drawn randomly according to $\mathscr{T}_m$, the uniform distribution over all the $\binom{n}{m}$ subsets of V of size m. This model has been used to study generalization properties of transductive learning methods for classification and regression (Hanneke, 2006; El-Yaniv and Pechyony, 2006; Cortes et al, 2008; Johnson and Zhang, 2008). In particular, Hanneke (2006) obtains a generalization bound for

graph-based classification algorithms under this model in terms of graph cuts; El-Yaniv and Pechyony (2006) and Cortes et al (2008) obtain bounds for transductive classification and regression algorithms, respectively, based on the notion of algorithmic stability. Johnson and Zhang (2008) also use algorithmic stability in their study of generalization properties of graph-based regression algorithms; however the bounds they derive hold in expectation over the draw of the training sample rather than with high probability.

Obtaining stability-based bounds that hold with high probability under the above model is more difficult since the vertices in S are no longer independent; most stability-based bounds, such as those derived in (Bousquet and Elisseeff, 2002) or that of Theorem 2 in the previous section, rely on McDiarmid's bounded differences inequality (McDiarmid, 1989) which applies to functions of independent random variables. However, in an elegant piece of work, El-Yaniv and Pechyony (2006) recently derived an analogue of McDiarmid's inequality that is applicable specifically to functions of random variables drawn without replacement from a finite sample, and used this to obtain stability-based bounds for transductive classification algorithms under the above model; a similar result was used by Cortes et al (2008) to obtain such bounds for transductive regression algorithms. Here we extend these results to obtain stability-based generalization bounds for our graph-based ranking algorithms under the above model.

We start with a slightly different notion of stability defined for (graph-based) transductive algorithms; in what follows, $V = \{1,\ldots,n\}$ is the set of vertices as before, $S = \{i_1,\ldots,i_m\} \subset V$ represents a subset of V of size m (in this section S will always be a subset; it can no longer be a multi-set), and for $i_k \in S$, $i'_k \in V \setminus S$, we denote $S^{(i_k,i'_k)} = (S \setminus \{i_k\}) \cup \{i'_k\}$.

Definition 2 (Uniform transductive loss stability) *Let $\mathscr{A}$ be a transductive ranking algorithm whose output on a training set $S \subset V$ we denote by f_S, and let ℓ be a ranking loss function. Let $\beta : \mathbb{N} \to \mathbb{R}$. We say that $\mathscr{A}$ has* uniform transductive loss stability β *with respect to ℓ if for all $m \in \mathbb{N}$, all subsets $S = \{i_1,\ldots,i_m\} \subset V$ and all $i_k \in S$, $i'_k \in V \setminus S$, we have for all $i, j \in V$,*

$$\left|\ell(f_S, i, j) - \ell(f_{S^{(i_k,i'_k)}}, i, j)\right| \quad \leq \quad \beta(m)\,.$$

Note that if a (graph-based) transductive ranking algorithm has uniform loss stability β with respect to a loss ℓ (in the sense of Definition 1 in the previous section), then it also has uniform *transductive* loss stability β with respect to ℓ. In particular, by virtue of Theorem 3, we immediately have the following:

Theorem 6 *Let $\mathbf{K} \in \mathbb{R}^{n\times n}$ be a symmetric positive semi-definite matrix, and let $K_{\max} = \max_{1\leq i\leq n}\{K_{ii}\}$. Let $\lambda > 0$, and let $\mathscr{A}$ be the graph-based transductive ranking algorithm that, given a training set $S \subset V$, learns a ranking function $f_S : V \to \mathbb{R}$ by solving the optimization problem (29). Then $\mathscr{A}$ has uniform transductive loss stability β with respect to the ranking loss ℓ_1, where for all $m \in \mathbb{N}$,*

$$\beta(m) \quad = \quad \frac{16K_{\max}}{\lambda m}\,.$$

Now, using the concentration inequality of El-Yaniv and Pechyony (2006) and arguments similar to those in (Agarwal and Niyogi, 2008), we can establish the following analogue of Theorem 2 for (graph-based) transductive ranking algorithms with good transductive loss stability:[6]

[6]Note that the stability and generalization results in this section apply to transductive ranking algorithms learning over any finite domain X, not necessarily graph-based algorithms learning over a vertex set V; we restrict our exposition to graph-based algorithms for simplicity of notation.

Theorem 7 *Let $\mathscr{A}$ be a transductive ranking algorithm whose output on a training set $S \subset V$ we denote by f_S, and let ℓ be a bounded ranking loss function such that $0 \leq \ell(f,i,j) \leq B$ for all $f : V \to \mathbb{R}$ and $i, j \in V$. Let $\beta : \mathbb{N} \to \mathbb{R}$ be such that $\mathscr{A}$ has uniform transductive loss stability β with respect to ℓ. Then for any $0 < \delta < 1$, with probability at least $1-\delta$ over the draw of S according to $\mathscr{T}_m$, the generalization ℓ-error of the learned function f_S is bounded by*

$$R_V^{\ell}(f_S) \quad < \quad R_S^{\ell}(f_S) + \frac{4(n-m)}{n}\beta(m) + 2(m\beta(m)+B)\sqrt{\frac{2(n-m)}{mn}\ln\left(\frac{1}{\delta}\right)}.$$

Details of the proof are provided in Appendix B. Combining the above result with Theorem 6, we then have the following generalization result for our algorithms:

