An Analysis of Graph Cut Size for Transductive Learning

Steve Hanneke SHANNEKE@CS.CMU.EDU
Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA 15213 USA

Abstract

I consider the setting of transductive learning of vertex labels in graphs, in which a graph with n vertices is sampled according to some unknown distribution; there is a true labeling of the vertices such that each vertex is assigned to exactly one of k classes, but the labels of only some (random) subset of the vertices are revealed to the learner. The task is then to find a labeling of the remaining (unlabeled) vertices that agrees as much as possible with the true labeling. Several existing algorithms are based on the assumption that adjacent vertices are usually labeled the same. In order to better understand algorithms based on this assumption, I derive data-dependent bounds on the fraction of mislabeled vertices, based on the number (or total weight) of edges between vertices differing in predicted label (i.e., the size of the *cut*).

1. Introduction

Graphs are the primary object of study for many machine learning application areas. For example, in social network analysis, one is often interested in finding patterns in a graph representing social interactions among individuals in a population, and basing further predictions on those patterns. Similar examples exist for analyzing citation networks, computer networks, biological networks, and the world wide web. In particular, for certain applications one is interested in classifying the vertices of a graph, each into one of k classes; one often has access to the correct labels for some subset of the vertices, and is interested only in producing accurate predictions for the remaining (unlabeled) vertices. Such applications are substantially different from traditional machine learning applications, since they cannot easily be viewed in terms of repeated independent sampling from a distribution. Instead, they are more readily modeled in terms of sampling the entire graph from a distribution over the space of all n-vertex graphs, and then allowing an oracle to label a random subset of the vertices according to some unknown pattern.

Appearing in *Proceedings of the 23rd International Conference on Machine Learning*, Pittsburgh, PA, 2006. Copyright 2006 by the author(s)/owner(s).

Several existing algorithms are applicable to this setting, many of which are based on the assumption that adjacent (or heavily linked) vertices tend to be labeled the same. For the case of two classes, (Blum & Chawla, 2001) propose an algorithm which labels a graph so as to respect the known labels, while at the same time minimizing the number (or total weight) of edges between vertices having different predicted labels. Because this problem is solved by a reduction to the minimum weight s-t cut graph partitioning problem, it is referred to as the *mincut* learning algorithm, and the number (or weight) of edges between differently labeled vertices is referred to as the *cut size* of the labeling. The algorithm is appealing for its simplicity and solid foundation in graph partitioning. However, existing analyses of using cut size to bias learning have been largely based on assumptions of how the graph is constructed. The analysis by (Blum & Chawla, 2001) assumes that the algorithm is initially presented with examples drawn independently from a distribution, and then *constructs* the graph in a prescribed way. More recent variants of this approach (Joachims, 2003; Blum et al., 2004) have similar motivating assumptions, but by way of comments and roughly sketched arguments, indicate the desire for a more general analysis applicable to any graph. In particular, (Joachims, 2003) poses an open question essentially asking whether it is possible to derive data-dependent bounds on the fraction of mistaken predictions, based on the cut size of the labeling. In this paper, I constructively answer this question in the affirmative.

In Section 2, I give a brief overview of the setting and introduce the notation used throughout. This is followed by a formal discussion of using cut size in a learning bias in Section 3, which also includes a discussion of algorithms for the multiclass problem. Section 4, which contains the main contributions of this paper, provides data-dependent bounds on the fraction of mistaken predictions based on the cut size of the labeling. I give tight implicit bounds as well as somewhat looser explicit bounds, and discuss interesting special cases.

2. Transductive Vertex Classification

To introduce notation, I will outline the fundamental concepts for this setting. The input to the learning algorithm includes a graph $G = (V, E)$, sampled from an unknown distribution over graphs having n vertices. For simplicity, assume the edges are undirected, and unless otherwise noted, are not weighted; where appropriate I will comment on natural extensions to weighted graphs. For a given graph $G = (V, E)$, a hypothesis space $\mathcal{H}_V$ is the set of distinct vertex labelings from which the learner can select. Throughout this analysis, I will only consider $\mathcal{H}_V$ to be the completely unrestricted hypothesis space, containing all k^n possible labelings of the graph with k types of labels $\{1, 2, \ldots, k\}$. To focus only on interesting cases, I will assume $k \geq 2$. Also, for notational simplicity when V is clear from the context, I will simply write $\mathcal{H}$. There is a target labeling $f \in \mathcal{H}$ for the vertices of G, so that each vertex $v \in V$ is assigned one of k possible values $\{1, 2, \ldots, k\}$; denote by $f(v)$ the specific target label for v. In addition to G, the learning algorithm is given the target labels for some subset of n_ℓ vertices selected uniformly at random[1] from V. Let us refer to these n_ℓ vertices as *training* vertices and the remaining $n - n_\ell$ (unlabeled) vertices as *test* vertices. To make this more formal, I will refer to any $\lambda \in \{0, 1\}^n$ as a *label mask*, and it will be convenient to define $\Lambda_{n_\ell} = \{\lambda \mid \|\lambda\|^2 = n_\ell,\ \lambda \in \{0, 1\}^n\}$. Thus, for a graph $G = (V, E)$ on $V = \{v_1, v_2, \ldots, v_n\}$ and a label mask $\lambda \in \Lambda_{n_\ell}$, the n_ℓ *training* vertices are defined as those v_i such that $\lambda_i = 1$.

