Data driven semi-supervised learning

Maria-Florina Balcan
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
ninamf@cs.cmu.edu

Dravyansh Sharma
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
dravyans@cs.cmu.edu

Abstract

We consider a novel data driven approach for designing semi-supervised learning algorithms that can effectively learn with only a small number of labeled examples. We focus on graph-based techniques, where the unlabeled examples are connected in a graph under the implicit assumption that similar nodes likely have similar labels. Over the past two decades, several elegant graph-based semi-supervised learning algorithms for inferring the labels of the unlabeled examples given the graph and a few labeled examples have been proposed. However, the problem of how to create the graph (which impacts the practical usefulness of these methods significantly) has been relegated to heuristics and domain-specific art, and no general principles have been proposed. In this work we present a novel data driven approach for learning the graph and provide strong formal guarantees in both the distributional and online learning formalizations. We show how to leverage problem instances coming from an underlying problem domain to learn the graph hyperparameters for commonly used parametric families of graphs that provably perform well on new instances from the same domain. We obtain low regret and efficient algorithms in the online setting, and generalization guarantees in the distributional setting. We also show how to combine several very different similarity metrics and learn multiple hyperparameters, our results hold for large classes of problems. We expect some of the tools and techniques we develop along the way to be of independent interest, for data driven algorithms more generally.

1 Introduction

In recent years machine learning has found gainful application in diverse domains. A major bottleneck of the currently used approaches is the heavy dependence on expensive labeled data. Advances in cheap computing and storage have made it relatively easier to store and process large amounts of unlabeled data. Therefore, an important focus of the present research community is to develop general domain-independent methods to learn effectively from the unlabeled data, along with a small amount of labels. Achieving this goal would significantly elevate the state-of-the-art machine intelligence, which currently lags behind the human capability of learning from a few labeled examples. Our work is a step in this direction, and provides algorithms and guarantees that enable fundamental techniques for semi-supervised learning to provably adapt to problem domains.

Graph-based approaches have been popular for learning from unlabeled data for the past two decades [Zhu and Goldberg, 2009]. Labeled and unlabeled examples form the graph nodes and (possibly weighted) edges denote the feature similarity between examples. The graph therefore captures how each example is related to other examples, and by optimizing a suitably regularized objective over it one obtains an efficient discriminative, nonparametric method for learning the labels. There are several well-studied ways to define and regularize an objective on the graph [Chapelle et al., 2010],

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

Table 1: Optimization objectives for graph-based SSL. $D_{ij} := \mathbb{I}[i=j]\sum_k W_{ik}, \mathcal{L} := D^{-1/2}(D - W)D^{-1/2}$ and the objective is $l(f) = \alpha \sum_{u \in L}(f(u) - y_u)^2 + \beta H(f, W) + \gamma \|f\|^2$.

Algorithm	(α, β, γ)	$H(f, W), \|\cdot\|$	Constraints on f
Mincut	$(\infty, 1, 0)$	$f^T(D-W)f$	$f \in \{0,1\}^n$
Harmonic function	$(\infty, 1, 0)$	$f^T(D-W)f$	$f \in [0,1]^n$
Normalized cut	$(\infty, 1, 0)$	$f^T(D-W)f$	$f^T\mathbf{1} = 0, f^Tf = n^2, f \in [0,1]^n$
Label propagation	$(1, \mu, 1)$	$f^T\mathcal{L}f, \|\cdot\|_2$	$f \in [0,1]^n$

and all yield comparable results which strongly depend on the graph used. A general formulation is described as follows, variations on which are noted under related work.

Problem formulation Given sets L and U of labeled and unlabeled examples respectively, and a similarity metric d over the data, the goal is to use d to extrapolate labels in L to U. A graph G is constructed with $L + U$ as the nodes and weighted edges W with $w(u, v) = g(d(u, v))$ for some $g : \mathbb{R}_{\geq 0} \to \mathbb{R}_{\geq 0}$. We seek labels $f(\cdot)$ for nodes u of G which minimize a regularized loss function $l(f) = \alpha \sum_{v \in L} \hat{l}(f(v), y_v) + \beta H(f, W) + \gamma \|f\|^2$, under some constraints on f. The objective H captures the *smoothness* (regularization) induced by the graph (see Table 1 for examples) and $\hat{l}(f(v), y_v)$ is the misclassification loss (computed here on labeled examples).

The graph G takes a central position in this formulation. However, the majority of the research effort on this problem has focused on how to design and optimize the regularized loss function $l(f)$, the effectiveness of which crucially depends on G. There is no known principled study on how to build G and prior work largely treats this as a domain-specific art [Chapelle et al., 2010]. Is it possible to acquire the required domain expertise, without involving human experts? In this work we provide an affirmative answer by formulating graph selection as *data-driven design.* More precisely, we are required to solve not only one instance, but multiple instances of the underlying algorithmic problem that come from the same domain [Gupta and Roughgarden, 2016, Balcan, 2020]. We show learning a near-optimal graph over commonly used infinite parameterized families is possible in both online and distributional settings. In the process we generalize and extend data-driven learning techniques, and obtain practical methods to build the graphs with strong guarantees. In particular, we show how the techniques can learn several parameters at once, and also learn a broader class of parameters than previously known.

Our contributions and key challenges. We present a first theoretically grounded work for graph-based learning from limited labeled data, while extending general data-driven design techniques.

Data-driven algorithm design. Firstly, for one dimensional loss functions, we show a novel structural result which applies when discontinuities (for loss as function of the algorithm parameter) occur along roots of exponential polynomials with random coefficients with bounded joint distributions (previously known only for algebraic polynomials in Balcan et al. [2020b]). This is crucial for showing learnability in the Gaussian graph kernels setting. Secondly, Balcan et al. [2020b] only applies when the discontinuities occur along algebraic curves with random coefficients in just two dimensions. By a novel algebraic and learning theoretic argument we are able to analyze higher (arbitrary constant number of) dimensions, making the technique much more generally applicable.

Semi-supervised learning. We examine commonly used parameterized graph families, denoted by general notation $G(\rho)$, where ρ corresponds to a semi-supervised learning algorithm. We consider online and distributional settings, providing efficient algorithms to obtain low regret and low error respectively for learning ρ. Most previously studied settings involve polynomially many discontinuities for loss as function of the hyperparameter ρ on a fixed instance, implying efficient algorithms, which may not be the case for our setting. To resolve this, we describe efficient semi-bandit implementations, and in particular introduce a novel min-cut and flow recomputation algorithm on graphs with continuously changing edge weights which may be of independent interest. For the distributional setting, we provide asymptotically tight bounds on the pseudodimension of the parameter learning problem. Our lower bounds expose worst case challenges, and involve precise constructions of problem instances by setting node similarities which make assigning labels provably hard.

Our techniques are extremely general and are shown to apply for nearly all combinations of optimization algorithms (Table 1) and parametric graph families (Definition 1).

Related work *Semi-supervised learning* is a paradigm for learning from labeled and unlabeled data (Zhu and Goldberg [2009]). It resembles human learning behavior more closely than fully supervised and fully unsupervised models (Zhu et al. [2007], Gibson et al. [2013]). A popular approach for semi-supervised learning is to optimize a graph-based objective. Several methods have been proposed to predict labels given a graph including st-mincuts (Blum and Chawla [2001]), soft mincuts that optimize a harmonic objective (Zhu et al. [2003]), label propagation (Xiaojin and Zoubin [2002]), and many more (Shi and Malik [2000], Belkin et al. [2006]). All algorithms have comparable performance provided the graph G encodes the problem well [Zhu and Goldberg, 2009]. However, it is not clear how to create the graph itself on which the extensive literature stands, barring some heuristics (Zhu et al. [2005], Zemel and Carreira-Perpiñán [2004]). Sindhwani et al. [2005] construct *warped* kernels aligned with the data geometry, but the performance may vary strongly with warping and it is not clear how to optimize over it. We provide the first techniques that yield provably near-optimal graphs.

Gupta and Roughgarden [2016, 2017] define a formal learning framework for selecting algorithms from a family of heuristics or setting hyperparameters. It is further developed by Balcan et al. [2017] and noted as a fundamental algorithm design perspective [Blum, 2020]. It has been successfully applied to several combinatorial problems like integer programming and clustering [Balcan et al., 2018a, 2019, 2018c] and for giving powerful guarantees like adversarial robustness, adaptive learning and differential privacy [Balcan et al., 2018b, 2020a,c, Vitercik et al., 2019, Balcan et al., 2020e,d]. Balcan et al. [2018b, 2020b] introduce general data-driven design techniques under some smoothness assumptions. We extend the techniques to significantly broader problem settings, and investigate the structure of graph-based label learning formulation to apply the new techniques.

2 Setup and definitions

We are given some unlabeled points $U \subset \mathcal{X}$ and labeled points $L \subset \mathcal{X} \times \mathcal{Y}$, such that $|L| + |U| = n$. One constructs a graph G by placing (possibly weighted) edges $w(u, v)$ between pairs of data points u, v which are 'similar', and labels for the unlabeled examples are obtained by optimizing some graph-based score. We have an oracle O which on querying provides us the labeled and unlabeled examples, and we need to pick graph $G(\rho)$ from some family $\mathcal{G}$ of graphs, parameterized using a parameter $\rho \in \mathcal{P}$. We commit to using some graph labeling algorithm $A(G, L, U)$ (abbreviated as $A_{G,L,U}$) which provides labels for examples in U, and we should pick a ρ such that $A(G(\rho), L, U)$ results in small error in its predictions on U. More formally, for a loss function $l : \mathcal{Y} \times \mathcal{Y} \to [0, 1]$ and a target labeling $\tau : U \to \mathcal{Y}$, we need to find $\operatorname{argmin}_{\rho \in \mathcal{P}} l_{A(G(\rho),L,U)} := \sum_U l(A_{G(\rho),L,U}(u), \tau(u))$.

We will now describe some graph families $\mathcal{G}$ and algorithms $A_{G,L,U}$. We assume there is a feature based *similarity function* $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$, a metric which monotonically captures pairwise similarity. Commonly used parametric methods to build a graph using the similarity function follow.

Definition 1. *Graph kernels.*[1]

a) Threshold graph, $G(r)$. Parameterized by a threshold r, we set $w(u, v) = \mathbb{I}[d(u, v) \leq r]$.
b) Polynomial kernel, $G(\tilde{\alpha})$. $w(u, v) = (\tilde{d}(u, v) + \tilde{\alpha})^d$ for fixed degree d, parameterized by $\tilde{\alpha}$.
c) Gaussian RBF or exponential kernel, $G(\sigma)$. $w(u, v) = e^{-d(u,v)^2/\sigma^2}$, parameterized by σ.

Remark 1. *Another popular family of graphs used in practice is the k nearest neighbor graphs, where $k \in \{0, 1, \ldots, n-1\}$, n is the number of nodes in the graph, is the parameter. Even though k-NN graphs may result in different graphs the ones considered in the paper, learning how to build an optimal graph over the algorithm family $G(k)$ is much simpler. Online learning of the parameter k in this setting can be recognized as an instance of learning with experts advice for a finite hypothesis class (Section 3.1 of Shalev-Shwartz et al. [2011]), where an upper bound of $O(\sqrt{T \log n})$ is known for the Weighted Majority algorithm. Online-to-batch conversion provides generalization guarantees in the distributional setting (Section 5 of Shalev-Shwartz et al. [2011]). We remark that our algorithm families need more sophisticated analysis due to continuous ranges of the algorithm parameters.*

[1] With some notational abuse, we have d as the integer polynomial degree, and $d(\cdot, \cdot)$ as the similarity function. Common choices are setting $d(u, v)$ as the Euclidean norm and $\tilde{d}(u, v)$ as the dot product when $u, v \in \mathbb{R}^n$.

The threshold graph adds (unweighted) edges to G only when the examples are closer than some $r \geq 0$. We refer to this setting by the *unweighted graph* setting, and the others by the *weighted graph* setting. The similarity function $\tilde{d}(u, v)$ in Definitions 1b increases monotonically with similarity of examples (as opposed to the other two). Once the graph is constructed using one of the above kernels, we can assign labels using some algorithm $A_{G,L,U}$. A popular, effective approach is to optimize a quadratic objective $\frac{1}{2}\sum_{u,v} w(u, v)(f(u) - f(v))^2$. f may be discrete, $f(u) \in \{0, 1\}$ corresponds to finding a mincut separating the oppositely labeled vertices [Blum and Chawla, 2001], or $f \in [0, 1]$ may be continuous and we can round f to obtain the labels [Zhu et al., 2003]. These correspond to the *mincut* and *harmonic function* algorithms respectively from Table 1.