Theorem 8 *Let $\mathbf{K} \in \mathbb{R}^{n \times n}$ be a symmetric positive semi-definite matrix, and let $K_{\max} = \max_{1 \leq i \leq n}\{K_{ii}\}$. Let $\lambda > 0$, and for any $S = \{i_1, \ldots, i_m\} \subset V$, let f_S be the ranking function learned by solving the optimization problem (29). Then for any $0 < \delta < 1$, with probability at least $1-\delta$ over the draw of S according to $\mathscr{T}_m$, the generalization error of the learned function f_S is bounded by*

$$R_V(f_S) \quad < \quad R_S^{\ell_1}(f_S) + \frac{64K_{\max}(n-m)}{\lambda mn} + 2\left(\frac{16K_{\max}}{\lambda} + M\right)\sqrt{\frac{2(n-m)}{mn}\ln\left(\frac{1}{\delta}\right)}.$$

Proof Noting that ℓ_1 is bounded as $0 \leq \ell_1(f,i,j) \leq M$ for all $f : V \to \mathbb{R}$ and $i, j \in V$, we have from Theorems 6 and 7 that

$$R_V^{\ell_1}(f_S) \quad < \quad R_S^{\ell_1}(f_S) + \frac{64K_{\max}(n-m)}{\lambda mn} + 2\left(\frac{16K_{\max}}{\lambda} + M\right)\sqrt{\frac{2(n-m)}{mn}\ln\left(\frac{1}{\delta}\right)}.$$

The result follows by observing that $R_V(f_S) \leq R_V^{\ell_1}(f_S)$.

□

As in Section 5.1, we can combine the above result with Theorem 5 to get the following bound in the case of the Laplacian kernel for a connected, undirected graph:

Corollary 2 *Let $G = (V, E, w)$ be a connected, weighted, undirected graph, and let $\mathbf{L}$ be the (normalized) Laplacian matrix of G. Let $d = \max_{1 \leq i \leq n} d_i$ and $w_{\min} = \min_{(i,j) \in E} w(i,j)$, and let ρ be the unweighted diameter of G. Let $\lambda > 0$, and for any $S = \{i_1, \ldots, i_m\} \subset V$, let f_S be the ranking function learned by solving the optimization problem (15). Then for any $0 < \delta < 1$, with probability at least $1-\delta$ over the draw of S according to $\mathscr{T}_m$, the generalization error of the learned function f_S is bounded by*

$$R_V(f_S) \quad < \quad R_S^{\ell_1}(f_S) + \frac{64\rho d(n-m)}{\lambda mn w_{\min}} + 2\left(\frac{16\rho d}{\lambda w_{\min}} + M\right)\sqrt{\frac{2(n-m)}{mn}\ln\left(\frac{1}{\delta}\right)}.$$

Proof Follows immediately from Theorems 8 and 5.

□

Table 1: Ranking labels assigned to digit images. Since the goal is to rank the digits in ascending order, with 0s at the top and 9s at the bottom, the labels assigned to 0s are highest and those assigned to 9s the lowest.

Digit	0	1	2	3	4	5	6	7	8	9
Label y	10	9	8	7	6	5	4	3	2	1

Table 2: Distribution of the 10 digits in the set of 2,000 images used in our experiments.

Digit	0	1	2	3	4	5	6	7	8	9
Number of instances	207	230	198	207	194	169	202	215	187	191

6 Experiments

We evaluated our graph-based ranking algorithms on two popular data sets frequently used in the study of learning algorithms in transductive and semi-supervised settings: the MNIST data set consisting of images of handwritten digits, and the 20 newsgroups data set consisting of documents from various newsgroups. While these data sets are represented as similarity graphs in a transductive setting for our purposes, the objects in both data sets (images in the first and newsgroup documents in the second) can also be represented as vectors in appropriate Euclidean spaces, allowing us to compare our results with those obtained using the state-of-the-art RankBoost algorithm (Freund et al, 2003) in an inductive setting.

6.1 Handwritten Digit Ranking – MNIST Data

The MNIST data set[7] consists of images of handwritten digits labeled from 0 to 9. The data set has typically been used to evaluate algorithms for (multi-class) classification, where the classification task is to classify images according to their digit labels. Here we consider a ranking task in which the goal is to rank the images in ascending order by digits; in other words, the goal is to rank the 0s at the top, followed by the 1s, then the 2s, and so on, with the 9s at the bottom. Accordingly, we assign ranking labels y to images such that images of 0s are assigned the highest label, images of 1s the next highest, and so on, with images of 9s receiving the lowest label. The specific labels assigned in our experiments are shown in Table 1.

The original MNIST data contains 60,000 training images and 10,000 test images. In our experiments, we used a subset of 2,000 images; these were taken from the 'more difficult' half of the images in the original test set (specifically, images 8,001–10,000 of the original test set). The distribution of the 10 digits in these 2,000 images is shown in Table 2.

Each image in the MNIST data is a 28×28 grayscale image, and can therefore be represented as a vector in $\mathbb{R}^{784}$. A popular method for constructing a similarity graph for MNIST data is to use a nearest-neighbor approach based on Euclidean distances between these vectors. We constructed a 25-nearest-neighbor graph over the 2,000 images, in which an edge from image i to image j was included if image j was among the 25 nearest neighbors of image i (by Euclidean distance); for each such edge (i, j), we set $w(i, j) = 1$. This led to a directed graph, which formed our data representation in the transductive setting. As described in Section 4.2, we used a teleporting random walk (with $\eta = 0.01$) to construct the graph Laplacian $\mathbf{L}$; the resulting Laplacian kernel $\mathbf{L}^+$ was then used in our graph-based ranking algorithm.

[7] Available at `http://yann.lecun.com/exdb/mnist/`

For comparison, we implemented the RankBoost algorithm of Freund et al (2003) in an inductive setting, using the vector representations of the images. In this setting, the algorithm receives the vectors in $\mathbb{R}^{784}$ corresponding to the training images (along with their ranking labels as described in Table 1), but no information about the remaining images; the algorithm then learns a ranking function $f : \mathbb{R}^{784} \to \mathbb{R}$. In our experiments, we used threshold rankers with range $\{0,1\}$ (similar to boosted stumps; see (Freund et al, 2003)) as weak rankings.