Definition 1. *Define the* empirical error *and* prediction error *of a hypothesis h with respect to a target labeling f, vertex set $V = \{v_1, v_2, \ldots, v_n\}$, and label mask $\lambda \in \Lambda_{n_\ell}$ by (respectively)*

$$\hat{R}(h, f, V, \lambda) = \frac{1}{n_\ell} \sum_{i=1}^{n} \lambda_i I[f(v_i) \neq h(v_i)]$$

$$R(h, f, V, \lambda) = \frac{1}{n - n_\ell} \sum_{i=1}^{n} (1 - \lambda_i) I[f(v_i) \neq h(v_i)]$$

[1]This is arguably a strong assumption, and in practice there may be more natural ways to select the training vertices for a particular application. This could perhaps be viewed as the transductive analogue of the commonly made i.i.d. assumption in inductive learning.

That is, the *empirical* error is the fraction of *training* vertices the labeling is mistaken on, while the *prediction* error is the fraction of *test* vertices the labeling is mistaken on. To simplify notation, when f, λ, and V are clear from the context, I will abbreviate these by $\hat{R}(h)$ and $R(h)$, respectively.

3. A Learning Bias Toward Small Cuts

The objective is to pick a labeling that minimizes the prediction error. However, since this quantity generally cannot be directly computed from the algorithm's input, we need to bias the algorithm somehow. In many applications, it is natural to assume that adjacent vertices will usually tend to have the same label. This assumption leads naturally to a learning bias based on cut size.

Definition 2. *For any labeling $h \in \mathcal{H}$ of a graph $G = (V, E)$, define the* cut size *of the labeling, denoted $c(h)$, as*

$$c(h) = \sum_{\{u,v\} \in E} I\left[h(u) \neq h(v)\right]$$

For weighted edges, each term in the summation is also multiplied by the edge weight. The assumption that adjacent vertices tend to have the same labels leads to a bias toward small cut sizes. One way to implement this bias in an algorithm is to search for a labeling having smallest cut size while still respecting the target labels of training vertices. For the case of $k = 2$ classes, (Blum & Chawla, 2001) give an algorithm for doing exactly that, based on an efficient reduction to a minimum s-t cut graph partitioning problem, which can be solved exactly in polynomial time (see e.g., (Cormen et al., 2001)), and for which a variety of more efficient approximate solutions are known (see e.g., (Benczúr & Karger, 1996; Karger, 1999)).

The so-called multiterminal minimum cut problem is a natural generalization of the minimum s-t cut partitioning problem. Given a weighted graph $G = (V, E)$ and a set of "terminal" vertices $\{s_1, s_2, \ldots, s_k\} \subseteq V$, the problem is to find a set of edges $\mathcal{C} \subseteq E$ having minimum total weight such that any path between any terminal s_i and any other terminal s_j must include at least one edge from $\mathcal{C}$. Put another way, the task is to find a minimum weight set of edges, the removal of which disconnects all of the terminals. A multiclass mincut algorithm can be described by a reduction to the multiterminal minimum cut problem as follows.[2]

[2]As with the original mincut learning algorithm, this algorithm can be applied to both weighted and unweighted graphs; as usual, in the unweighted case, simply take the weights of edges from G to be 1.

```
1. Add $k$ new vertices $\{s_1, s_2, \dots, s_k\}$ to the graph
2. $\forall i \in \{1, 2, ..., k\}$, add an edge of $\infty$ weight between
   $s_i$ and any training vertex $v$ for which $f(v) = i$
3. Find a multiterminal minimum cut $\mathcal{C}$ of the
   constructed graph, taking $\{s_1, \dots, s_k\}$ as terminals
4. For each $i \in \{1, 2, ..., k\}$, label with $i$ all test vertices
   from which $\exists$ a path to $s_i$ not including edges in $\mathcal{C}$
5. Any test vertices not yet labeled can be labeled
   arbitrarily (e.g., randomly)
```

Unfortunately, the multiterminal minimum cut problem is MAX-SNP-Hard (Dahlhaus et al., 1994). However, this is a well-studied problem in computer science, and as such there are a variety of clever approximation algorithms for the problem. In particular, (Călinescu et al., 2000) give a linear programming relaxation resulting in an efficient algorithm guaranteed to find a cut size no larger than $(\frac{3}{2} - \frac{1}{k})$ times the optimal size. This result has since been slightly improved for the general case (Karger et al., 2004), and has been reduced to within a factor of $\frac{12}{11}$ of the optimal cut size for the special case of $k = 3$ (Cheung et al., 2005; Karger et al., 2004). Note that these are worst-case guarantees, and these algorithms often produce optimal solutions in practice.