We also need some well-known definitions from prior work (Appendix A). In particular, we use *dispersion* from [Balcan et al., 2020b]. The sequence of random loss functions $l_1, \ldots, l_T$ is β-*dispersed* for the Lipschitz constant L if, for all T and for all $\epsilon \geq T^{-\beta}$, $\mathbb{E}\left[\max_{\rho,\rho' \in \mathcal{C}, \|\rho-\rho'\|_2 \leq \epsilon} \left|\{t \in [T] \mid l_t(\rho) - l_t(\rho') > L\|\rho - \rho'\|_2\}\right|\right] \leq \tilde{O}(\epsilon T)$.

3 New general dispersion-based tools for data-driven design

We present new general tools for analyzing data-driven algorithms. Our new tools apply to a very broad class of algorithm design problems, for which we derive sufficient *smoothness* conditions to infer dispersion of a random sequence of problems, i.e. the algorithmic performance as a function of the algorithm parameters is dispersed. Recall that dispersion, roughly speaking, captures the rate at which discontinuities concentrate in any region of the domain. Balcan et al. [2020b] provide a general tool for verifying dispersion if non-Lipschitzness occurs along roots of (algebraic) polynomials in one and two dimensions. We improve upon their results in two major ways.

Our first result is that dispersion for one-dimensional loss functions follows when the points of discontinuity occur at the roots of exponential polynomials if the coefficients are random, lie within a finite range, and are drawn according to a bounded joint distribution. The key idea is use algebraic arguments and Taylor series approximation to show that for any small interval containing roots of the random exponential polynomial, the corresponding sets of coefficients lie on $n - 1$ dimensional linear subspaces with a probability measure proportional to the length of the interval (Appendix C.3).

Theorem 2. *Let $\phi(x) = \sum_{i=1}^{n} a_i e^{b_i x}$ be a random function, such that coefficients a_i are real and of magnitude at most R, and distributed with joint density at most κ. Then for any interval I of width at most ϵ, P(ϕ has a zero in I)$\leq \tilde{O}(\epsilon)$ (dependence on b_i, n, κ, R suppressed).*

Proof Sketch. For $n = 1$ there are no roots, so assume $n > 1$. Suppose ρ is a root of $\phi(x)$. Then $\mathbf{a} = (a_1, \ldots, a_n)$ is orthogonal to $\varrho(\rho) = (e^{b_1\rho}, \ldots, e^{b_n\rho})$ in $\mathbb{R}^n$. For a fixed ρ, the set S_ρ of coefficients $\mathbf{a}$ for which ρ is a root of $\phi(y)$ lie along an $n - 1$ dimensional linear subspace of $\mathbb{R}^n$. Now ϕ has a root in any interval I of length ϵ, exactly when the coefficients lie on S_ρ for some $\rho \in I$. The desired probability is therefore upper bounded by $\max_\rho \textsc{Vol}(\cup S_y \mid y \in [\rho - \epsilon, \rho + \epsilon])/\textsc{Vol}(S_y \mid y \in \mathbb{R})$ which we will show to be $\tilde{O}(\epsilon)$. The key idea is that if $|\rho - \rho'| < \epsilon$, then $\varrho(\rho)$ and $\varrho(\rho')$ are within a small angle $\theta_{\rho,\rho'} = \tilde{O}(\epsilon)$ for small ϵ (the probability bound is vacuous for large ϵ). But any point in S_ρ is at most $\tilde{O}(\theta_{\rho,\rho'})$ from a point in $S_{\rho'}$, which implies the desired bound. □

We further go beyond single-parameter discontinuties, which occur as points along a line to general small dimensional parameter spaces $\mathbb{R}^p$, where discontinuties can occur along algebraic hypersurfaces. We employ tools from algebraic geometry to establish a bound on shattering of algebraic hypersurfaces by axis-aligned paths (Theorem 3), which implies dispersion using a VC dimension based argument (Theorem 4). Our result is a first general sufficient condition for dispersion for any constant number p of parameters, and applies to a broad class of algorithm families. Full proofs are in Appendix C.4.

Theorem 3. *There is a constant k depending only on d and p such that axis-aligned line segments in $\mathbb{R}^p$ cannot shatter any collection of k algebraic hypersurfaces of degree at most d.*

Proof Sketch. Let $\mathcal{C}$ denote a collection of k algebraic hypersurfaces of degree at most d in $\mathbb{R}^p$. We say that a subset of $\mathcal{C}$ is *hit* by a line segment if the subset is exactly the set of curves in $\mathcal{C}$ which intersect the segment. We can upper bound the subsets of $\mathcal{C}$ hit by line segments in a fixed axial direction x in two steps. Along a fixed line, Bezout's Theorem bounds the number of intersections

and therefore subsets hit by different line segments. Using the Tarski–Seidenberg Theorem, the lines along x can be shown to belong to equivalence classes corresponding to cells in the cylindrical algebraic decomposition of the projection of the hypersurfaces, orthogonal to x. Finally, this extends to axis-aligned segments by noting they may hit only p times as many subsets. □

Theorem 4. *Let $l_1, \ldots, l_T : \mathbb{R}^p \to \mathbb{R}$ be independent piecewise L-Lipschitz functions, each having discontinuities specified by a collection of at most K algebraic hypersurfaces of bounded degree. Let L denote the set of axis-aligned paths between pairs of points in $\mathbb{R}^p$, and for each $s \in L$ define $D(T, s) = |\{1 \leq t \leq T \mid l_t \text{ has a discontinuity along } s\}|$. Then we have $\mathbb{E}[\sup_{s \in L} D(T, s)] \leq \sup_{s \in L} \mathbb{E}[D(T, s)] + O(\sqrt{T \log(TK)})$.*

4 Learning the graph online

We will warm up this section with a simple example demonstrating the need for and challenges posed by the problem of learning how to build a good graph from data. We consider the setting of learning thresholds for unweighted graphs (Definition 1a). We give a simple demonstration that in a single instance *any threshold* may be optimal for labelings consistent with graph smoothness assumptions, therefore providing motivation for the learning in our setting. Our construction (depicted in Figure 1) captures the intuition that any unlabeled point may get weakly connected to examples from one class for a small threshold but may get strongly connected to another class as the threshold is increased to a larger value. Therefore depending on the unknown true label either threshold may be optimal or suboptimal, and it makes sense to learn the correct value through repeated problem instances.

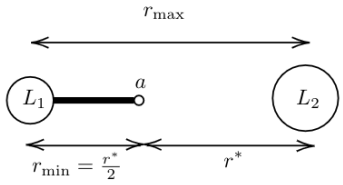


Figure 1: $G(r)$ connects a to nodes in L_1 for $r_{\min} \leq r < r^*$. $|L_1| < |L_2|$.

Theorem 5. *Let $r_{\min}$ denote the smallest value of threshold r for which every unlabeled node of $G(r)$ is reachable from some labeled node, and $r_{\max}$ be the smallest value of threshold r for which $G(r)$ is the complete graph. There exists a data instance (L, U) such that for any $r_\zeta = \zeta r_{\min} + (1 - \zeta) r_{\max}$ for $\zeta \in (0, 1)$, there exists a set of labelings $\mathcal{U}$ of the unlabeled points such that for some $U_\zeta, \bar{U}_\zeta \in \mathcal{U}$, r_ζ minimizes $l_{A(G(r), L, U_\zeta)}$ but not $l_{A(G(r), L, \bar{U}_\zeta)}$.*

4.1 Dispersion and online learning

We consider the problem of learning the graph online. In this setting, we are presented with instances of the problem online and want to learn the best value of the parameter ρ while making predictions. For now, we assume we get all the labels for past instances which may be used to determine the loss for any ρ (*full information*). At time $t \in [T]$ we predict $\rho_t \in \mathcal{P}$ (the parameter space) based on labeled and unlabeled examples $(L_i, U_i), i \in [t]$ and past labels $\tau(u)$ for each $u \in U_j, j < t$ and seek to minimize regret $R_T := \sum_{t=1}^{T} l_{A(G(\rho_t), L_t, U_t)} - \min_{\rho \in \mathcal{P}} \sum_{t=1}^{T} l_{A(G(\rho), L_t, U_t)}$.

A key difficulty in the online optimization for our settings is that the losses are discontinuous functions of the graph parameters ρ. We can efficiently solve this problem if we can show that the loss functions are dispersed, in fact $\frac{1}{2}$-dispersed functions may be learned with $\tilde{O}(\sqrt{T})$ regret (Balcan et al. [2018b, 2020c]). Algorithm 1 adapts the general algorithm of Balcan et al. [2018b] to data-driven graph-based learning and achieves low regret for dispersed functions. Recall that dispersion roughly says that the discontinuities in the loss function are not too concentrated. We will exploit an assumption that the embeddings are approximate, so small random perturbations to the distance metric will likely not affect learning. This mild distributional assumption allows us to show that Algorithm 1 learns ρ.

Algorithm 1 Data-driven Graph-based SSL

Input: Graphs G_t with labeled and unlabeled nodes (L_t, U_t), node similarities $d(u,v)_{u,v \in L_t \cup U_t}$.

Hyperparameter: step size parameter $\lambda \in (0,1]$.
Output: Graph parameter ρ_t for times $t = 1, 2, \ldots, T$.
Set $w_1(\rho) = 1$ for all $\rho \in \mathbb{R}_{\geq 0}$.
for $t = 1, 2, \ldots, T$ **do**
  Sample ρ with probability $p_t(\rho) = \frac{w_t(\rho)}{W_t}$, output as ρ_t, where $W_t := \int_C w_t(\rho) d\rho$.
  Compute average loss function $l_t(\rho) = \frac{1}{|U_t|} \sum_{u \in U} l(A_{G_t(\rho), L_t, U_t}(u), \tau(u))$.
  For each $\rho \in C,$ set $w_{t+1}(\rho) = e^{\lambda u_t(\rho)} w_t(\rho)$, where $u_t(\rho) = 1 - l_t(\rho) \in [0,1]$.

4.1.1 Dispersion of the loss functions.

We first show dispersion for the unweighted graph family, with threshold parameter r. Here dispersion follows from a simple assumption that the distance $d(u,v)$ for any pair of nodes u, v follows a κ-bounded distribution[2], and observing that discontinuities of the loss (as a function of r) must lie on the set of distances $d(u,v)$ in the samples (for any optimization algorithm). Using a VC dimension argument on the loss sequence we show (Appendix C.1).

Theorem 6. *Let $l_1, \ldots, l_T : \mathbb{R} \to \mathbb{R}$ denote an independent sequence of losses as a function of parameter r, when the graph is created using a threshold kernel $w(u,v) = \mathbb{I}[d(u,v) \leq r]$ and labeled by applying any algorithm on the graph. If $d(u,v)$ follows a κ-bounded distribution for any u, v, the sequence is $\frac{1}{2}$-dispersed, and the regret of Algorithm 1 is $\tilde{O}(\sqrt{T})$.*

We also show dispersion for weighted graph kernels, but under slightly stronger assumptions. We assume that distances $d(u,v)$ are jointly κ-bounded on a closed and bounded support. The plan is show that if the similarity function is smooth, then the discontinuities lie along roots of a polynomial with random finite coefficients with a κ'-bounded joint distribution, and use results for dispersion analysis from Balcan et al. [2020b]. We establish the following theorem (proof in Appendix C.2).