The results are shown in Figure 1. Experiments were conducted with varying numbers of labeled examples; the results for each number are averaged over 10 random trials (in each trial, a training set of the desired size was selected randomly from the set of 2,000 images, subject to containing equal numbers of images of all digits; this reflected the roughly uniform distribution of digits in the data set). Error bars show standard error. We used two evaluation measures: the ranking error as defined in Eqs. (1-2), and the Spearman rank correlation coefficient, which measures the correlation between a learned ranking and the true ranking defined by the y labels. The top panel of Figure 1 shows these measures evaluated on the complete set of 2,000 images (recall from Section 2 that in our transductive ranking setting we wish to measure ranking performance on the complete vertex set). We also show in the bottom panel of the figure the above measures evaluated on only the unlabeled data for each trial. Note that the ranking error is not necessarily bounded between 0 and 1; as can be seen from the definition, it is bounded between 0 and the average (absolute) difference between ranking labels across all pairs in the data set used for evaluation. In our case, for the complete set of 2,000 images, this upper bound is 3.32. The Spearman rank correlation coefficient lies between -1 and 1, with larger positive values representing a stronger positive correlation. The parameter C in the graph ranking algorithm was selected from the set $\{0.01, 0.1, 1, 10, 100\}$ using 5-fold cross validation in each trial. The RankBoost algorithm was run for 100 rounds in each trial (increasing the number of rounds further did not yield any improvement in performance).

As can be seen, even though the similarity graph used in the graph ranking algorithm is derived from the same vector respresentation as used in RankBoost, the graph ranking approach leads to a significant improvement in performance. This can be attributed to the fact that the graph ranking approach operates in a transductive setting where information about the objects to which the learned ranking is to be applied is available in the form of similarity measurements, whereas the RankBoost algorithm operates in an inductive setting where no such information is provided. This suggests that in application domains where the instance space is finite and known in advance, exploiting this knowledge in the form of an appropriate similarity graph over the instances can improve prediction over standard inductive learning.

6.2 Document Ranking – 20 Newsgroups Data

The 20 newsgroups data set[8] consists of documents comprised of newsgroup messages, classified according to newsgroup. We used the 'mini' version of the data set in our experiments, which contains a total of 2,000 messages, 100 each from 20 different newsgroups. These newsgroups can be grouped together into categories based on subject matter, allowing for a hierarchical classification. This leads to a natural ranking task associated with any target newsgroup: documents from the given newsgroup are to be ranked highest, followed by documents from other newsgroups in the same category, followed finally by documents in other categories. In particular, we categorized the 20 newsgroups as shown in Table 3[9], and chose the `alt.atheism` newsgroup as our target. The ranking labels y assigned to documents in the resulting ranking task are shown in Table 4.

[8] Available at `www.ics.uci.edu/~kdd/databases/20newsgroups/20newsgroups.html`

[9] This categorization was taken from `http://people.csail.mit.edu/jrennie/20Newsgroups/`