4. Cut Bounds

Let us now turn to the subject of deriving data-dependent bounds on the prediction error based on cut size. For simplicity, assume $n \geq k$, and that the graph initially has fewer than k components[3]. I begin with some definitions.

Definition 3. *For a graph $G = (V, E)$, let $c_k(G)$ denote the minimum number of edges whose removal separates G into at least k nonempty connected components. Put another way, $c_k(G)$ is the smallest possible multiterminal minimum cut size over all choices of k distinct terminal vertices from V. I refer to $c_k(G)$ as the* minimum k-cut size *of G. For fixed k, it can be computed in polynomial time (see e.g., (Goldschmidt & Hochbaum, 1994)).*

As with cut size, this can be generalized for weighted graphs by the sum of weights of edges in a set having smallest total weight whose removal separates G into at least k nonempty connected components.

[3]Note that if the graph has k or more components, we can still obtain a bound by a weighted average of the bounds for its components (each holding with probability $1 - \delta$), which holds with probability $1 - N\delta$ for N components (due to the union bound).

Definition 4. *Let*

$$F_T(m) = \sum_{t=0}^{m} \frac{\binom{T}{t}\binom{n-T}{n_\ell - t}}{\binom{n}{n_\ell}}$$

denote the cumulative distribution function of the hypergeometric distribution, with nonnegative integer population sizes T and $n - T$, and sample size n_ℓ; for notational simplicity, I have made the dependence of $F_T(m)$ on n and n_ℓ implicit. Also, define $\binom{x}{y} = 0$ when $x < y$. Let

$$R_{max}(r, \delta) = \max\left\{ \frac{T - rn_\ell}{n - n_\ell} \,\middle|\, F_T(rn_\ell) \geq \delta, T \in \mathbb{Z} \right\}$$

The following is a version of the PAC-MDL theorem by (Blum & Langford, 2003). For completeness, I include its proof in Appendix A.

Lemma 1. *(PAC-MDL) For any n-vertex graph G and hypothesis space H, if $q : H \to (0, 1)$ is a function such that $\sum_{h \in H} q(h) \leq 1$, then for any target labeling f, $\delta \in (0, 1)$, and label mask λ selected uniformly at random from Λ_{n_ℓ}, with probability at least $1 - \delta$, every hypothesis $h \in H$ will have*

$$R(h) \leq R_{max}\left(\hat{R}(h), q(h)\delta\right).$$

Definition 5. *The following definitions will be useful. For an n-vertex graph $G = (V, E)$, and hypothesis $h \in \mathcal{H}$, let*

$$\rho(h) = \frac{c(h)}{c_k(G)},$$

$$\mathfrak{s}(h) = \max\{\lfloor 2k\rho(h) \rfloor, k - 1\},$$

and

$$\mathcal{B}(h) = \begin{cases} k^{\mathfrak{s}(h)} \binom{n}{2(k-1)\rho(h)} \binom{\mathfrak{s}(h)}{2(k-1)\rho(h)}^{-1} & \text{if } n \geq \lfloor 2k\rho(h) \rfloor \\ k^n & \text{otherwise} \end{cases}$$

where

$$\binom{x}{y} = \frac{\Gamma(x+1)}{\Gamma(y+1)\Gamma(x-y+1)}$$

is a generalized form of binomial coefficient, and $\Gamma(\cdot)$ is the well-known gamma function. Note that $\rho(h)$ essentially represents the relative cut size of h with respect to the smallest possible k-cut size in G. I will find it convenient to overload this notation, so that for an integer c, $\rho(c) = \frac{c}{c_k(G)}$, $\mathfrak{s}(c) = \max\{\lfloor 2k\rho(c) \rfloor, k-1\}$, and similarly for $\mathcal{B}(c)$. Additionally, for $\delta \in (0, 1)$ define

$$\hat{\mathfrak{c}}(\delta) = \min\{c \mid \tbinom{n}{n_\ell}\delta \leq (c+1)\mathcal{B}(c+1), c \in \{0,1,\dots,|E|\}\} \cup \{|E|\}$$

With these definitions in mind, we have the following novel theorem.

Theorem 1. *(Implicit) For any unweighted n-vertex graph*[4] $G = (V, E)$*, target labeling* f*,* $\delta \in (0, 1)$*, and label mask* λ *selected uniformly at random from* Λ_{n_ℓ}*, with probability at least* $1 - \delta$*, every* $h \in \mathcal{H}$ *will satisfy*

$$R(h) \leq R_{max}\left(\hat{R}(h), \frac{\delta \mathcal{B}(h)^{-1}}{\hat{\mathfrak{c}}(\delta) + 1}\right)$$

The following lemma will be useful in proving the theorem. Its proof is included in Appendix B.