Theorem 7. *Let $l_1, \ldots, l_T : \mathbb{R} \to \mathbb{R}$ denote an independent sequence of losses as a function of $\tilde{\alpha}$, for graph with edges $w(u,v) = (\tilde{d}(u,v) + \tilde{\alpha})^d$ labeled by optimizing the quadratic objective $\sum_{u,v} w(u,v)(f(u) - f(v))^2$. If $\tilde{d}(u,v)$ follows a κ-bounded distribution with a closed and bounded support, the sequence is $\frac{1}{2}$-dispersed, and the regret of Algorithm 1 may be upper bounded by $\tilde{O}(\sqrt{T})$.*

Proof Sketch. The solution of the quadratic objective is given by $f_U = (D_{UU} - W_{UU})^{-1} W_{UL} f_L$. The key technical challenge is to show that for any $u \in U$, $f(u) = 1/2$ is a polynomial equation in $\tilde{\alpha}$ with degree at most nd, and coefficients that are jointly $K\kappa$-bounded, where K is a constant that only depends on d and the support of $\tilde{d}(u,v)$. Therefore the labeling, and consequently also the loss function, may only change when $\tilde{\alpha}$ is a root of one of $|U|$ polynomials of degree at most dn. The dispersion result is now a simple application of results from Balcan et al. [2020b]. □

Remark 2. *Theorem 6 applies to all objectives in Table 1, and Theorem 7 extends to all except the mincut. We can also extend the analysis to obtain similar results when using the exponential kernel $w(u,v) = e^{-||u-v||^2/\sigma^2}$. The results of Balcan et al. [2020b] no longer directly apply as the points of discontinuity are no longer roots of polynomials, and we need to analyze points of discontinuities of exponential polynomials, i.e. $\phi(x) = \sum_{i=1}^{k} a_i e^{b_i x}$ (See Section 3 and Appendix C.3).*

Remark 3 (Extension to local and global classification Zhou et al. [2004])**.** *Above results can be extended to the classification algorithm used in Zhou et al. [2004]. The key observation is that the labels are given by a closed-form matrix, $f^* = (I - \alpha D^{-1/2} W D^{1/2})Y$ or $f^* = (D - \alpha W)Y$ (for the two variants considered). For threshold graphs $G(r)$, the regret bound in Theorem 6 applies to any classification algorithm. Extension to polynomial kernels $G(\tilde{\alpha})$ is described below. For fixed α (in the notation of Zhou et al. [2004], in expression for f^* above), the discontinuities in the loss as a function of the parameter $\tilde{\alpha}$ lie along roots of polynomials in the parameter $\tilde{\alpha}$ and therefore the same proof as Theorem 7 applies (essentially we get polynomial equations with slightly different but still*

[2] A density function $f : \mathbb{R} \to \mathbb{R}$ is κ-bounded if $\max_{x \in \mathbb{R}}\{f(x)\} \leq \kappa$. $\mathcal{N}(\mu, \sigma)$ is $\frac{1}{2\pi\sigma}$-bounded for any μ.

K-bounded coefficients). On the other hand, if we consider α as another graph parameter, we can still learn the kernel parameter $\tilde{\alpha}$ together with α by applying Theorem 18 and Theorem 4 (instead of Theorem 19) in the proof of Theorem 7.

4.1.2 Combining several similarity measures.

Multiple natural metrics often existin multimodal semi-supervised learning [Balcan et al., 2005]. Different metrics may have their own advantages and issues and often a weighted combination of metrics, say $\sum_i \rho_i d_i(\cdot,\cdot)$, works better than any individual metric. The combination weights ρ_i are additional graph hyperparameters. A combination of metrics is known to boost performance theoretically and empirically for linkage-based clustering [Balcan et al., 2019]. However the argument therein crucially relies on the algorithm depending on relative distances and not the actual values, and therefore does not extend directly to our setting. We develop a first general tool for analyzing dispersion for multi-dimensional parameters (Section 3), which implies the multi-parameter analogue of Theorem 7, stated below. See Appendix C.4 for proof details.

Theorem 8. *Let $l_1, \ldots, l_T : \mathbb{R}^p \to \mathbb{R}$ denote an independent sequence of losses as a function of parameters $\rho_i, i \in [p]$, when the graph is created using a polynomial kernel $w(u,v) = (\sum_{i=1}^{p-1} \rho_i \tilde{d}(u,v) + \rho_p)^d$ and labeled by optimizing the quadratic objective $\sum_{u,v} w(u,v)(f(u) - f(v))^2$. If $\tilde{d}(u,v)$ follows a κ-bounded distribution with a closed and bounded support, the sequence is $\frac{1}{2}$-dispersed, and the regret of Algorithm 1 may be upper bounded by $\tilde{O}(\sqrt{T})$.*

4.1.3 Semi-bandit setting and efficient algorithms.

Online learning with full information is usually inefficient in practice since it involves computing and working with the entire domain of hyperparameters. For our setting in particular this is computationally infeasible for weighted graphs since the number of pieces (in loss as a piecewise constant function of the parameter) may be exponential in the worst case (see Section 5). Fortunately we have a workaround provided by Balcan et al. [2020b] where dispersion implies learning in a semi-bandit setting as well. This setting differs from the full information online problem as follows. In each round as we select the parameter ρ_i, we only observe losses for a single interval containing ρ_i (as opposed to the entire domain). We call the set of these observable intervals the *feedback set*, and these provide a partition of the domain.

Algorithm 2 Efficient Data-driven Graph-based SSL

Input: Graphs G_t with labeled and unlabeled nodes (L_t, U_t), node similarities $d(u,v)_{u,v \in L_t \cup U_t}$.

Hyperparameter: step size parameter $\lambda \in (0,1]$.
Output: Graph parameter ρ_t for times $t = 1, 2, \ldots, T$.
Set $w_1(\rho) = 1$ for all $\rho \in C$
for $t = 1, 2, \ldots, T$ **do**
  Sample ρ with probability $p_t(\rho) = \frac{w_t(\rho)}{W_t}$, output as ρ_t, where $W_t := \int_C w_t(\rho) d\rho$..
  Compute the feedback set $A^{(t)}(\rho)$ containing ρ_t.
  For example, for the min-cut objective use Algorithm 3 (Appendix C.5.1) and set $A^{(t)}(\rho) = \textsc{DynamicMinCut}(G_t, \rho_t, 1/\sqrt{T})$. For the quadratic objective use Algorithm 4 (Appendix C.5.2) to set $A^{(t)}(\rho) = \textsc{HarmonicFeedbackSet}(G_t, \rho_t, 1/\sqrt{T})$.
  Compute average loss function $l_t(\rho) = \frac{1}{|U_t|} \sum_{u \in U} l(A_{G_t(\rho), L_t, U_t}(u), \tau(u))$.
  For each $\rho \in C$, set $w_{t+1}(\rho) = e^{\lambda \hat{l}_t(\rho)} w_t(\rho)$, where $\hat{l}_t(\rho) = \frac{\mathbb{I}[\rho \in A^{(t)}(\rho)]}{\int_{A^{(t)}(\rho)} p_t(\rho)} l_t(\rho)$.

For the case of learning the unweighted threshold graph, computing the feedback set containing a given r is easy as we only need the next and previous thresholds from among the $O(n^2)$ values of pairwise distances where loss may be discontinuous in r. We present algorithms for computing the semi-bandit feedback sets (constant performance interval containing any σ) for the weighted graph setting (Definition 1c). We propose a novel hybrid combinatorial-continuous algorithm for the mincut objective (Algorithm 3, Appendix C.5.1) which re-computes the mincut in a graph with dynamic edge weights by flow decomposition and careful flow augmentation as σ is varied until a new mincut

is detected. For the harmonic objective, we can obtain similar efficiency (Algorithm 4, Appendix C.5.2). We seek points where $f_u(\sigma) = \frac{1}{2}$ for some $u \in U$ closest to given σ_0. For each u we can find the local minima of $\left(f_u(\sigma) - \frac{1}{2}\right)^2$ or simply the root of $f_u(\sigma) - \frac{1}{2}$ using gradient descent or Newton's method. The gradient computation uses matrix inversion which can be computed in $O(n^3)$ time, and we can obtain quadratic convergence rates for finding the root. Formally, we establish Theorem 9 (Appendix C.5).

Theorem 9. *For the each objective in Table 1 and exponential kernel (Definition 1c), there exists an algorithm which outputs the interval containing σ in time $\tilde{O}(n^4)$.*

5 Distributional setting

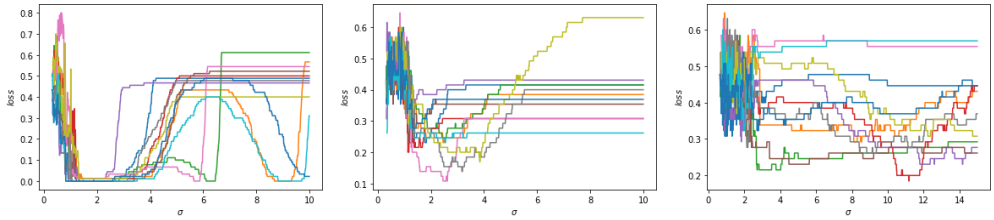
In the distributional setting, we are presented with instances of the problem assumed to be drawn from an unknown distribution $\mathcal{D}$ and want to learn the best value of the graph parameter ρ, that is one that minimizes loss $l_{A(G(\rho),L,U)}$, in expectation over the data distribution $\mathcal{D}$. We show a divergence in the weighted and unweighted graph learning problems. We analyze and provide asymptotically tight bounds for the pseudodimension of the set of loss functions parameterized by the graph family parameter ρ, i.e. $\mathcal{H}_\rho = \{l_{A(G(\rho),L,U)} \mid \rho \in \mathcal{P}\}$. For learning the unweighted threshold graphs, the pseudodimension is $O(\log n)$ which implies existence of an efficient algorithm with generalization guarantees in this setting. However, the pseudodimension is shown to be $\Omega(n)$ for the weighted graph setting, and therefore smoothness assumptions are necessary for learning over the algorithm family. Both these bounds are shown to be tight up to constant factors.

We also establish uniform convergence guarantees. For the unweighted graph setting, our pseudodimension bounds are sufficient for uniform convergence. We resort to bounding the Rademacher complexity in the weighted graph setting which allows us to prove distribution dependent generalization guarantees, that hold under distributional niceness assumptions of Section 4.1 (unlike pseudodimension which gives generalization guarantees that are worst-case over the distribution). The online learning results above only work for smoothed but adversarial instances, while the pseudodimension-based distributional learning sample complexity results work for any type (no smoothness needed) of independent and identically distributed instances. So these results are not superseded by the online learning results and provide new upper and lower bounds for the problem.

Pseudodimension bounds. We provide an upper bound on the pseudodimension of the set of loss functions for unweighted graphs $\mathcal{H}_r = \{l_{A(G(r),L,U)} \mid 0 \le r < \infty\}$, where $G(r)$ is specified by Definition 1a. Our bounds hold for general quadratic objectives (Table 1) and imply learnability with polynomially many samples. For the upper bound, we show that given any m instances we can partition the real line into $O(mn^2)$ intervals such that all values of r behave identically for all instances within any fixed interval. We also show an asymptotically tight lower bound on the pseudodimension of $\mathcal{H}_r$, by presenting a collection of graph thresholds and precisely designed labeling instances which are shattered by the thresholds. For full proof details see Appendix D.

Theorem 10. *The pseudo-dimension of $\mathcal{H}_r$ is $\Theta(\log n)$, where n is number of graph nodes.*

Proof Sketch. Upper bound. As r is increased from 0 to infinity, at most $\binom{n}{2} + 1$ distinct graphs may be obtained. Thus given set $\mathcal{S}$ of m instances $(A^{(i)}, L^{(i)})$, we can partition the real line into $O(mn^2)$ intervals such that all values of r behave identically for all instances within any fixed interval. The loss function is a piecewise constant with only $O(mn^2)$ pieces. Each piece can have a witness above or below it as r is varied for the corresponding interval, and so the binary labeling of $\mathcal{S}$ is fixed in that interval. The pseudo-dimension m satisfies $2^m \le O(mn^2)$ and is therefore $O(\log n)$.
Lower bound: We have three labeled nodes, a_1 with label 0 and b_1, b_2 labeled 1, and $n' = O(n)$ unlabeled nodes $U = \{u_1, \ldots, u_{n'}\}$. We can show that given a sequence $\{r_1, \ldots, r_{n'}\}$ of values of r, it is possible to construct an instance with suitable true labels of U such that the loss as a function of r oscillates above and below some witness as r moves along the sequence of intervals $(r_i, r_{i+1})_{i \ge 0}$. At the initial threshold r_0, all unlabeled points have a single incident edge, connecting to a_1, so all predicted labels are 0. As the threshold is increased to r_i, (the distances are set so that) u_i gets connected to both nodes with label 1 and its predicted label changes to 1. If the sequence of nodes u_i is alternately labeled, the loss decreases and increases alternately as all the predicted labels turn to 1 as r is increased to $r_{n'}$. This oscillation between a high and a low value can be achieved for any



(a) MNIST (b) Omniglot (c) CIFAR-10

Figure 2: Multiple instances of the same problem, loss as a function of σ.

subsequence of distances $r_1, \ldots, r_{n'}$, and a witness may be set as a loss value between the oscillation limits. By precisely choosing the subsequences so that the oscillations align with the bit flips in the binary digit sequence, we can construct m instances which satisfy the 2^m shattering constraints. $\square$

For learning weighted graphs $G(\sigma)$, we can show a $\Theta(n)$ bound on the pseudodimension of the set of loss functions $\mathcal{H}_\sigma = \{l_{A(G(\sigma),L,U)} \mid 0 \leq \sigma < \infty\}$. The lower bound consists of inductively constructed graphs with carefully set edges in a precisely designed sequence (Appendix D).