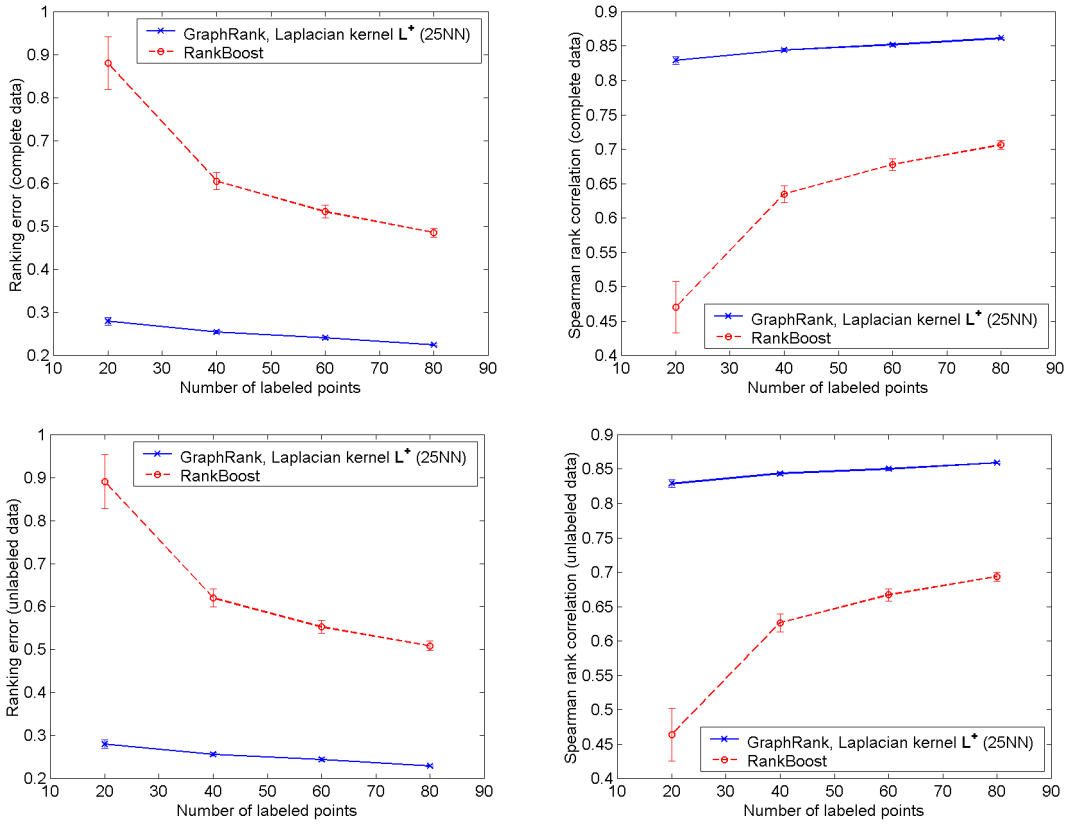

Figure 1: Comparison of our graph ranking algorithm (labeled GraphRank) with RankBoost on the task of ranking MNIST images in ascending order by digits. The graph ranking algorithm operates in a transductive setting and uses a Laplacian kernel derived from a 25-nearest-neighbor (25NN) similarity graph over the images; RankBoost operates in an inductive setting and uses the vector representations of the images. The left plots show ranking error; the right plots show Spearman rank correlation. Each point is an average over 10 random trials; error bars show standard error. The plots in the top panel show performance on the complete set of 2,000 images; those in the bottom panel show performance on unlabeled data only. (See text for details.)

Following Belkin and Niyogi (2004), we tokenized the documents using the Rainbow software package (McCallum, 1996), using a stop list of approximately 500 common words and removing message headers. The vector representation of each message then consisted of the counts of the most frequent 6,000 words, normalized so as to sum to 1. The graph representation of the data was derived from the resulting document vectors; in particular, we constructed an undirected similarity graph over the 2,000 documents using Gaussian/RBF similarity weights given by $w(i,j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2)$, where $\mathbf{x}_i \in \mathbb{R}^{6,000}$ denotes the vector representation of document i. Since the resulting weight matrix $\mathbf{W}$ is positive semi-definite, it can be used directly as the kernel matrix in our graph ranking algorithm. However, as discussed in Section 4.1, this is equivalent to using an inductive (kernel-based) learning method with kernel function $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/2)$. An alternative is to use a positive semi-definite matrix derived from $\mathbf{W}$; in our experiments we used $\mathbf{W}^{(25)}$ which, as described in Section 4.1, is given by $\mathbf{W}^{(25)} = \sum_{i=1}^{25} \mu_i \mathbf{v}_i \mathbf{v}_i^T$ (where

Table 3: Categorisation of the 20 newsgroups based on subject matter.

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.guns talk.politics.mideast talk.politics.misc	alt.atheism soc.religion.christian talk.religion.misc

Table 4: Ranking labels assigned to newsgroup documents. The alt.atheism newsgroup was chosen as the target, to be ranked highest.

Newsgroup	**Label** y	**Newsgroup**	**Label** y
alt.atheism	3	rec.sport.hockey	1
comp.graphics	1	sci.crypt	1
comp.os.ms-windows.misc	1	sci.electronics	1
comp.sys.ibm.pc.hardware	1	sci.med	1
comp.sys.mac.hardware	1	sci.space	1
comp.windows.x	1	soc.religion.christian	2
misc.forsale	1	talk.politics.guns	1
rec.autos	1	talk.politics.mideast	1
rec.motorcycles	1	talk.politics.misc	1
rec.sport.baseball	1	talk.religion.misc	2

$\{(\mu_i, \mathbf{v}_i)\}$ is the eigen-system of $\mathbf{W}$). Again, for comparison, we also implemented the RankBoost algorithm in an inductive setting, using the vector representations of the documents in $\mathbb{R}^{6,000}$. As in the MNIST experiments, we used threshold rankers with range $\{0,1\}$ as weak rankings.

The results are shown in Figure 2. As before, the results for each number of labeled examples are averaged over 10 random trials (random choices of training set, subject to containing equal numbers of documents from all newsgroups). Again, error bars show standard error. In this case, for the complete set of 2,000 documents, the ranking error is bounded between 0 and 0.35. The parameter C in the graph ranking algorithm was selected as before from the set $\{0.01, 0.1, 1, 10, 100\}$ using 5-fold cross validation in each trial. The RankBoost algorithm was run for 100 rounds in each trial (again, increasing the number of rounds further did not yield any improvement in performance).

There are two observations to be made. First, the graph ranking algorithm with RBF kernel $\mathbf{W}$, which effectively operates in the same inductive setting as the RankBoost algorithm, significantly outperforms RankBoost (at least with the form of weak rankings used in our implementation; these are the same as the weak rankings used by Freund et al (2003)). Second, the transductive method obtained by using $\mathbf{W}^{(25)}$ as the kernel matrix improves performance over $\mathbf{W}$ when the number of labeled examples is small. As one might expect, this suggests that the value of information about unlabeled data is greatest when the number of labeled examples is small.