Lemma 2. *Let* $G = (V, E)$ *be any unweighted graph on n vertices. For any nonnegative integer c, there are no more than*

$$\mathcal{B}(c) = \begin{cases} k^{\mathfrak{s}(c)} \binom{n}{2(k-1)\rho(c)} \binom{\mathfrak{s}(c)}{2(k-1)\rho(c)}^{-1} & \textit{if } \mathfrak{s}(c) < n \\ k^n & \textit{otherwise} \end{cases}$$

labelings $h \in \mathcal{H}$ *such that* $c(h) \leq c$*.*

Proof of Theorem 1. Let $S_c = \{h | c(h) \leq c,\ h \in \mathcal{H}\}$. We have that

$$\sum_{h \in S_{\hat{\mathfrak{c}}(\delta)}} \frac{\mathcal{B}(h)^{-1}}{\hat{\mathfrak{c}}(\delta) + 1} \leq \frac{1}{\hat{\mathfrak{c}}(\delta) + 1} \sum_{c=0}^{\hat{\mathfrak{c}}(\delta)} \frac{|S_c|}{\mathcal{B}(c)} \leq 1$$

where the second inequality follows from Lemma 2. Since $\forall h \in \mathcal{H}, \mathcal{B}(h) > 0$, Lemma 1 implies

$$\mathcal{P}r\left\{\forall h \in S_{\hat{\mathfrak{c}}(\delta)}, R(h) \leq R_{max}\left(\hat{R}(h), \frac{\delta \mathcal{B}(h)^{-1}}{\hat{\mathfrak{c}}(\delta) + 1}\right)\right\} \geq 1 - \delta$$

Finally, noting that $\forall h \in \mathcal{H}$,

$$c(h) > \hat{\mathfrak{c}}(\delta) \Rightarrow R_{max}\left(\hat{R}(h), \frac{\delta \mathcal{B}(h)^{-1}}{\hat{\mathfrak{c}}(\delta) + 1}\right) = 1 \geq R(h)$$

completes the proof for all $h \in \mathcal{H}$. □

While Theorem 1 provides a fairly tight bound, it is implicitly dependent on a variety of factors in ways that can make it difficult to build intuition. As such, in order to make the relation between cut size and error more explicit, I present the following relaxation of Theorem 1, building on a result of (Derbeko et al., 2004). As we shall see, though significantly looser, this bound approximates the basic dependence of Theorem 1 on cut size.

[4]This result actually applies to any unweighted multigraph. Also, see below for a discussion of extensions to weighted graphs.

Theorem 2. *(Explicit) Let* $n_u = n - n_\ell$*. For an n-vertex unweighted graph* $G = (V, E)$*, target function* f*,* $\delta \in (0, 1)$*, and label mask* λ *selected uniformly at random from* Λ_{n_ℓ}*, with probability at least* $1 - \delta$*, every* $h \in \mathcal{H}$ *such that* $2k\rho(h) < n$ *satisfies*[5]

$$R(h) \leq \hat{R}(h) + \sqrt{\left(\frac{n}{n_u}\right)\left(\frac{n_u + 1}{n_u}\right)\left(\frac{\bar{\mathfrak{s}}(h) \ln\left(\frac{kne}{\bar{\mathfrak{s}}(h)+1}\right) + \ln \frac{(|E|+1)}{\delta}}{2n_\ell}\right)}$$

where $\bar{\mathfrak{s}}(h) = \max\{2(k-1)\rho(h), k-1\}$*.*

Proof Sketch. This proof follows analogously to the proof of Theorem 1, except using the following relaxation of Lemma 1, developed by (Derbeko et al., 2004) using Serfling's bound.

Lemma 3. *(Adapted from (Derbeko et al., 2004)) For any n-vertex graph G and hypothesis space H, if* $q : H \rightarrow (0, 1)$ *is a function such that* $\sum_{h \in H} q(h) \leq 1$*, then for any target labeling* f*,* $\delta \in (0, 1)$*, and label mask* λ *selected uniformly at random from* Λ_{n_ℓ}*, with probability at least* $1 - \delta$*, every hypothesis* $h \in H$ *will have*

$$R(h) \leq \hat{R}(h) + \sqrt{\left(\frac{n}{n_u}\right)\left(\frac{n_u + 1}{n_u}\right)\left(\frac{\ln \frac{1}{q(h)} + \ln \frac{1}{\delta}}{2n_\ell}\right)}$$

For detailed proof of Lemma 3, see (Derbeko et al., 2004).