Theorem 11. *The pseudo-dimension of $\mathcal{H}_\sigma$ is $\Theta(n)$.*

Uniform convergence. Our results above implies a uniform convergence guarantee for the offline distributional setting, for both weighted and unweighted graph families. For the unweighted case, we can use the pseudodimension bounds above, and for the weighted case we use dispersion guarantees from section 4.1. For either case it suffices to bound the empirical Rademacher complexity. We will need the following theorem (slightly rephrased) from Balcan et al. [2018b].

Theorem 12. *[Balcan et al., 2018b] Let $\mathcal{F} = \{f_\rho : \mathcal{X} \to [0,1], \rho \in \mathcal{C} \subset \mathbb{R}^d\}$ be a parametereized family of functions, where $\mathcal{C}$ lies in a ball of radius R. For any set $\mathcal{S} = \{x_i, \ldots, x_T\} \subseteq \mathcal{X}$, suppose the functions $u_{x_i}(\rho) = f_\rho(x_i)$ for $i \in [T]$ are piecewise L-Lipschitz and β-dispersed. Then $\hat{R}(\mathcal{F}, \mathcal{S}) \leq O(\min\{\sqrt{(d/T)\log RT} + LT^{-\beta}, \sqrt{Pdim(\mathcal{F})/T}\})$.*

Now, using classic results from learning theory, we conclude that ERM has good generalization.

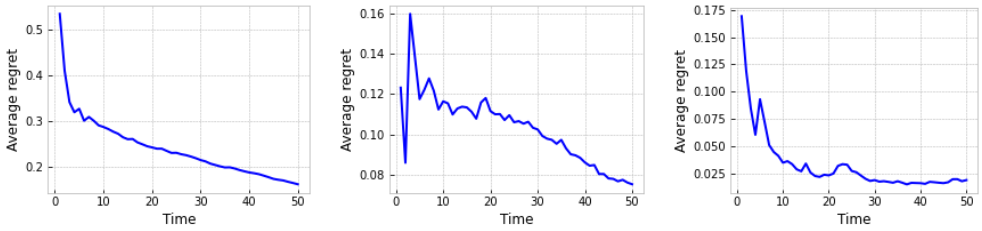
Theorem 13. *For both weighted and unweighted graph $w(u,v)$ defined above, with probability at least $1-\delta$, the average loss on any sample $x_1, \ldots, x_T \sim D^T$, the loss suffered w.r.t. to any parameter $\rho \in \mathbb{R}^d$ satisfies $|\frac{1}{T}\sum_{i=1}^T l_\rho(x_i) - \mathbb{E}_{x\sim D} l_\rho(x)| \leq O\left(\sqrt{\frac{d \log T \log 1/\delta}{T}}\right)$.*

6 Experiments

In this section we evaluate the performance of our learning procedures when finding application-specific semi-supervised learning algorithms (i.e. graph parameters). Our experiments[3] demonstrate that the best parameter for different applications varies greatly, and that the techniques presented in this paper can lead to large gains. We look at image classification based on standard pixel embedding.

Setup: We consider the task of semi-supervised classfication on image datasets. We restrict our attention to binary classification and pick two classes (labels 0 or 1) for each dataset. We then draw random subsets of the dataset (with class restriction) of size $n = 100$ and randomly select L examples for labeling. For any data subset S, we measure distance between any pairs of images using the L_2 distance between their pixel intensities. We would like to determine data-specific good values for σ, when predictions are made by optimizing the harmonic objective (Table 1). We use three popular benchmark datasets — MNIST [LeCun et al., 1998], Omniglot [Lake et al., 2015] and CIFAR-10 [Szegedy et al., 2015]. We generate a random semi-supervised learning instance from the data by sampling 100 random examples and further sampling L random examples from the subset for labeling. $L = 10$ for MNIST, while $L = 20$ for Omniglot and CIFAR-10.

[3]Code: `https://drive.google.com/drive/folders/1IqIw2Mp23W35UUwlz1hy24Eba5sPpVH_`



(a) MNIST (b) Omniglot (c) CIFAR-10

Figure 3: Average regret vs. T for online learning of parameter σ

Results and discussion: For the MNIST dataset we get optimal parameters with near-perfect classification even with small values of L, while for other datasets the error of the optimal parameter is over 0.1 even with larger values of L, indicating differences in the inherent difficulties of the classification tasks (like label noise and how well separated the classes are). We examine the full variation of performance of graph-based semi-supervised learning for all possible graphs $G(\sigma)$ for $\sigma \in [0, 10]$. The losses are piecewise constant and can have large discontinuities in some cases. The optimal parameter values vary with the dataset, but we observe at least 10%, and up to 80%, absolute gaps in performance between optimal and suboptimal values within the same dataset.

Another interesting observation is the variation of optima across data subsets, indicating transductively optimal parameters may not generalize well. We plot the variation of loss with parameter σ for several subsets of the same size $N = 100$ for MNIST and Omniglot datasets in Figure 2. In MNIST we have two optimal ranges in most subsets but only one shared optimum (around $\sigma = 2$) across different subsets. This indicates that local search based techniques that estimate the optimal parameter values on a given data instance may lead to very poor performance on unseen instances. The CIFAR-10 example further shows that the optimal algorithm may not be easy to empirically discern.

We also implement our online algorithms and compute the average regret for finding the optimal graph parameter σ for the different datasets. To obtain smooth curves we plot the average over 50 iterations for learning from 50 problem instances each ($T = 50$, Figure 3). We observe fast convergence to the optimal parameter regret for all the datasets considered. The starting part of these curves ($T = 0$) indicates regret for randomly setting the graph parameters, averaged over iterations, which is strongly outperformed by our learning algorithms as they learn from problem instances.

7 Ethics and broader impact

This work takes a step in making semi-supervised learning techniques domain independent and more practically effective. The resulting automation reduces dependence on human labelers and domain experts needed in current approaches. Dataset bias and ethics of applications will need to be individually considered when applying our approach to real world problems.

8 Acknowledgments

This material is based on work supported by the National Science Foundation under grants CCF-1535967, CCF-1910321, IIS-1618714, IIS-1901403, and SES-1919453; the Defense Advanced Research Projects Agency under cooperative agreement HR00112020003; an AWS Machine Learning Research Award; an Amazon Research Award; a Bloomberg Research Grant; a Microsoft Research Faculty Fellowship. The views expressed in this work do not necessarily reflect the position or the policy of the Government and no official endorsement should be inferred.

References

Doug Altner and Ozlem Ergun. Rapidly solving an online sequence of maximum flow problems. *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (L. Michel, ed.),(Spain)*, pages 283–287, 2008.

Maria-Florina Balcan. Data-Driven Algorithm Design. In Tim Roughgarden, editor, *Beyond Worst Case Analysis of Algorithms*. Cambridge University Press, 2020.

Maria-Florina Balcan, Avrim Blum, Patrick Pakyan Choi, John Lafferty, Brian Pantano, Mugizi Robert Rwebangira, and Xiaojin Zhu. Person identification in webcam images: An application of semi-supervised learning. In *ICML 2005 Workshop on Learning with Partially Classified Training Data*, volume 2, page 6, 2005.

Maria-Florina Balcan, Vaishnavh Nagarajan, Ellen Vitercik, and Colin White. Learning-theoretic foundations of algorithm configuration for combinatorial partitioning problems. In *Conference on Learning Theory*, pages 213–274. PMLR, 2017.

Maria-Florina Balcan, Travis Dick, Tuomas Sandholm, and Ellen Vitercik. Learning to branch. In *International conference on machine learning*, pages 344–353. PMLR, 2018a.

Maria-Florina Balcan, Travis Dick, and Ellen Vitercik. Dispersion for data-driven algorithm design, online learning, and private optimization. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 603–614. IEEE, 2018b.

Maria-Florina Balcan, Travis Dick, and Manuel Lang. Learning to link. In *International Conference on Learning Representations*, 2019.

Maria-Florina Balcan, Avrim Blum, Dravyansh Sharma, and Hongyang Zhang. On the power of abstention and data-driven decision making for adversarial robustness. *arXiv preprint arXiv:2010.06154*, 2020a.

Maria-Florina Balcan, Travis Dick, and Wesley Pegden. Semi-bandit optimization in the dispersed setting. In *Conference on Uncertainty in Artificial Intelligence*, pages 909–918. PMLR, 2020b.

Maria-Florina Balcan, Travis Dick, and Dravyansh Sharma. Learning piecewise Lipschitz functions in changing environments. In *International Conference on Artificial Intelligence and Statistics*, pages 3567–3577. PMLR, 2020c.

Maria-Florina Balcan, Tuomas Sandholm, and Ellen Vitercik. Learning to optimize computational resources: Frugal training with generalization guarantees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3227–3234, 2020d.

Maria-Florina Balcan, Tuomas Sandholm, and Ellen Vitercik. Refined bounds for algorithm configuration: The knife-edge of dual class approximability. In *International Conference on Machine Learning*, pages 580–590. PMLR, 2020e.

Maria-Florina F Balcan, Travis Dick, and Colin White. Data-driven clustering via parameterized lloyd's families. *Advances in Neural Information Processing Systems*, 31:10641–10651, 2018c.

Peter L Bartlett and Shahar Mendelson. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3(Nov):463–482, 2002.

Mikhail Belkin, Partha Niyogi, and Vikas Sindhwani. Manifold regularization: A geometric framework for learning from labeled and unlabeled examples. *Journal of machine learning research*, 7 (Nov):2399–2434, 2006.

Avrim Blum. Technical perspective: Algorithm selection as a learning problem. *Communications of the ACM*, 63(6):86–86, 2020.

Avrim Blum and Shuchi Chawla. Learning from labeled and unlabeled data using graph mincuts. In *ICML*, 2001.

Olivier Chapelle, Bernhard Schlkopf, and Alexander Zien. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010. ISBN 0262514125.

Matthew England and James H Davenport. The complexity of cylindrical algebraic decomposition with respect to polynomial degree. In *International Workshop on Computer Algebra in Scientific Computing*, pages 172–192. Springer, 2016.

Bryan R Gibson, Timothy T Rogers, and Xiaojin Zhu. Human semi-supervised learning. *Topics in cognitive science*, 5(1):132–172, 2013.

Rishi Gupta and Tim Roughgarden. A PAC approach to application-specific algorithm selection. In *Proceedings of the 7th Innovations in Theoretical Computer Science Conference*, 2016.

Rishi Gupta and Tim Roughgarden. A PAC approach to application-specific algorithm selection. *SIAM Journal on Computing*, 46(3):992–1017, 2017.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

David Pollard. *Convergence of stochastic processes*. Springer Science & Business Media, 2012.

Shai Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and trends in Machine Learning*, 4(2):107–194, 2011.

Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

Vikas Sindhwani, Partha Niyogi, and Mikhail Belkin. Beyond the point cloud: from transductive to semi-supervised learning. In *Proceedings of the 22nd international conference on Machine learning*, pages 824–831, 2005.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

Ellen Vitercik, Maria-Florina Balcan, and Tuomas Sandholm. Estimating approximate incentive compatibility. In *ACM Conference on Economics and Computation*, 2019.