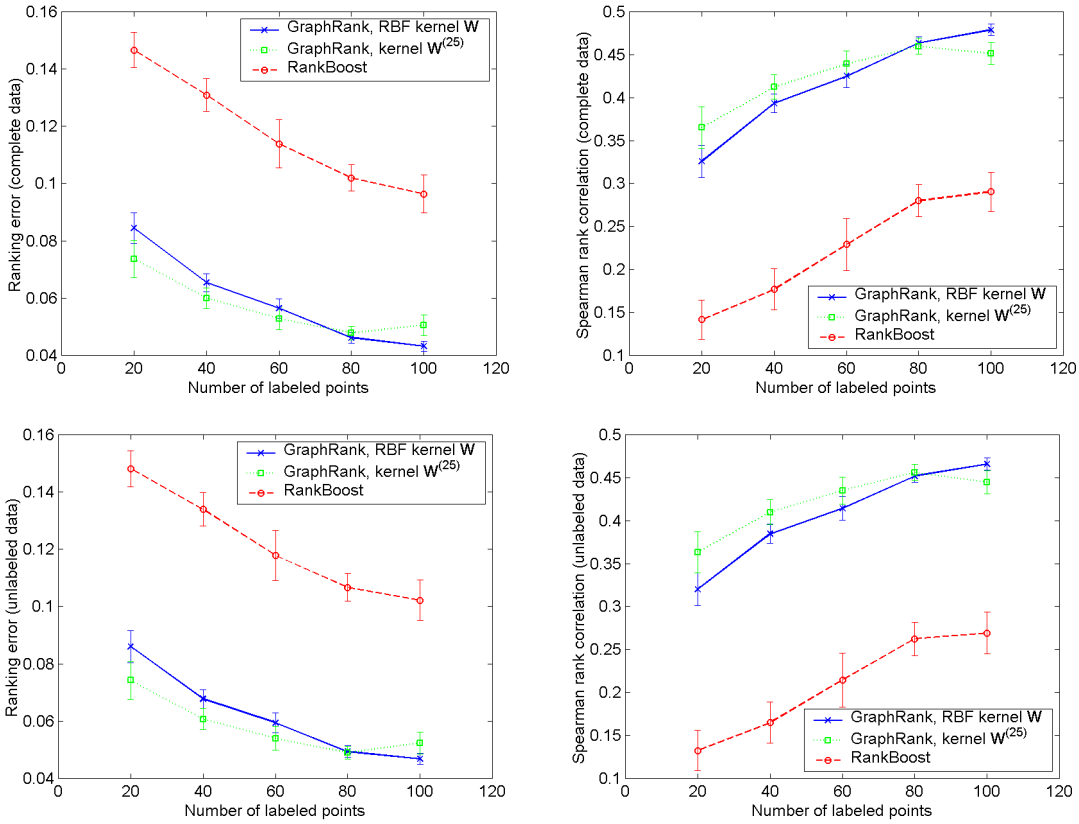

Figure 2: Comparison of our graph ranking algorithm (labeled GraphRank) with RankBoost on the task of ranking newsgroup documents, with the `alt.atheism` newsgroup as target. The graph ranking algorithm with RBF kernel $\mathbf{W}$ effectively operates in an inductive setting, as does RankBoost; GraphRank with $\mathbf{W}^{(25)}$ as kernel operates in a transductive setting. The left plots show ranking error; the right plots show Spearman rank correlation. Each point is an average over 10 random trials; error bars show standard error. The plots in the top panel show performance on the complete set of 2,000 documents; those in the bottom panel show performance on unlabeled data only. (See text for details.)

7 Discussion

Our goal in this paper has been to develop ranking algorithms in a transductive, graph-based setting, where the instance space is finite and is represented in the form of a similarity graph. Building on recent developments in regularization theory for graphs and corresponding Laplacian-based methods for classification and regression, we have developed an algorithmic framework for learning ranking functions on such graphs.

Our experimental results show that when the instance space is finite and known in advance, exploiting this knowledge in the form of an appropriate similarity graph over the instances can improve prediction over standard inductive learning. While the ranking tasks in our experiments are chosen from application domains where such comparisons with inductive learning can be made, the value of our algorithms is likely to be even greater for ranking tasks in application domains where the data naturally comes in the form of pair-wise similarities (as is often the case, for example, in computational biology applications, where

pair-wise similarities between biological sequences are provided); in such cases, existing inductive learning methods cannot always be applied.

Our algorithms have an SVM-like flavour in their formulations; indeed, they can be viewed as minimizing a regularized ranking error within a reproducing kernel Hilbert space (RKHS). From a theoretical standpoint, this means that they benefit from theoretical results such as those establishing stability and generalization properties of algorithms that perform regularization within an RKHS. From a practical standpoint, it means that the implementation of these algorithms can benefit from the large variety of techniques that have been developed for scaling SVMs to large data sets (*e.g.*, (Joachims, 1999; Platt, 1999)).

We have focused in this paper on a particular setting of the ranking problem where order relationships among objects are indicated by (differences among) real-valued labels associated with the objects. However the framework we have developed can be used also in (transductive versions of) other ranking settings, such as when order relationships are provided in the form of explicit pair-wise preferences (see for example (Cohen et al, 1999; Freund et al, 2003) for early studies of this form of ranking problem in an inductive setting).

Acknowledgements

The author would like to thank Partha Niyogi for stimulating discussions on many topics related to this work, and Mikhail Belkin for useful pointers. This work was supported in part by NSF award DMS-0732334.

A Proof of Theorem 5

The proof is based on the proof of a similar result of (Herbster et al, 2005), which was given for the unnormalized Laplacian of an unweighted graph.

Proof [of Theorem 5] Since $\mathbf{L}^+$ is positive semi-definite, we have $L^+_{ii} \geq 0$. If $L^+_{ii} = 0$, the result holds trivially. Therefore assume $L^+_{ii} > 0$. Then $\exists\ j$ such that $L^+_{ij} < 0$ (since for all i, $\sum_{j=1}^n L^+_{ij}\sqrt{d_j} = 0$; this is due to the fact that the vector $(\sqrt{d_1},\ldots,\sqrt{d_n})^T$ is an egienvector of $\mathbf{L}^+$ with eigenvalue 0). Let Q_{ij} denote (the set of edges in) the shortest path in G from i to j (shortest in terms of number of edges; such a path exists since G is connected); let r be the number of edges in this path. Since $\|\mathbf{a}\|_1 \leq \sqrt{r}\|\mathbf{a}\|_2$ for any $\mathbf{a} \in \mathbb{R}^r$, we have

$$\sum_{(u,v)\in Q_{ij}} \left(\frac{L^+_{iu}}{\sqrt{d_u}} - \frac{L^+_{iv}}{\sqrt{d_v}}\right)^2 \;\geq\; \frac{1}{r}\left(\sum_{(u,v)\in Q_{ij}} \left|\frac{L^+_{iu}}{\sqrt{d_u}} - \frac{L^+_{iv}}{\sqrt{d_v}}\right|\right)^2. \tag{37}$$