Additionally, the proof of Theorem 2 relies on the following inequality, which applies when $2k\rho(h) < n$.

$$\mathcal{B}(h) \leq \left(\frac{kne}{\bar{\mathfrak{s}}(h) + 1}\right)^{\bar{\mathfrak{s}}(h)}$$

This can be derived as follows.

$$\mathcal{B}(h) \leq \left(\frac{k^{\bar{\mathfrak{s}}(h)}}{\Gamma(\bar{\mathfrak{s}}(h)+1)}\right)\left(\frac{\Gamma(n+1)\Gamma(\bar{\mathfrak{s}}(h) - 2(k-1)\rho(h)+1)}{\Gamma(n - 2(k-1)\rho(h) + 1)}\right)$$
$$\leq \left(\frac{ke}{\bar{\mathfrak{s}}(h) + 1}\right)^{\bar{\mathfrak{s}}(h)} \frac{n^{2(k-1)\rho(h)}}{(\bar{\mathfrak{s}}(h) - 2(k-1)\rho(h) + 1)^{(2(k-1)\rho(h) - \bar{\mathfrak{s}}(h))}}$$
$$\leq \left(\frac{kne}{\bar{\mathfrak{s}}(h) + 1}\right)^{\bar{\mathfrak{s}}(h)}$$

The first and second inequalities are based on properties that are well known for (integer) factorials, and with a bit of effort one can show they also hold for the general (continuous) form. □

[5]Note that Theorem 1 gives the trivial bound of $R(h) \leq 1$ when $2k\rho(h) \geq n$, so we can define this relaxation to be 1 for that case as well.

Theorems 1 and 2, as well as the Lemmas on which they depend, actually apply to any unweighted multigraph[6]; that is, the same proofs can be used without modification for multigraphs. We can use this fact to generalize them for any weighted graph having nonnegative rational-valued edge weights as follows.

Corollary 1. *Denote by $w(u,v)$ the weight of edge $\{u,v\} \in E$, and define*

$$W = \textstyle\sum_{\{u,v\}\in E} w(u,v),$$
$$L = LCD(\{w(u,v) | \{u,v\} \in E\}),$$
$$D = GCD(\{w(u,v)L | \{u,v\} \in E\}),$$

and

$$Q = \frac{L}{D},$$

where LCD denotes the least common denominator and GCD denotes the greatest common divisor.[7] *If all edge weights are nonnegative rational numbers, and $c_k(G) > 0$, then under the conditions of Theorem 1, every $h \in \mathcal{H}$ will satisfy*

$$R(h) \leq R_{max}\left(\hat{R}(h), \frac{\delta \mathcal{B}(h)^{-1}}{\tilde{\mathfrak{c}}(\delta) + 1}\right)$$

where

$$\tilde{\mathfrak{c}}(\delta) = \min\{\tilde{c} | \tbinom{n}{n_\ell}\delta \leq (\tilde{c}+1)\mathcal{B}(\tfrac{\tilde{c}+1}{Q}), \tilde{c} \in \{0,1,\dots,QW\}\} \cup \{QW\}$$

Proof. Consider the multigraph $\tilde{G}$ on V formed by replacing every edge $\{u,v\} \in E$ with $Qw(u,v)$ unweighted edges. This gives a multigraph having QW edges. For any $h \in \mathcal{H}$, let $\tilde{c}(h)$ and $\tilde{\mathcal{B}}(h)$ denote the cut size and $\mathcal{B}$ value evaluated in $\tilde{G}$. Note that $\tilde{c}(h) = Qc(h)$ and $\tilde{\mathcal{B}}(h) = \mathcal{B}(h)$, and furthermore (overloading notation as before), $\tilde{\mathcal{B}}(\tilde{c}) = \mathcal{B}(\frac{\tilde{c}}{Q})$. Applying Theorem 1 for this unweighted multigraph gives the desired result. □

In this same way, a result similar to Corollary 1 can be formulated for extending Theorem 2 to nonnegative rational-valued edge weights as well. The details are left as an exercise. Note that this bound is invariant to scaling of the weights.

As a special case it is interesting to examine the case of $k = 2$ classes. One can show[8] that for $k = 2$,

$$|S_c| \leq 2n^{2\rho(c)}.$$

Corollary 2. *Using the above fact, one can show for any connected n-vertex graph $G = (V,E)$ having nonnegative rational weights, with W, Q, and n_u defined as above, for any target labeling f, $\delta \in (0,1)$, and label mask λ selected uniformly at random from Λ_{n_ℓ}, that with probability at least $1-\delta$, every $h \in \mathcal{H}$ satisfies*

$$R(h) \leq \hat{R}(h) + \sqrt{\left(\frac{n}{n_u}\right)\left(\frac{n_u+1}{n_u}\right)\left(\frac{\frac{c(h)}{c_2(G)}\ln(n) + \frac{1}{2}\ln\frac{2(QW+1)}{\delta}}{n_\ell}\right)}$$

This essentially relates the error to the cut size in a very basic way, such that relative to the smallest possible k-cut size, the smaller the cut size of the labeling is, the better our guarantee on the error becomes. Note that a related result was discussed by (Blum et al., 2004) for sample complexity, derived by other means building on results of (Kleinberg et al., 2004).