Zhu Xiaojin and Ghahramani Zoubin. Learning from labeled and unlabeled data with label propagation. *Tech. Rep., Technical Report CMU-CALD-02–107, Carnegie Mellon University*, 2002.

Richard Zemel and Miguel Carreira-Perpiñán. Proximity graphs for clustering and manifold learning. *Advances in neural information processing systems*, 17:225–232, 2004.

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. *Advances in neural information processing systems*, 16(16):321–328, 2004.

Xiaojin Zhu and Andrew B Goldberg. Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, 3(1):1–130, 2009.

Xiaojin Zhu, Zoubin Ghahramani, and John D Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of the 20th International conference on Machine learning (ICML-03)*, pages 912–919, 2003.

Xiaojin Zhu, John Lafferty, and Ronald Rosenfeld. *Semi-supervised learning with graphs*. PhD thesis, Carnegie Mellon University, Language Technologies Institute, 2005.

Xiaojin Zhu, Timothy Rogers, Ruichen Qian, and Chuck Kalish. Humans perform semi-supervised classification too. In *AAAI*, volume 2007, pages 864–870, 2007.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] Remarked with results as applicable, e.g. Theorem 7 does not extend to mincut objective.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] Section 7.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] URL.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A] Can be reproduced on personal computers within reasonable time (16GB, 2.3 GHz Dual-Core).
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Appendix

A Definitions from previous work

We first note the definitions of some well-known useful learning theoretic complexity measures. Recall the definitions of pseudodimension and Rademacher complexity, well-known measures for hypothesis-space complexity in statistical learning theory. Bounding these quantities implies immediate bounds on learning error using classic learning theoretic results. In Section 5 we bound the pseudodimension and Rademacher complexity for the problems of learning unweighted and weighted graphs.

Definition 14. *Pseudo-dimension [Pollard, 2012]. Let $\mathcal{H}$ be a set of real valued functions from input space $\mathcal{X}$. We say that $C = (x_1, \ldots, x_m) \in \mathcal{X}^m$ is pseudo-shattered by $\mathcal{H}$ if there exists a vector $r = (r_1, \ldots, r_m) \in \mathbb{R}^m$ (called "witness") such that for all $b = (b_1, \ldots, b_m) \in \{\pm 1\}^m$ there exists $h_b \in \mathcal{H}$ such that sign$(h_b(x_i) - r_i) = b_i$. Pseudo-dimension of $\mathcal{H}$ is the cardinality of the largest set pseudo-shattered by $\mathcal{H}$.*

Definition 15. *Rademacher complexity [Bartlett and Mendelson, 2002]. Let $\mathcal{F} = \{f_\rho : \mathcal{X} \to [0,1], \rho \in \mathcal{C} \subset \mathbb{R}^d\}$ be a parameterized family of functions, and sample $\mathcal{S} = \{x_i, \ldots, x_T\} \subseteq \mathcal{X}$. The empirical Rademacher complexity of $\mathcal{F}$ with respect to $\mathcal{S}$ is defined as $\hat{R}(\mathcal{F}, \mathcal{S}) = \mathbb{E}_\sigma \left[\sup_{f \in \mathcal{F}} \frac{1}{T} \sum_{i=1}^{T} \sigma_i f(x_i)\right]$, where $\sigma_i \sim U(\{-1, 1\})$ are Rademacher variables.*

We will also need the definition of *dispersion* which, informally speaking, captures how amenable a non-Lipschitz function is to online learning. As noted in [Balcan et al., 2018b, 2020c], dispersion is necessary and sufficient for learning piecewise Lipschitz functions.

Definition 16. *Dispersion [Balcan et al., 2020b]. The sequence of random loss functions $l_1, \ldots, l_T$ is β-dispersed for the Lipschitz constant L if, for all T and for all $\epsilon \geq T^{-\beta}$, we have that, in expectation, at most $\tilde{O}(\epsilon T)$ functions (the soft-O notation suppresses dependence on quantities beside ϵ, T and β, as well as logarithmic terms) are not L-Lipschitz for any pair of points at distance ϵ in the domain $\mathcal{C}$. That is, for all T and for all $\epsilon \geq T^{-\beta}$,*

$$\mathbb{E}\left[\max_{\substack{\rho,\rho' \in \mathcal{C} \\ \|\rho-\rho'\|_2 \leq \epsilon}} \left|\{t \in [T] \mid l_t(\rho) - l_t(\rho') > L\,\|\rho - \rho'\|_2\}\right|\right] \leq \tilde{O}(\epsilon T).$$

B Motivation for data-driven design

Theorem 5. *Let $r_{\min}$ denote the smallest value of threshold r for which every unlabeled node of $G(r)$ is reachable from some labeled node, and $r_{\max}$ be the smallest value of threshold r for which $G(r)$ is the complete graph. There exists a data instance (L, U) such that for any $r_\zeta = \zeta r_{\min} + (1-\zeta) r_{\max}$ for $\zeta \in (0,1)$, there exists a set of labelings $\mathcal{U}$ of the unlabeled points such that for some $U_\zeta, \bar{U}_\zeta \in \mathcal{U}$, r_ζ minimizes $l_{A(G(r),L,U_\zeta)}$ but not $l_{A(G(r),L,\bar{U}_\zeta)}$.*

Proof. Note that for any $r < r_{\min}$, there is no graph similarity information for at least one node, and therefore all labels cannot be predicted. Also, the graph is unchanged for all $r \geq r_{\max}$. Therefore, $r \in [r_{\min}, r_{\max}]$ captures all graphs of interest on a given data instance.

Intuitively the statement claims that any threshold r (modulo the scaling factors for the data embedding) may be optimal or suboptimal for some data labeling for a given constructed instance. Therefore it is useful to consider several problem instances and learn the optimal value of r for the data distribution. We will present an example where an unlabeled point is closest to some labeled point of one class but closer to more points of another class on average. So for small thresholds it may be labeled as the first class and for larger thresholds as the second class.

Let $L = L_1 \cup L_2$ with $|L_1| < |L_2|$ and $d(u,v) = 0$ for $u, v \in L_i, i \in \{1,2\}$, $d(u,v) = 3r^*/2$ for $u \in L_i, v \in L_j, i \neq j$, where r^* is a positive real. Further let $U = \{a\}$ such that $d(a, u_i) = ir^*/2$ for each $u_i \in L_i$. It is straightforward to verify that the triangle inequality is satisfied. Further note that $r_{\min} = r^*/2$ and $r_{\max} = 3r^*/2$. Our set of labelings $\mathcal{U}$ will include one that labels a according to each class. Now we have two cases

1. $\zeta \in (0, \frac{1}{2})$: $r_{\min} \leq r < r^*$, $G(r_\zeta)$ connects a to L_1 but not L_2 and we have that the loss is minimized exactly for the labeling where a matches L_1.
2. $\zeta \in [\frac{1}{2}, 1)$: $r^* \leq r \leq r_{\max}$, $G(r_\zeta)$ connects a to both L_1 and L_2. But since $|L_1| < |L_2|$, we predict that the label of a matches that of L_2.

Finally we note that $d(u, v)$ may not be exactly zero when $u \neq v$ for a metric. This is easily fixed by making tiny perturbations to the labeled points, for any given r_ζ. □

The example presented above captures some essential challenges of our setting in the following sense. Firstly, we see that the loss function may be non-Lipschitz (as a function of the parameter r), which makes the optimization problem more challenging. More importantly, it highlights that graph similarity only approximately corresponds to label similarity, and how the accuracy of this correspondence is strongly influenced by the graph parameters. In this sense, it may not be possible to learn from a single instance, and considering a data-driven setting is crucial.

C Dispersion and Online learning

In this appendix we include details of proofs and algorithms from section 4.1.

C.1 Dispersion for threshold graphs

We will need the following simple lemma.

Lemma 17. *Let $\bar{l}(r) = l_{A(G(r),L,U)}$ be the loss function for graph $G(r)$ created using the threshold kernel $w(u, v) = \mathbb{I}[d(u, v) \leq r]$. Then $\bar{l}(r)$ is piecewise constant and any discontinuity occurs at $r^* = d(u, v)$ for some graph nodes u, v.*

Proof. This essentially follows from the observation that as r is increased, the graph gets a new edge only for some $r^* = d(u, v)$. Therefore no matter what the optimization algorithm is used to predict labels to minimize the loss, the loss is fixed given the graph, and has discontinuities potentially only when new edges are added. □

We are now ready to establish dispersion for the unweighted graph setting.

Theorem 6. *Let $l_1, \ldots, l_T : \mathbb{R} \to \mathbb{R}$ denote an independent sequence of losses as a function of parameter r, when the graph is created using a threshold kernel $w(u, v) = \mathbb{I}[d(u, v) \leq r]$ and labeled by applying any algorithm on the graph. If $d(u, v)$ follows a κ-bounded distribution for any u, v, the sequence is $\frac{1}{2}$-dispersed, and the regret of Algorithm 1 is $\tilde{O}(\sqrt{T})$.*

Proof. Assume a fixed but arbitrary ordering of nodes in each $V_t = L_t \cup U_t$ denoted by $V_t^{(i)}, i \in [n]$. Define $d_{i,j} = \{d(u, v) \mid u = V_t^{(i)}, v = V_t^{(j)}, t \in [T]\}$. Since $d_{i,j}$ is κ-bounded, the probability that it falls in any interval of length ϵ is $O(\kappa\epsilon)$. Since different problem instances are independent and using the fact that the VC dimension of intervals is 2, with probability at least $1 - \delta/D$, every interval of width ϵ contains at most $O\left(\kappa\epsilon T + \sqrt{T \log D/\delta}\right)$ discontinuities from each $d_{i,j}$ (using Lemma 17). Now a union bound over the failure modes for $d_{i,j}$ for different i, j gives $O\left(n^2\kappa\epsilon T + n^2\sqrt{T \log n/\delta}\right)$ discontinuities with probability at least $1-\delta$ for any ϵ-interval. Setting $\delta = 1/\sqrt{T}$, for each $\epsilon \geq 1/\sqrt{T}$ the maximum number of discontinuities in any ϵ-interval is at most $(1 - \delta)O\left(\kappa n^2\sqrt{T \log n\sqrt{T}}\right) + \delta T = \tilde{O}(\epsilon T)$, in expectation, proving $\frac{1}{2}$-dispersion. □

C.2 A general tool for analyzing dispersion

If the weights of the graph are given by a polynomial kernel $w(u, v) = (\tilde{d}(u, v) + \tilde{\alpha})^d$, we can apply the general tool developed by Balcan et al. [2020b] to learn $\tilde{\alpha}$, which we summarize below.

1. Bound the probability density of the random set of discontinuities of the loss functions.

2. Use a VC-dimension based uniform convergence argument to transform this into a bound on the dispersion of the loss functions.

Formally, we have the following theorems from Balcan et al. [2020b], which show how to use this technique when the discontinuities are roots of a random polynomial.

Theorem 18 (Balcan et al. [2020b]). *Consider a random degree d polynomial ϕ with leading coefficient 1 and subsequent coefficients which are real of absolute value at most R, whose joint density is at most κ. There is an absolute constant K depending only on d and R such that every interval I of length $\leq \epsilon$ satisfies Pr(ϕ has a root in I) $\leq \kappa\epsilon/K$.*

Theorem 19 (Balcan et al. [2020b]). *Let $l_1, \ldots, l_T : \mathbb{R} \to \mathbb{R}$ be independent piecewise L-Lipschitz functions, each having at most K discontinuities. Let $D(T, \epsilon, \rho) = |\{1 \leq t \leq T \mid l_t$ is not L-Lipschitz on $[\rho - \epsilon, \rho + \epsilon]\}|$ be the number of functions that are not L-Lipschitz on the ball $[\rho - \epsilon, \rho + \epsilon]$. Then we have $E[\max_{\rho \in \mathbb{R}} D(T, \epsilon, \rho)] \leq \max_{\rho \in \mathbb{R}} E[D(T, \epsilon, \rho)] + O(\sqrt{T \log(TK)})$.*

We will now use Theorems 18 and 19 to establish dispersion in our setting. We first need a simple lemma about κ-bounded distributions. We remark that similar properties have been proved in Balcan et al. [2018b, 2020b], in other problem contexts. Specifically, Balcan et al. [2018b] show the lemma for a ratio of random variables, $Z = X/Y$, and Balcan et al. [2020b] establish it for the sum $Z = X + Y$ but for independent variables X, Y.