Now, we have

$$\begin{aligned}
\sum_{(u,v)\in Q_{ij}} \left|\frac{L^+_{iu}}{\sqrt{d_u}} - \frac{L^+_{iv}}{\sqrt{d_v}}\right| &\geq \sum_{(u,v)\in Q_{ij}} \left(\frac{L^+_{iu}}{\sqrt{d_u}} - \frac{L^+_{iv}}{\sqrt{d_v}}\right) \\
&= \frac{L^+_{ii}}{\sqrt{d_i}} - \frac{L^+_{ij}}{\sqrt{d_j}} \\
&> \frac{L^+_{ii}}{\sqrt{d_i}},
\end{aligned} \tag{38}$$

where the equality follows since all other terms in the sum cancel out, and the last inequality follows since $L^+_{ij} < 0$. Furthermore, we have

$$\begin{aligned} L^+_{ii} &= (\mathbf{L}^+_i)^T \mathbf{L} (\mathbf{L}^+_i) \\ &= \frac{1}{2} \sum_{(u,v)\in E} w(u,v) \left(\frac{L^+_{iu}}{\sqrt{d_u}} - \frac{L^+_{iv}}{\sqrt{d_v}} \right)^2 \\ &\geq \frac{1}{2} \cdot 2 \sum_{(u,v)\in Q_{ij}} w(u,v) \left(\frac{L^+_{iu}}{\sqrt{d_u}} - \frac{L^+_{iv}}{\sqrt{d_v}} \right)^2 \\ &\geq w_{\min} \sum_{(u,v)\in Q_{ij}} \left(\frac{L^+_{iu}}{\sqrt{d_u}} - \frac{L^+_{iv}}{\sqrt{d_v}} \right)^2 , \end{aligned} \tag{39}$$

where the second equality follows from Eq. (14) (applied to $\mathbf{f} = \mathbf{L}^+_i$, the ith column of $\mathbf{L}^+$), and the first inequality follows since E contains both (u,v) and (v,u) for all edges $(u,v) \in Q_{ij}$. Combining Eqs. (37–39), we thus get that

$$L^+_{ii} \geq \frac{w_{\min}}{r} \frac{(L^+_{ii})^2}{d_i} ,$$

which gives

$$L^+_{ii} \leq \frac{r d_i}{w_{\min}} .$$

The result follows since $r \leq \rho$ and $d_i \leq d$.

□

B Proof of Theorem 7

We shall need the following concentration inequality due to El-Yaniv and Pechyony (2006), stated here using our notation from Section 5.2:

Theorem 9 (El-Yaniv and Pechyony (2006)) *Let $V = \{1,\ldots,n\}$, and let ϕ be a real-valued function defined on size-m subsets of V such that the following is satisfied: there exists a constant $c > 0$ such that for all subsets $S = \{i_1,\ldots,i_m\} \subset V$ and all $i_k \in S$, $i'_k \in V \setminus S$,*

$$\left| \phi(S) - \phi(S^{(i_k,i'_k)}) \right| \ \leq \ c .$$

Then for any $\varepsilon > 0$,

$$\mathbf{P}_{S\sim\mathscr{T}_m} \Big(\phi(S) - \mathbf{E}_{S\sim\mathscr{T}_m}[\phi(S)] \ \geq \ \varepsilon \Big) \ \leq \ \exp\left(\frac{-\varepsilon^2}{2c^2 \left(\sum_{r=n-m+1}^{n} \left(\frac{(n-m)^2}{r^2} \right) \right)} \right) .$$

We shall also need the following lemma:

Lemma 1 *Let $\mathscr{A}$ be a transductive ranking algorithm whose output on a training set $S \subset V$ we denote by f_S. Let ℓ be a ranking loss, and let $\beta : \mathbb{N} \to \mathbb{R}$ be such that A has uniform transductive loss stability β with respect to ℓ. Then*

$$\mathbf{E}_{S\sim\mathscr{T}_m} \left[R^\ell_V(f_S) - R^\ell_S(f_S) \right] \ \leq \ \frac{4(n-m)}{n} \beta(m) .$$

Proof We have,

$$\begin{aligned}
\mathbf{E}_{S\sim\mathscr{T}_m}\left[R_V^\ell(f_S)\right] &= \frac{1}{\binom{n}{m}}\sum_{S\subset V,|S|=m} R_V(f_S) \\
&= \frac{1}{\binom{n}{m}}\sum_{S\subset V,|S|=m}\frac{1}{\binom{n}{2}}\sum_{1\le i<j\le n}\ell(f_S,i,j) \\
&= \frac{1}{\binom{n}{m}}\frac{1}{\binom{n}{2}}\sum_{S\subset V,|S|=m}\Big[I_1(S)+I_2(S)+I_3(S)+I_4(S)\Big], \qquad (40)
\end{aligned}$$

where

$$I_1(S) = \sum_{\substack{i<j\\ i,j\in S}}\ell(f_S,i,j), \qquad (41)$$

$$\begin{aligned}
I_2(S) &= \sum_{\substack{i<j\\ i\in S, j\notin S}}\ell(f_S,i,j) \\
&\le \sum_{\substack{i<j\\ i\in S, j\notin S}}\frac{1}{m-1}\sum_{\substack{k\in S\\ k\neq i}}\left[\ell(f_{S^{(k,j)}},i,j)+\beta(m)\right], \qquad (42)
\end{aligned}$$