For comparison of the implicit bound of Theorem 1 to the explicit bound of Theorem 2, I give plots of these bounds versus cut size for unweighted graphs in Figure 1: $k = 2$ in the left plot, and $k = 4$ in the right plot. In both cases, $\hat{R}(h) = 0$, $n = 10,000$, $|E| = 1,000,000$, $c_k(G) = 10(k-1)$, $n_\ell = 500$, and $\delta = 0.01$. For the case of $k = 2$, I also plot the bound from Corollary 2. Note that the basic shapes of these bounds are approximately the same, indicating that the basic dependence on cut size implicit in Theorem 1 is successfully captured in a more explicit form in Theorem 2 and Corollary 2 (though to a lesser extent in the latter). Also note that for both plots, the PAC-MDL bound of Lemma 1 with uniform prior $q(\cdot) = \frac{1}{|\mathcal{H}|}$ over hypothesis space $\mathcal{H}$ is constant at 1, indicating these bounds are significantly tighter.

5. Discussion and Open Problems

For the quantities it uses, the bound given in Theorem 1 is fairly tight. The only sources of slackness come from the bound in Lemma 2 and the use of the union bound. However, as (Karger, 1996) notes, one can show that (for example) for $k = 2$, there exist graphs G for which a lower bound on $|S_c|$ can be derived that gives the following inequalities for any integer $c \geq c_2(G)$.

$$2\binom{n}{\lfloor 2\rho(c) \rfloor} \leq |S_c| \leq \mathcal{B}(c) \leq 4^{\rho(c)}\binom{n}{2\rho(c)}$$

This indicates there is only minor room for improvement in Lemma 2 using these quantities. However, it is likely that in some cases a tighter or perhaps more interesting bound may be obtainable using other data-dependent quantities, which yield a tighter bound than Lemma 2. For example, it might be possible to extend the techniques used by (Chandran & Ram, 2004),

[6]A multigraph is a graph for which there may exist multiple edges between a pair of vertices.

[7]Q may be interpreted as the unique smallest positive rational number such that $\forall \{u,v\} \in E$, $Qw(u,v) \in \mathbb{Z}$.

[8]This is almost an immediate consequence of Karger's result that there are no more than $n^{2\alpha}$ "α-approximate" 2-cuts (Karger, 1999).

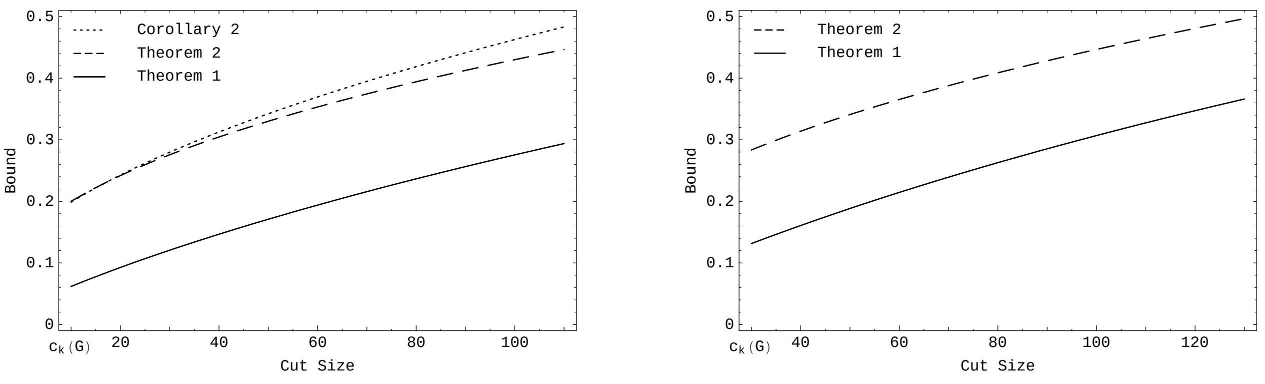

Figure 1. Comparison of cut bounds for $k = 2$ (left) and $k = 4$ (right).

who obtain bounds on the number of minimum 2-cuts in terms of graph radius, diameter, minimum degree, maximum degree, chordality, girth, and other properties of graphs. Their bounds can generally be tighter than those obtained by Karger's random contraction technique, and extending their techniques to the more general setting of bounding the number of labelings of a given cut size seems an interesting approach to improving the results derived above and reaching a better understanding of what the relevant factors are for the performance of this type of algorithm.