Lemma 20. *Suppose X and Y are real-valued random variables taking values in $[m, m + M]$ and $[m', m' + M']$ for some $m, m', M, M' \in \mathbb{R}^+$ and suppose that their joint distribution is κ-bounded. Then,*

(i) $Z = X + Y$ is drawn from a $K_1\kappa$-bounded distribution, where $K_1 \leq \min\{M, M'\}$.

(ii) $Z = XY$ is drawn from a $K_2\kappa$-bounded distribution, where $K_2 \leq \min\{M/m, M'/m'\}$.

Proof. Let $f_{X,Y}(x, y)$ denote the joint density of X, Y.

(i) The case where X, Y are independent has been studied (Lemma 25 in Balcan et al. [2020b]), the following is slightly more involved. The cumulative density function for Z is given by

$$\begin{aligned} F_Z(z) &= \Pr(Z \leq z) = \Pr(X + Y \leq z) = \Pr(X \leq z - Y) \\ &= \int_{m'}^{m'+M'} \int_{m}^{z-y} f_{X,Y}(x, y) dx dy. \end{aligned}$$

The density function for Z can be obtained using Leibniz's rule as

$$\begin{aligned} f_Z(z) = \frac{d}{dz} F_Z(z) &= \frac{d}{dz} \int_{m'}^{m'+M'} \int_{m}^{z-y} f_{X,Y}(x, y) dx dy \\ &= \int_{m'}^{m'+M'} \left(\frac{d}{dz} \int_{m}^{z-y} f_{X,Y}(x, y) dx \right) dy \\ &= \int_{m'}^{m'+M'} f_{X,Y}(z - y, y) dy \\ &\leq M'\kappa. \end{aligned}$$

A symmetric argument shows that $f_Z(z) \leq M\kappa$, together with above this completes the proof.

(ii) The cumulative density function for Z is given by

$$\begin{aligned} F_Z(z) &= \Pr(Z \leq z) = \Pr(XY \leq z) = \Pr(X \leq z/Y) \\ &= \int_{m'}^{m'+M'} \int_{m}^{z/y} f_{X,Y}(x, y) dx dy. \end{aligned}$$

The density function for Z can be obtained using Leibniz's rule as

$$\begin{aligned} f_Z(z) = \frac{d}{dz} F_Z(z) &= \frac{d}{dz} \int_{m'}^{m'+M'} \int_{m}^{z/y} f_{X,Y}(x,y) dx dy \\ &= \int_{m'}^{m'+M'} \left(\frac{d}{dz} \int_{m}^{z/y} f_{X,Y}(x,y) dx \right) dy \\ &= \int_{m'}^{m'+M'} \frac{1}{y} f_{X,Y}(z/y, y) dy \\ &\leq \int_{m'}^{m'+M'} \frac{1}{m'} f_{X,Y}(z/y, y) dy \\ &\leq \frac{M'}{m'} \kappa. \end{aligned}$$

Similarly we can show that $f_Z(z) \leq M\kappa/m$, together with above this completes the proof.

□

Theorem 7. *Let $l_1, \ldots, l_T : \mathbb{R} \to \mathbb{R}$ denote an independent sequence of losses as a function of $\tilde{\alpha}$, for graph with edges $w(u,v) = (\tilde{d}(u,v) + \tilde{\alpha})^d$ labeled by optimizing the quadratic objective $\sum_{u,v} w(u,v)(f(u) - f(v))^2$. If $\tilde{d}(u,v)$ follows a κ-bounded distribution with a closed and bounded support, the sequence is $\frac{1}{2}$-dispersed, and the regret of Algorithm 1 may be upper bounded by $\tilde{O}(\sqrt{T})$.*

Proof. $w(u,v)$ is a polynomial in $\tilde{\alpha}$ of degree d with coefficient of $\tilde{\alpha}^i$ given by $c_i = D_{d,i} \tilde{d}(u,v)^{E_{d,i}}$ for $i \in [d]$. Since the support of $\tilde{d}(u,v)$ is closed and bounded, we have $m \leq \tilde{d}(u,v) \leq M$ with probability 1 for some $M > 1, m > 0$ (since $\tilde{d}(u,v)$ is a metric, $\tilde{d}(u,v) > 0$ for $u \neq v$).

To apply Theorem 18, we note that we have an upper bound on the coefficients, $R < (dM)^d$. Moreover, if $f(x)$ denotes the probability density of $d(u,v)$ and $F(x)$ its cumulative density,

$$\begin{aligned} \Pr(c_i \leq x_i) &= \Pr\left(D_{d,i} \tilde{d}(u,v)^{E_{d,i}} \leq x_i \right) \\ &= \Pr\left(\tilde{d}(u,v) \leq \left(\frac{x_i}{D_{d,i}} \right)^{1/E_{d,i}} \right) = F\left(\left(\frac{x_i}{D_{d,i}} \right)^{1/E_{d,i}} \right). \end{aligned}$$

Thus,

$$\Pr(c_i \leq x_i \text{ for each } i \in [d]) = F\left(\min_i \left(\frac{x_i}{D_{d,i}} \right)^{1/E_{d,i}} \right).$$

The joint density of the coefficients is therefore $K\kappa$-bounded where K only depends on d, m. ($K \leq \max_i {D_{d,i}}^{-1/E_{d,i}} m^{-1+1/E_{d,i}}$).

Consider the harmonic solution of the quadratic objective Zhu et al. [2003] which is given by $f_U = (D_{UU} - W_{UU})^{-1} W_{UL} f_L$. For any $u \in U$, $f(u) = 1/2$ is a polynomial equation in $\tilde{\alpha}$ with degree at most nd. The coefficients of these polynomials are formed by multiplying sets of weights $w(u,v)$ of size up to n and adding the products, and are also bounded density on a bounded support (using above observation in conjunction with Lemma 20). The dispersion result now follows by an application of Theorems 18 and 19. The regret bound is implied by results from Balcan et al. [2018b, 2020c]. □

C.3 Dispersion for roots of exponential polynomials

In this section we will extend the applicability of the dispersion analysis technique from Appendix C.2 to exponential polynomials, i.e. functions of the form $\phi(x) = \sum_{i=1}^{n} a_i e^{b_i x}$. We will now extend the analysis to obtain similar results when using the exponential kernel $w(u,v) = e^{-||u-v||^2/\sigma^2}$. The results of Balcan et al. [2020b] no longer directly apply as the points of discontinuity are no longer roots of polynomials. To this end, we extend and generalize arguments from Balcan et al. [2020b] below. We need to generalize Theorem 18 to exponential polynomials below.

Theorem 21. *Let $\phi(x) = \sum_{i=1}^{n} a_i e^{b_i x}$ be a random function, such that coefficients a_i are real and of magnitude at most R, and distributed with joint density at most κ. Then for any interval I of width at most ϵ, P(ϕ has a zero in I)$\leq \tilde{O}(\epsilon)$ (dependence on b_i, n, κ, R suppressed).*

Proof. For $n = 1$ there are no roots, so assume $n > 1$. Suppose ρ is a root of $\phi(x)$. Then $\mathbf{a} = (a_1, \ldots, a_n)$ is orthogonal to $\varrho(\rho) = (e^{b_1\rho}, \ldots, e^{b_n\rho})$ in $\mathbb{R}^n$. For a fixed ρ, the set S_ρ of coefficients $\mathbf{a}$ for which ρ is a root of $\phi(y)$ lie along an $n-1$ dimensional linear subspace of $\mathbb{R}^n$. Now ϕ has a root in any interval I of length ϵ, exactly when the coefficients lie on S_ρ for some $\rho \in I$. The desired probability is therefore upper bounded by $\max_\rho \textsc{Vol}(\cup S_y \mid y \in [\rho-\epsilon, \rho+\epsilon])/\textsc{Vol}(S_y \mid y \in \mathbb{R})$ which we will show to be $\tilde{O}(\epsilon)$. The key idea is that if $|\rho - \rho'| < \epsilon$, then $\varrho(\rho)$ and $\varrho(\rho')$ are within a small angle $\theta_{\rho,\rho'} = \tilde{O}(\epsilon)$ for small ϵ (the probability bound is vacuous for large ϵ). But any point in S_ρ is at most $\tilde{O}(\theta_{\rho,\rho'})$ from a point in $S_{\rho'}$, which implies the desired bound (similar arguments to Theorem 18).

We will now flesh out the above sketch. Indeed,

$$\begin{aligned}
\sin\theta_{\rho,\rho'} &= \sqrt{1 - \frac{(\langle \varrho(\rho), \varrho(\rho')\rangle)^2}{\|\varrho(\rho)\|\,\|\varrho(\rho')\|}} \\
&= \sqrt{1 - \frac{\left(\sum_i e^{b_i\rho} e^{b_i\rho'}\right)^2}{\sum_i e^{2b_i\rho} \sum_i e^{2b_i\rho'}}} \\
&= \sqrt{\frac{\sum_{i\neq j} e^{2(b_i\rho + b_j\rho')} - e^{(b_i+b_j)(\rho+\rho')}}{\sum_i e^{2b_i\rho} \sum_i e^{2b_i\rho'}}}.
\end{aligned}$$

Now, for $\rho' = \rho + \varepsilon$, $|\varepsilon| < \epsilon$,

$$\begin{aligned}
\sin\theta_{\rho,\rho'} &= \sqrt{\frac{\sum_{i\neq j} e^{2(b_i\rho + b_j\rho + b_j\varepsilon)} - e^{(b_i+b_j)(2\rho+\varepsilon)}}{\sum_i e^{2b_i\rho} \sum_i e^{2b_i\rho'}}} \\
&= \sqrt{\frac{\sum_{i\neq j} e^{2\rho(b_i+b_j)}\left(e^{2b_j\varepsilon} - e^{(b_i+b_j)\varepsilon}\right)}{\sum_i e^{2b_i\rho} \sum_i e^{2b_i\rho'}}}.
\end{aligned}$$

Using the Taylor's series approximation for $e^{2b_j\varepsilon}$ and $e^{(b_i+b_j)\varepsilon}$, we note that the largest terms that survive are quadratic in ε. $\sin\theta_{\rho,\rho'}$, and therefore also $\theta_{\rho,\rho'}$, is $\tilde{O}(\epsilon)$.

Next it is easy to show that any point in S_ρ is at most $\tilde{O}(\theta_{\rho,\rho'})$ from a point in $S_{\rho'}$. For $n = 2$, S_ρ and $S_{\rho'}$ are along lines orthogonal to ρ and ρ' and are thus themselves at an angle $\theta_{\rho,\rho'}$. Since we further assume that the coefficients are bounded by R, any point on S_ρ is within $O(R\theta_{\rho,\rho'}) = \tilde{O}(\theta_{\rho,\rho'})$ of the nearest point in $S_{\rho'}$. For $n > 2$, consider the 3-space spanned by ρ, ρ' and an arbitrary $\varsigma \in S_\rho$. S_ρ and $S_{\rho'}$ are along 2-planes in this space with normal vectors ρ, ρ' respectively. Again it is straightforward to see that the nearest point in the projection of $S_{\rho'}$ to ς is $\tilde{O}(\theta_{\rho,\rho'})$.

The remaining proof is identical to that of Theorem 18 (see Theorem 18 of Balcan et al. [2020b]), and is omitted for brevity.

□

We will also need the following lemma for the second step noted above, i.e. obtain a result similar to Theorem 19 for exponential polynomials.

Lemma 22. *The equation $\sum_{i=1}^{n} a_i e^{b_i x} = 0$ where $a_i, b_i \in \mathbb{R}$ has at most $n-1$ distinct solutions $x \in \mathbb{R}$.*

Proof. We will use induction on n. It is easy to verify that there is no solution for $n = 1$. We assume the statement holds for all $1 \leq n \leq N$. Consider the equation $\phi_{N+1}(x) = \sum_{i=1}^{N+1} a_i e^{b_i x} = 0$.