(where the inequality follows from β-stability), and similarly,

$$\begin{aligned}
I_3(S) &= \sum_{\substack{i<j\\ i\notin S, j\in S}}\ell(f_S,i,j) \\
&\le \sum_{\substack{i<j\\ i\in S, j\notin S}}\frac{1}{m-1}\sum_{\substack{k\in S\\ k\neq j}}\left[\ell(f_{S^{(k,i)}},i,j)+\beta(m)\right], \qquad (43)
\end{aligned}$$

$$\begin{aligned}
I_4(S) &= \sum_{\substack{i<j\\ i,j\notin S}}\ell(f_S,i,j) \\
&\le \sum_{\substack{i<j\\ i,j\notin S}}\frac{1}{m(m-1)}\sum_{\substack{k,l\in S\\ k\neq l}}\left[\ell(f_{S^{(k,i),(l,j)}},i,j)+2\beta(m)\right]. \qquad (44)
\end{aligned}$$

Note that in each of the above upper bounds on $I_1(S)$, $I_2(S)$, $I_3(S)$ and $I_4(S)$, the loss terms in the summations are all of the form $\ell(f_{S'},i,j)$ with $i,j\in S'$ (and $i<j$). Adding these over all $S\subset V$ with $|S|=m$, we find that for each S and for each pair $i,j\in S$ (and $i<j$), the loss term $\ell(f_S,i,j)$ occurs multiple times; collecting all

the coefficients for each of these terms and substituting in Eq. (40), we get:

$$\begin{aligned}
\mathbf{E}_{S\sim\mathscr{T}_m}\left[R_V^\ell(f_S)\right] &\leq \frac{1}{\binom{n}{m}}\frac{1}{\binom{n}{2}}\left[\sum_{S\subset V,|S|=m}\sum_{\substack{i<j\\ i,j\in S}}\ell(f_S,i,j)\left(1+\frac{2(n-m)}{m-1}+\frac{(n-m)(n-m-1)}{m(m-1)}\right)\right.\\
&\qquad\left.+\binom{n}{m}\beta(m)\Big(2m(n-m)+2(n-m)(n-m-1)\Big)\right]\\
&= \left[\frac{1}{\binom{n}{m}}\sum_{S\subset V,|S|=m}\frac{1}{\binom{m}{2}}\sum_{\substack{i<j\\ i,j\in S}}\ell(f_S,i,j)\right]+\frac{4(n-m)}{n}\beta(m)\\
&= \left[\frac{1}{\binom{n}{m}}\sum_{S\subset V,|S|=m}R_S^\ell(f_S)\right]+\frac{4(n-m)}{n}\beta(m)\\
&= \mathbf{E}_{S\sim\mathscr{T}_m}\left[R_S^\ell(f_S)\right]+\frac{4(n-m)}{n}\beta(m)\,. \qquad (45)
\end{aligned}$$

The result follows.

□

We are now ready to prove Theorem 7. The proof is similar to the proof of the corresponding result of (Agarwal and Niyogi, 2008) (stated as Theorem 2 in Section 5.1), which is derived under the usual model of independent sampling; the main difference in our proof is the use of Theorem 9 in place of McDiarmid's inequality, and the use of Lemma 1 in place of an analogous result of (Agarwal and Niyogi, 2008).

Proof [of Theorem 7] Define a real-valued function ϕ on subsets of V of size m as follows:

$$\phi(S) \;=\; R_V^\ell(f_S)-R_S^\ell(f_S)\,.$$

Then following the same steps as in the proof of (Agarwal and Niyogi, 2008)[Theorem 8], it is easy to show that for any $S=\{i_1,\ldots,i_m\}\subset V$ and any $i_k\in S$, $i'_k\in V\setminus S$,

$$\left|\phi(S)-\phi(S^{(i_k,i'_k)})\right| \;\leq\; 2\left(\beta(m)+\frac{B}{m}\right).$$

Therefore, applying Theorem 9, we get for any $\varepsilon>0$,

$$\mathbf{P}_{S\sim\mathscr{T}_m}\Big(\phi(S)-\mathbf{E}_{S\sim\mathscr{T}_m}[\phi(S)] \;\geq\; \varepsilon\Big) \;\leq\; \exp\left(\frac{-\varepsilon^2}{8\left(\beta(m)+\frac{B}{m}\right)^2\left(\sum_{r=n-m+1}^{n}(\frac{(n-m)^2}{r^2})\right)}\right).$$

Setting the right hand side equal to δ and solving for ε gives that with probability at least $1-\delta$ over the draw of S according to $\mathscr{T}_m$,

$$\phi(S) \;<\; \mathbf{E}_{S\sim\mathscr{T}_m}[\phi(S)] \;+\; 2\left(\beta(m)+\frac{B}{m}\right)\sqrt{2\left(\sum_{r=n-m+1}^{n}\frac{(n-m)^2}{r^2}\right)\ln\left(\frac{1}{\delta}\right)}\,.$$

Now, using the identity

$$\frac{1}{r^2} \;\leq\; \int_{t=r-\frac{1}{2}}^{r+\frac{1}{2}}\frac{1}{t^2}\,dt$$

for all $r \in \mathbb{N}$ (this identity was also used in (Cortes et al, 2008) for a similar purpose), we get

$$\begin{aligned}
\sum_{r=n-m+1}^{n} \frac{(n-m)^2}{r^2} &\leq (n-m)^2 \int_{t=n-m+\frac{1}{2}}^{n+\frac{1}{2}} \frac{1}{t^2}\,dt \\
&= (n-m)^2 \frac{m}{(n-m+\frac{1}{2})(n+\frac{1}{2})} \\
&\leq (n-m)^2 \frac{m}{(n-m)n} \\
&= \frac{m(n-m)}{n}.
\end{aligned}$$

Substituting above, this gives that with probability at least $1-\delta$ over the draw of S according to $\mathscr{T}_m$,

$$\phi(S) \;<\; \mathbf{E}_{S\sim\mathscr{T}_m}\left[\phi(S)\right] \;+\; 2\left(m\beta(m)+B\right)\sqrt{\frac{2(n-m)}{mn}\ln\left(\frac{1}{\delta}\right)}.$$

The result then follows by Lemma 1.