Additionally, there have been several proposed extensions of the mincut algorithm of (Blum & Chawla, 2001). In particular, (Joachims, 2003) proposes a constrained spectral partitioning algorithm that trades off cut size, training errors, and difference between class frequency ratios in the training vertices versus the test vertices. Finding data-dependent bounds on the prediction error for this learning bias (and related ones) remains an enticing open problem.

Acknowledgments

I thank Avrim Blum, Mugizi Rwebangira, Eric Xing, and the anonymous ICML reviewers for their insightful comments. I am also grateful to Stephen Fienberg and Eric Xing for indirect support of this research.

References

Benczúr, A., & Karger, D. (1996). Approximating s-t minimum cuts in $\tilde{O}(n^2)$ time. *ACM Symposium on Theory of Computing.*

Blum, A., & Chawla, S. (2001). Learning from labeled and unlabeled data using graph mincuts. *Proceedings of the International Conference on Machine Learning.*

Blum, A., Lafferty, J., Rwebangira, M., & Reddy, R. (2004). Semi-supervised learning using randomized mincuts. *International Conference on Machine Learning.*

Blum, A., & Langford, J. (2003). PAC-MDL bounds. *16th Annual Conference on Learning Theory.*

Chandran, L. S., & Ram, L. S. (2004). On the number of minimum cuts in a graph. *SIAM Journal of Discrete Mathematics*, *18*, 177–194.

Cheung, K. K. H., Cunningham, W. H., & Tang, L. (2005). Optimal 3-terminal cuts and linear programming. *Mathematical Programming.*

Cormen, T., Leiserson, C., Rivest, R., & Stein, C. (2001). *Introduction to algorithms.* MIT Press. 2nd edition.

Călinescu, G., Karloff, H., & Rabani, Y. (2000). An improved approximation algorithm for multiway cut. *Journal of Computer and System Sciences*, *60*, 564–574.

Dahlhaus, E., Johnson, D., Papadimitriou, C., Seymour, P., & Yannakakis, M. (1994). The complexity of multiterminal cuts. *SIAM Journal on Computing*, *23.*

Derbeko, P., El-Yaniv, R., & Meir, R. (2004). Explicit learning curves for transduction and application to clustering and compression algorithms. *Journal of Artificial Intelligence Research.*

Goldschmidt, O., & Hochbaum, D. S. (1994). A polynomial algorithm for the k-cut problem for fixed k. *Mathematics of Operations Research*, *19*, 24–37.

Joachims, T. (2003). Transductive learning via spectral graph partitioning. *Proceedings of the International Conference on Machine Learning.*

Karger, D. (1996). Minimum cuts in near-linear time. *ACM Symposium on the Theory of Computing.*

Karger, D. (1999). Random sampling in cut, flow, and network design problems. *Mathematics of Operations Research*, *24*, 383–413.

Karger, D. R., Klein, P., Stein, C., Thorup, M., & Young, N. E. (2004). Rounding algorithms for a geometric embedding of minimum multiway cut. *Mathematics of Operations Research*, *29*, 436–461.

Kleinberg, J., Sandler, M., & Slivkins, A. (2004). Network failure detection and graph connectivity. *15th ACM-SIAM Symposium on Discrete Algorithms.*

A. Proof of Lemma 1

Proof. For completeness, I now state the proof of Lemma 1. For a given $h \in H$, let T_h be a constant representing the actual number of mistakes h makes on the entire graph. All probabilities are defined with respect to the random variable $\hat{R}(h)$ (a function of λ). It is true that for any $\eta \in (0,1)$,

$$\begin{aligned}\eta \geq & \quad \mathcal{P}r\left\{F_{T_h}(\hat{R}(h)n_\ell) < \eta\right\} \\ \geq & \quad \mathcal{P}r\left\{T_h > \max\left\{T \,\middle|\, F_T(\hat{R}(h)n_\ell) \geq \eta, T \in \mathbb{Z}\right\}\right\} \\ = & \quad \mathcal{P}r\left\{R(h) > R_{max}(\hat{R}(h), \eta)\right\}\end{aligned}$$

Note that $\hat{R}(h)n_\ell \in \{T | F_T(\hat{R}(h)n_\ell) \geq \eta, T \in \mathbb{Z}\}$ so the set is nonempty. Applying the union bound as follows completes the proof.

$$\begin{aligned}&\mathcal{P}r\left\{\forall h \in H, \quad R(h) \leq R_{max}(\hat{R}(h), q(h)\delta)\right\} \\ &\geq 1 - \sum_{h \in H} \mathcal{P}r\left\{R(h) > R_{max}(\hat{R}(h), q(h)\delta)\right\} \\ &\geq 1 - \sum_{h \in H} q(h)\delta \ \geq 1 - \delta\end{aligned}$$

□

B. Proof of Lemma 2

Proof. The proof of this lemma follows a technique used by David Karger to prove a related result (Karger, 1999).