WLOG $a_1 \neq 0$ and we can write

$$\phi_{N+1}(x) = \sum_{i=1}^{N+1} a_i e^{b_i x} = a_1 e^{b_1 x} \left(1 + \sum_{i=2}^{N+1} \frac{a_i}{a_1} e^{(b_i - b_1)x} \right) = a_1 e^{b_1 x} \left(1 + g(x)\right).$$

By our induction hypothesis, $g'(0) = 0$ has at most $N-1$ solutions, and so $(1+g(x))'$ has at most $N-1$ roots. By Rolle's theorem, $(1+g(x))$ has at most N roots, and therefore $\phi_{N+1}(x) = 0$ has at most N solutions. □

Lemma 22 implies that Theorem 19 may be applied. The number of discontinuities may be exponentially high in this case. Indeed solving the quadratic objective can result in an exponential equation of the form in Lemma 22 with $O(|U|^n)$ terms.

C.4 Learning several metrics simultaneously

We start by getting a couple useful definitions out of the way.

Definition 23 (Homogeneous algebraic hypersurface)**.** *An algebraic hypersurface is an algebraic variety (a system of polynomial equations) that may be defined by a single implicit equation of the form $p(x_1, \ldots, x_n) = 0$, where p is a multivariate polynomial. The degree d of the algebraic hypersurface is the total degree of the polynomial p. We say that the algebraic hypersurface is homogeneous if p is a homogeneous polynomial, i.e. $p(\lambda x_1, \ldots, \lambda x_m) = \lambda^d p(x_1, \ldots, x_n)$.*

In the following we will refer to homogeneous algebraic hypersurfaces as simply algebraic hypersurfaces. We will also need the standard definition of set shattering, which we restate in our context as follows.

Definition 24 (Hitting and Shattering)**.** *Let $\mathcal{C}$ denote a set of curves in $\mathbb{R}^p$. We say that a subset of $\mathcal{C}$ is hit by a curve s if the subset is exactly the set of curves in $\mathcal{C}$ which intersect the curve s. A collection of curves $\mathcal{S}$ shatters the set $\mathcal{C}$ if for each subset C of $\mathcal{C}$, there is some element s of $\mathcal{S}$ such that s hits C.*

To extend our learning results to learning graphs built from several metrics, we will now state and prove a couple theorems involving algebraic hypersurfaces. Our results generalize significantly the techniques from Balcan et al. [2020b] by bringing in novel connections with algebraic geometry.

Theorem 3. *There is a constant k depending only on d and p such that axis-aligned line segments in $\mathbb{R}^p$ cannot shatter any collection of k algebraic hypersurfaces of degree at most d.*

Proof. Let $\mathcal{C}$ denote a collection of k algebraic hypersurfaces of degree at most d in $\mathbb{R}^p$. We say that a subset of $\mathcal{C}$ is *hit* by a line segment if the subset is exactly the set of curves in $\mathcal{C}$ which intersect the segment, and *hit* by a line if some segment of the line hits the subset. We seek to upper bound the number of subsets of $\mathcal{C}$ which may be hit by axis-aligned line segments. We will first consider shattering by line segments in a fixed axial direction x. We can easily extend this to axis-aligned segments by noting they may hit only p times as many subsets.

Let L_c be a line in the x direction. The subsets of $\mathcal{C}$ which may be hit by (segments along) L_c is determined by the pattern of intersections of L_c with hypersurfaces in $\mathcal{C}$. By Bezout's theorem, there are at most $kd+1$ distinct regions of L_c due to the intersections. Therefore at most $\binom{kd+1}{2}$ distinct subsets may be hit.

Define the equivalence relation $L_{c_1} \sim L_{c_2}$ if the same hypersurfaces in $\mathcal{C}$ intersect L_{c_1} and L_{c_2}, and in the same order (including with multiplicities). To determine these equivalence classes, we will project the hypersurfaces in $\mathcal{C}$ on to a hyperplane orthogonal to the x-direction. By the Tarski-Seidenberg-Łojasiewicz Theorem, we get a semi-algebraic collection $\mathcal{C}_x$, i.e. a set of polynomial equations and constraints in the projection space. Each cell of $\mathcal{C}_x$ corresponds to an equivalence class. Using well-known upper bounds for *cylindrical algebraic decomposition* (see for example England and Davenport [2016]), we get that the number of equivalence classes is at most $O\left((2d)^{2^p-1} k^{2^p-1} 2^{2^{p-1}}\right)$.

Putting it all together, the number of subsets hit by any axis aligned segment is at most

$$O\left(p\binom{kd+1}{2}(2d)^{2^p-1} k^{2^p-1} 2^{2^{p-1}}\right).$$

We are done as this is less than 2^k for fixed d and p and large enough k, and therefore all subsets may not be hit.

□

Theorem 4. *Let $l_1, \ldots, l_T : \mathbb{R}^p \to \mathbb{R}$ be independent piecewise L-Lipschitz functions, each having discontinuities specified by a collection of at most K algebraic hypersurfaces of bounded degree. Let L denote the set of axis-aligned paths between pairs of points in $\mathbb{R}^p$, and for each $s \in L$ define $D(T, s) = |\{1 \leq t \leq T \mid l_t \text{ has a discontinuity along } s\}|$. Then we have $\mathbb{E}[\sup_{s \in L} D(T, s)] \leq \sup_{s \in L} \mathbb{E}[D(T, s)] + O(\sqrt{T \log(TK)})$.*

Proof. The proof is similar to that of Theorem 19 (see Balcan et al. [2020b]). The main difference is that instead of relating the number of ways intervals can label vectors of discontinuity points to the VC-dimension of intervals, we instead relate the number of ways line segments can label vectors of K algebraic hypersurfaces of degree d to the VC-dimension of line segments (when labeling algebraic hypersurfaces), which from Theorem 3 is constant. To verify dispersion, we need a uniform-convergence bound on the number of Lipschitz failures between the worst pair of points α, α' at distance $\leq \epsilon$, but the definition allows us to bound the worst rate of discontinuties along any path between α, α' of our choice. We can bound the VC dimension of axis aligned segments against bounded-degree algebraic hypersurfaces, which will allow us to establish dispersion by considering piecewise axis-aligned paths between points α and α'.

Let $\mathcal{C}$ denote the set of all algebraic hypersurfaces of degree d. For simplicity, we assume that every function has its discontinuities specified by a collection of exactly K algebraic hypersurfaces. For each function l_t, let $\gamma^{(t)} \in \mathcal{C}^K$ denote the ordered tuple of algebraic hypersurfaces in $\mathcal{C}$ whose entries are the discontinuity locations of l_t. That is, l_t has discontinuities along $(\gamma_1^{(t)}, \ldots, \gamma_K^{(t)})$, but is otherwise L-Lispchitz.

For any axis aligned path s, define the function $f_s : \mathcal{C}^K \to \{0, 1\}$ by

$$f_s(\gamma) = \begin{cases} 1 & \text{if for some } i \in [K]\ \gamma_i \text{ intersects } s \\ 0 & \text{otherwise,} \end{cases}$$

where $\gamma = (\gamma_1, \ldots, \gamma_K) \in \mathcal{C}^K$. The sum $\sum_{t=1}^T f_s(\gamma^{(t)})$ counts the number of vectors $(\gamma_1^{(t)}, \ldots, \gamma_K^{(t)})$ that intersect s or, equivalently, the number of functions $l_1, \ldots, l_T$ that are not L-Lipschitz on s. We will apply VC-dimension uniform convergence arguments to the class $\mathcal{F} = \{f_s : \mathcal{C}^K \to \{0, 1\} \mid s \text{ is an axis-aligned path}\}$. In particular, we will show that for an independent set of vectors $(\gamma_1^{(t)}, \ldots, \gamma_K^{(t)})$, with high probability we have that $\frac{1}{T} \sum_{t=1}^T f_s(\gamma^{(t)})$ is close to $\mathbb{E}[\frac{1}{T} \sum_{t=1}^T f_s(\gamma^{(t)})]$ for all paths s. This uniform convergence argument will lead to the desired bounds.

Indeed, Theorem 3 implies that VC dimension of $\mathcal{F}$ is $O(\log K)$. Now standard VC-dimension uniform convergence arguments for the class $\mathcal{F}$ imply that with probability at least $1 - \delta$, for all $f_s \in \mathcal{F}$

$$\left| \frac{1}{T} \sum_{t=1}^T f_s(\gamma^{(t)}) - \mathbb{E}\left[\frac{1}{T} \sum_{t=1}^T f_s(\gamma^{(t)}) \right] \right| \leq O\left(\sqrt{\frac{\log(K/\delta)}{T}} \right), \text{ or}$$

$$\left| \sum_{t=1}^T f_s(\gamma^{(t)}) - \mathbb{E}\left[\sum_{t=1}^T f_s(\gamma^{(t)}) \right] \right| \leq O\left(\sqrt{T \log(K/\delta)} \right).$$

Now since $D(T, s) = \sum_{t=1}^T f_s(\gamma^{(t)})$, we have for all s and δ, with probability at least $1 - \delta$, $\sup_{s \in L} D(T, s) \leq \sup_{s \in L} \mathbb{E}[D(T, s)] + O(\sqrt{T \log(K/\delta)})$. Taking expectation and setting $\delta = 1/\sqrt{T}$ completes the proof as it allows us to bound the expected discontinuities by $O(\sqrt{T})$ when the above high probability event fails. □

Theorem 4 above generalizes the second step of the dispersion tool from single parameter families to several hyperparameters, and uses Theorem 3 as a key ingredient. To complete the first step of in the multi-parameter setting, we can use a simple generalization of Theorem 18 by showing that few zeros are likely to occur on a piecewise axis-aligned path on whose pieces the zero sets of the multivariate polynomial is the zero set of a single-variable polynomial. Putting together we get Theorem 8.

Theorem 8. *Let $l_1, \ldots, l_T : \mathbb{R}^p \to \mathbb{R}$ denote an independent sequence of losses as a function of parameters $\rho_i, i \in [p]$, when the graph is created using a polynomial kernel $w(u,v) = (\sum_{i=1}^{p-1} \rho_i \tilde{d}(u,v) + \rho_p)^d$ and labeled by optimizing the quadratic objective $\sum_{u,v} w(u,v)(f(u) - f(v))^2$. If $\tilde{d}(u,v)$ follows a κ-bounded distribution with a closed and bounded support, the sequence is $\frac{1}{2}$-dispersed, and the regret of Algorithm 1 may be upper bounded by $\tilde{O}(\sqrt{T})$.*

Proof. Notice that $w(u,v)$ is a homogeneous polynomial in $\rho = (\rho_i, i \in [p])$. Further, the solutions of the quadratic objective subject to $f(u) = 1/2$ for some u are also homogeneous polynomial equations, of degree nd. Now to show dispersion, consider an axis-aligned path between any two parameter vectors ρ, ρ' such that $\|\rho - \rho'\| < \epsilon$ (notice that the definition of dispersion allows us to use any path between ρ, ρ' for counting discontinuities). To compute the expected number of non-Lipchitzness in along this path, notice that for any fixed segment of this path, all but one variable are constant and the discontinuities are the zeros of single variable polynomial with bounded-density random coefficients, and that Theorem 18 applies. Summing along these paths we get at most $\tilde{O}(p\epsilon)$ discontinuities in expectation for any $\|\rho - \rho'\| < \epsilon$. Theorem 4 now completes the proof of dispersion in this case. □

C.5 Semi-bandit efficient algorithms

In this appendix we present details of the efficient algorithms for computing the semi-bandit feedback sets in Algorithm 2. For unweighted graphs, we only have a polynomial number $O(n^2)$ of feedback sets and the feedback set for a given ρ_t is readily computed by looking up a sorted list of distances $d(u,v)_{u,v \in L_i \cup U_i}$. For the weighted graph setting, we need non-trivial algorithms.

C.5.1 Min-cut objective

First some notation for this section. We will use $G = (V, E)$ to denote an undirected graph with V as the set of nodes and $E \subseteq V \times V$ the weighted edges with capacity $d : E \to \mathbb{R}_{\geq 0}$. We are given special nodes $s, t \in V$ called *source* and *target* vertices. Recall the following definitions.