□

References

Agarwal S (2006) Ranking on graph data. In: Proceedings of the 23rd International Conference on Machine Learning

Agarwal S, Niyogi P (2008) Stability and generalization of ranking algorithms. Journal of Machine Learning Research To appear

Agarwal S, Graepel T, Herbrich R, Har-Peled S, Roth D (2005) Generalization bounds for the area under the ROC curve. Journal of Machine Learning Research 6:393–425

Belkin M, Niyogi P (2004) Semi-supervised learning on Riemannian manifolds. Machine Learning 56:209–239

Belkin M, Matveeva I, Niyogi P (2004) Regularization and semi-supervised learning on large graphs. In: Proceedings of the 17th Annual Conference on Learning Theory

Blum A, Lafferty J, Rwebangira MR, Reddy R (2004) Semi-supervised learning using randomized mincuts. In: Proceedings of the 21st International Conference on Machine Learning

Bousquet O, Elisseeff A (2002) Stability and generalization. Journal of Machine Learning Research 2:499–526

Boyd S, Vandenberghe L (2004) Convex Optimization. Cambridge University Press

Burges C, Shaked T, Renshaw E, Lazier A, Deeds M, Hamilton N, Hullender G (2005) Learning to rank using gradient descent. In: Proceedings of the 22nd International Conference on Machine Learning

Burges CJC (1998) A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery 2(2):121–167

Chung FRK (1997) Spectral Graph Theory. American Mathematical Society

Chung FRK (2005) Laplacians and the Cheeger inequality for directed graphs. Annals of Combinatorics 9:1–19

Clemencon S, Lugosi G, Vayatis N (2005) Ranking and scoring using empirical risk minimization. In: Proceedings of the 18th Annual Conference on Learning Theory

Cohen WW, Schapire RE, Singer Y (1999) Learning to order things. Journal of Artificial Intelligence Research 10:243–270

Cortes C, Mohri M, Rastogi A (2007) Magnitude-preserving ranking algorithms. In: Proceedings of 24th International Conference on Machine Learning

Cortes C, Mohri M, Pechyony D, Rastogi A (2008) Stability of transductive regression algorithms. In: Proceedings of 25th International Conference on Machine Learning

Cossock D, Zhang T (2006) Subset ranking using regression. In: Proceedings of the 19th Annual Conference on Learning Theory

Crammer K, Singer Y (2002) Pranking with ranking. In: Advances in Neural Information Processing Systems 14

El-Yaniv R, Pechyony D (2006) Stable transductive learning. In: Proceedings of the 19th Annual Conference on Learning Theory

Freund Y, Iyer R, Schapire RE, Singer Y (2003) An efficient boosting algorithm for combining preferences. Journal of Machine Learning Research 4:933–969

Gärtner T, Flach PA, Wrobel S (2003) On graph kernels: Hardness results and efficient alternatives. In: Proceedings of the 16th Annual Conference on Learning Theory

Hanneke S (2006) An analysis of graph cut size for transductive learning. In: Proceedings of 23rd International Conference on Machine Learning

Herbrich R, Graepel T, Obermayer K (2000) Large margin rank boundaries for ordinal regression. Advances in Large Margin Classifiers pp 115–132

Herbster M, Pontil M, Wainer L (2005) Online learning over graphs. In: Proceedings of 22nd International Conference on Machine Learning

Joachims T (1999) Making large-scale SVM learning practical. Advances in Kernel Methods - Support Vector Learning

Joachims T (2002) Optimizing search engines using clickthrough data. In: Proceedings of the ACM Conference on Knowledge Discovery and Data Mining

Johnson R, Zhang T (2007) On the effectiveness of Laplacian normalization for graph semi-supervised learning. Journal of Machine Learning Research 8:1489–1517

Johnson R, Zhang T (2008) Graph-based semi-supervised learning and spectral kernel design. IEEE Transactions on Information Theory 54(1):275–288

Kondor RI, Lafferty J (2002) Diffusion kernels on graphs and other discrete structures. In: Proceedings of the 19th International Conference on Machine Learning

McCallum AK (1996) Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering, http://www.cs.cmu.edu/ mccallum/bow

McDiarmid C (1989) On the method of bounded differences. In: Surveys in Combinatorics 1989, Cambridge University Press, pp 148–188

Platt J (1999) Fast training of support vector machines using sequential minimal optimization. Advances in Kernel Methods - Support Vector Learning

Roweis ST, Saul LK (2000) Nonlinear dimensionality reduction by locally linear embedding. Science 290(5500):2323–2326

Rudin C, Cortes C, Mohri M, Schapire RE (2005) Margin-based ranking meets boosting in the middle. In: Proceedings of the 18th Annual Conference on Learning Theory

Smola AJ, Kondor R (2003) Kernels and regularization on graphs. In: Proceedings of the 16th Annual Conference on Learning Theory

Strang G (1988) Linear Algebra and Its Applications, 3rd edn. Brooks Cole

Tenenbaum J, de Silva V, Langford JC (2000) A global geometric framework for nonlinear dimensionality reduction. Science 290(5500):2319–2323

Zhou D, Schölkopf B (2004) A regularization framework for learning from graph data. In: ICML Workshop on Statistical Relational Learning

Zhou D, Weston J, Gretton A, Bousquet O, Schölkopf B (2004) Ranking on data manifolds. In: Advances in Neural Information Processing Systems 16

Zhou D, Huang J, Schölkopf B (2005) Learning from labeled and unlabeled data on a directed graph. In: Proceedings of the 22nd International Conference on Machine Learning