If $\mathfrak{s}(c) \geq n$, the bound is k^n, which is the total number of possible ways to label the vertices, so the result holds for this special case.

Now for the most interesting case of $\mathfrak{s}(c) < n$, I use the following notion of *edge contraction*. When contracting an edge $\{u, v\}$, the graph after contraction appears identical to the graph before, except that u and v are replaced by a single vertex w, any edges between u and v, including the $\{u, v\}$ edge being contracted, are absent from the new graph, any edge $\{u, x\}$ for $x \neq v$ is replaced by $\{w, x\}$, and any edge $\{v, y\}$ for $y \neq u$ is replaced by $\{w, y\}$ in the graph after contraction. This process may result in multiple instances of a given edge (e.g., if there is a $\{u, x\}$ and $\{v, x\}$ in the graph before contraction, then there will be two $\{w, x\}$ edges after contraction). We keep all of these multiple edges in the post-contraction graph (i.e., the graph after contraction may actually be a multigraph).

Now consider an iterative algorithm that, on each iteration, selects an edge uniformly at random from the current graph G_i and contracts it to make a new graph G_{i-1} for the next iteration. Thus, on each iteration, there is one fewer vertex than on the previous iteration. Consider starting this algorithm with $G_n = G$ and running it until there are $\mathfrak{s}(c)$ vertices.

For a given labeling $h \in \mathcal{H}$, for $t \in \{\mathfrak{s}(c)+1, \mathfrak{s}(c)+2, \ldots, n\}$, define *contracted labelings* h_t as follows. $\forall x \in V,\ h_n(x) = h(x)$. For $t \leq n$, if $\{u_t, v_t\}$ is the edge in $G_t = (V_t, E_t)$ that is contracted by the algorithm to make graph G_{t-1}, replacing u_t and v_t with vertex w_{t-1} in G_{t-1}, then $\forall x \in V_t \setminus \{u_t, v_t\},\ h_{t-1}(x) = h_t(x)$ and

$$h_{t-1}(w_{t-1}) = \begin{cases} h_t(u_t), & \text{if } h_t(u_t) = h_t(v_t) \\ 0, & \text{otherwise.} \end{cases}$$

I say that h *survives* contraction to r vertices if $\nexists t \in \{r+1, r+2, \ldots, n\}$ such that $h_t(u_t) \neq h_t(v_t)$. Note that on any given run of the algorithm, there are precisely k^r labelings that survive contraction to r vertices, corresponding to the k^r labelings of G_r with labels in $\{1, 2, \ldots, k\}$.

At the stage of the contraction algorithm in which there are some number $r > \mathfrak{s}(c)$ of vertices, the minimum k-cut size in the contracted (r-vertex) graph G_r must be at least $c_k(G)$ (since every k-cut in the contracted graph corresponds to a k-cut of the same size in G). This implies that the average degree is at least $\frac{c_k(G)}{k-1}$, so there are at least $\frac{rc_k(G)}{2(k-1)}$ edges in the contracted graph G_r. This implies that, given that h survives contraction to r vertices, the probability of the algorithm picking an edge $\{u_r, v_r\}$ such that $h_r(u_r) \neq h_r(v_r)$ is at most $c(h)\frac{2(k-1)}{rc_k(G)} = \frac{2(k-1)\rho(h)}{r}$. This bounds the probability h does *not* survive the next contraction. Therefore, for any h with $c(h) \leq c$, the probability that h survives contraction to $\mathfrak{s}(c)$ vertices is at least

$$\begin{aligned}&\left(1 - \frac{2(k-1)\rho(c)}{n}\right)\left(1 - \frac{2(k-1)\rho(c)}{n-1}\right)\cdots\left(1 - \frac{2(k-1)\rho(c)}{\mathfrak{s}(c)+1}\right) \\ &= \binom{\mathfrak{s}(c)}{2(k-1)\rho(c)}\binom{n}{2(k-1)\rho(c)}^{-1}\end{aligned}$$

Once the graph is contracted to $\mathfrak{s}(c)$ vertices, suppose the algorithm stops and selects a labeling uniformly at random from all labelings that survive contraction to these $\mathfrak{s}(c)$ vertices and outputs that labeling. Since there are $k^{\mathfrak{s}(c)}$ such labelings, the probability the algorithm outputs a particular labeling given that it survives contraction to $\mathfrak{s}(c)$ vertices is $k^{-\mathfrak{s}(c)}$.

Putting all of this together, for every $h \in \mathcal{H}$ such that $c(h) \leq c$, the probability that the algorithm outputs h is at least

$$k^{-\mathfrak{s}(c)}\binom{\mathfrak{s}(c)}{2(k-1)\rho(c)}\binom{n}{2(k-1)\rho(c)}^{-1} = \mathcal{B}(c)^{-1}$$

Therefore, at most $\mathcal{B}(c)$ such labelings can exist. □