Definition 25. (s,t)-flows *An (s,t)-flow (or just flow if the source and target are clear from context) is a function $f : V \times V \to \mathbb{R}_{\geq 0}$ that satisfies the conservation constraint at every vertex v except possibly s and t given by $\sum_{(u,v) \in E} f(u,v) = \sum_{(v,u) \in E} f(v,u)$. The value of flow (also refered by just flow when clear from context) is the total flow out of s, $\sum_{u \in V} f(s,u) - \sum_{u \in V} f(u,s)$.*

Definition 26. (s,t)-cut *An (s,t)-cut (or just a cut if the source and target are clear from context) is a partition of V into S, T such that $s \in S, t \in T$. We will denote the set $\{(u,v) \in E \mid u \in S, v \in T\}$ of edges in the cut by ∂S or ∂T. The capacity of the cut is the total capacity of edges in the cut.*

For convenience we also define

Definition 27. *Path flow. An (s,t)-flow is a path flow along a path $p = (s = v_0, v_1, \ldots, v_n = t)$ if $f(u,w) > 0$ iff $(u,w) = (v_i, v_{i+1})$ for some $i \in [n-1]$.*

Definition 28. *Residual capacity graph. Given a set of path flows F, the residual capacity graph (or simply the residual graph) is the graph $G' = (V, E)$ with capacities given by $c'(e) = c(e) - \sum_{f \in F} f(e)$.*

We will list without proof some well-known facts about maximum flows and minimum cuts in a graph which will be useful in our arguments.

Fact.

1. *Let f be any feasible (s,t)-flow, and let (S,T) be any (s,t)-cut. The value of f is at most the capacity of (S,T). Moreover, the value of f equals the capacity of (S,T) if and only if f saturates every edge in the cut.*
2. *Max-flow min-cut theorem. The value of maximum (value of) (s,t)-flow equals the capacity of the minimum (s,t)-cut. It may be computed in $O(VE)$ time.*
3. *Flow Decomposition Theorem. Every feasible (s,t)-flow f can be written as a weighted sum of directed (s,t)-paths and directed cycles. Moreover, a directed edge (u,v) appears in at least one of these paths or cycles if and only if $f(u,v) > 0$, and the total number of paths and cycles is at most the number of edges in the network. It may be computed in $O(VE)$ time.*

Algorithm 3 DYNAMICMINCUT(G, σ_0, ϵ)

Input: Graph G with unlabeled nodes, query parameter σ_0, error tolerance ϵ.
Output: Piecewise constant interval containing σ_0.
Use a max-flow algorithm to compute max-flow and min-cut $\mathcal{C}$ for $G(\sigma)$, $\sigma_h = \sigma_0$.
Compute the flow decomposition of the max-flow, $\mathcal{F}$.
Let f_e be a unique *path flow* (i.e. along an st-path, Definition 27) through $e \in \mathcal{C}$.
Say e is *augmentable* if flow f_e can be increased by amount $w_e(\sigma) - w_e(\sigma_h)$ for some $\sigma > \sigma_h$. e acts as the bottleneck for increasing the flow f_e.
Initialize S to $\mathcal{C}$ (a set of saturated edges).
while All edges $e \in S$ are augmentable, **do**
    Increase flow in all f_e for $e \in S$ to keep e saturated.
    Find first saturating edge $e_1 \notin S$ for some $f_{e'}$ ($e' \in S$) and σ' to within ϵ.
    Reassociate flow through e_1, e' as f_{e_1}. $f_{e'}$ will now be along an alternate path in the residual capacities graph (Definition 28).
    Add e_1 to S.
    Set $\sigma_h = \sigma'$.
Similarly find the start of the interval σ_l by detecting saturation while reducing flows.
return $[\sigma_l, \sigma_h]$.

We now have the machinery to prove the correctness and analyze the time complexity of our Algorithm 3.

Theorem 9. *For the each objective in Table 1 and exponential kernel (Definition 1c), there exists an algorithm which outputs the interval containing σ in time $\tilde{O}(n^4)$.*

Proof. Mincut objective. First, we briefly recall the set up of the mincut objective. Let L_1 and L_2 denote the labeled points L of different classes. To obtain the labels for U, we seek the smallest cut $(V_1, V \setminus V_1)$ of G separating the nodes in L_1 and L_2. To frame as s, t-cut we can augment the data graph with nodes s, t, and add infinite capacity edges to nodes in L_1 and L_2 respectively. If $L_i \subseteq V_1$, label exactly the nodes in V_1 with label i. The loss function, $l(\sigma)$ gives the fraction of labels this procedure gets right for the unlabeled set U. We now discuss the correctness of Algorithm 3.

If the min-cut is the same for two values of σ, then so is prediction on each point and thus the loss function $l(\sigma)$. So we seek the smallest amount of change in σ so that the mincut changes. Our semi-bandit feedback set is given by the intervals for which the min-cut is fixed. Consider a fixed value of $\sigma = \sigma_0$ and the corresponding graph $G(\sigma_0)$. We can compute the max-flow on $G(\sigma_0)$, and simultaneously obtain a min-cut $(V_1, V \setminus V_1)$ in time $O(VE) = O(n^3)$. All the edges in ∂V_1 are saturated by the flow. Obtain the flow decomposition of the max-flow (again $O(VE) = O(n^3)$). For each $e_i \in \partial V_1$, let f_i be a path flow through e_i from the flow decomposition (cycle flows cannot saturate, or even pass through, e_i since it is on the min-cut). Note that the f_i are distinct due to the max-flow min-cut theorem. Now as σ is increased, we increment each f_i by the additional capacity in the corresponding edge e_i, until an edge e' in $E \setminus \partial V_1$ saturates (at a faster rate than the flow through it). This can be detected by expressing f_i as a function of σ for each f_i and computing the zero of an exponential polynomial capturing the change in residual capacity of any edge $e \notin \partial V_1$. Let f_j be one of the path flows through e'. We reassign this flow to e' (it will now increase with e' as its bottleneck) and find an alternate path avoiding this edge through non-saturated edges and e_j (if one exists) along which we send the new f_j. We now increment all the path flows as before keeping their bottleneck edges saturated. The procedure stops when we can no longer find an alternate path for some e_j. But this means we must have a new cut with the saturated edges, and therefore a new min-cut. This gives us a new critical value of σ, and the desired upper end for the feedback interval. Obtaining the lower end is possible by a symmetric procedure that decreases the path flows while keeping edges saturated.

We remark that our procedure differs from the well-known algorithms for obtaining min-cuts in a static graph. The greedy procedures for static graphs need directed edges (u, v) and (v, u) in the residual graph, and find paths through unsaturated edges through this graph to increase the flow, and cannot work with monotonically increasing path flows. We however start with a max flow and maintain the invariant that our flow equals some cut size throughout.

Finally note that each time we perform step 9 of the algorithm, a new saturated edge stays saturated for all further σ until the new cut is found. So we can do this at most $O(n^2)$ times. In each loop we need to obtain the saturation condition for $O(n)$ edges corresponding to one new path flow. Thus the entire procedure takes $O(n^3 K(n, \epsilon))$ time, where $K(n, \epsilon)$ is the complexity of solving an exponential equation $\phi(y) = \sum_{i=1}^{n} a_i y^{b_i} = 0$ to within error ϵ. For example, $K(n, \epsilon)$ is $O(n \log\log \frac{1}{\epsilon})$ for the Newton's method.

Other objectives. For continuous objectives, we seek a solution $f_u(\sigma) = \frac{1}{2}$ for some $u \in U$ closest to given σ_0. We can use gradient descent or Newton's method with σ_0 as the starting point. For the harmonic objective the gradient may be computed as in Appendix C.5.2. □

We remind the reader that a remarkable property of finding the min-cuts dynamically in our setting is an interesting "hybrid" combinatorial and continuous set-up, which may be of independent interest. A similar dynamic, but purely combinatorial, setting for recomputing flows efficiently online over a discrete graph sequence has been studied in Altner and Ergun [2008].

C.5.2 Quadratic objective

For completeness we describe how to compute the feedback set for the quadratic objective using Newton's method. Running time guarantees are noted in Theorem 9.

Algorithm 4 HARMONICFEEDBACKSET(G, σ_0, ϵ)

1: **Input:** Graph G with unlabeled nodes, query parameter σ_0, error tolerance ϵ.
2: **Output:** Piecewise constant interval containing σ_0.
3: Let $f_U = (D_{UU} - W_{UU})^{-1} W_{UL} f_L$ denote the harmonic objective minimizer, where $D_{ij} := \mathbb{I}[i = j] \sum_k W_{ik}$.
4: **for all** $u \in U$ **do**
5: Let $g_u(\sigma) = (f_u(\sigma) - \frac{1}{2})^2$.
6: $\sigma_1 = \sigma_0 - \frac{g_u(\sigma_0)}{g'_u(\sigma_0)}$, where $g'_u(\sigma)$ is given by

$$\begin{aligned}
\frac{\partial g_u}{\partial \sigma} &= 2\left(f_u(\sigma) - \frac{1}{2}\right)\frac{\partial f_u}{\partial \sigma}, \\
\frac{\partial f}{\partial \sigma} &= (I - P_{UU}^{-1})\left(\frac{\partial P_{UU}}{\partial \sigma} f_U - \frac{\partial P_{UL}}{\partial \sigma} f_L\right), \\
\frac{\partial P_{ij}}{\partial \sigma} &= \frac{\frac{\partial w(i,j)}{\partial \sigma} - P_{ij} \sum_{k \in L+U} \frac{\partial w(i,k)}{\partial \sigma}}{\sum_{k \in L+U} w(i,k)}, \\
\frac{\partial w(i,j)}{\partial \sigma} &= \frac{2 w(i,j) d(i,j)^2}{\sigma^3},
\end{aligned}$$

where $P = D^{-1} W$.
7: $n = 0$
8: **while** $|\sigma_{n+1} - \sigma_n| \geq \epsilon$ **do**
9: $n \leftarrow n + 1$
10: $\sigma_{n+1} = \sigma_n - \frac{g_u(\sigma_n)}{g'_u(\sigma_n)}$
11: $\sigma_u = \sigma_{n+1}$
12: $\sigma_l = \max_u\{\sigma_u \mid \sigma_u < \sigma_0\}$, $\sigma_h = \min_u\{\sigma_u \mid \sigma_u > \sigma_0\}$
13: **return** $[\sigma_l, \sigma_h]$.

D Distributional setting

In this appendix we include details of proofs and algorithms from section 5. Recall that we define the set of loss functions $\mathcal{H}_r = \{l_{A(G(r),L,U)} \mid 0 \leq r < \infty\}$, where $G(r)$ is the family of threshold graphs specified by Definition 1a, and $\mathcal{H}_\sigma = \{l_{A(G(\sigma),L,U)} \mid 0 \leq \sigma < \infty\}$, where $G(\sigma)$ is the family of exponential kernel graphs specified by Definition 1c. We show upper and lower bounds on the pseudodimension of these function classes below.

Theorem 10. *The pseudo-dimension of $\mathcal{H}_r$ is $\Theta(\log n)$, where n is number of graph nodes.*

Proof. There are at most $\binom{n}{2}$ distinct distances between pairs of data points. As r is increased from 0 to infinity, the graph changes only when r corresponds to one of these distances, and so at most $\binom{n}{2}+1$ distinct graphs may be obtained.
Thus given set $\mathcal{S}$ of m instances $(A^{(i)}, L^{(i)})$, we can partition the real line into $O(mn^2)$ intervals such that all values of r behave identically for all instances within any fixed interval. Since A and therefore its loss is deterministic once G is fixed, the loss function is a piecewise constant with only $O(n^2)$ pieces. Each piece can have a witness above or below it as r is varied for the corresponding interval, and so the binary labeling of $\mathcal{S}$ is fixed in that interval. The pseudo-dimension m satisfies $2^m \leq O(mn^2)$ and is therefore $O(\log n)$.

To establish the lower bound, we first prove the following useful statement which helps us construct general examples with desirable properties. In particular, the following lemma guarantees that given a sequence of values of r of size $O(n)$, it is possible to construct an instance S of partially labeled points such that the cost of the output of algorithm $A(G(r), L)$ on V as a function of r oscillates above and below some threshold as r moves along the sequence of intervals (r_i, r_{i+1}). Given this powerful guarantee, we can then pick appropriate sequences of r and generate a sample set of $\Omega(\log n)$ instances that correspond to cost functions that oscillate in a manner that helps us pick $\Omega(n)$ values of r that shatters the samples.

Lemma 29. *Given integer $n > 5$ and a sequence of n' r's such that $1 < r_1 < r_2 < \cdots < r_{n'} < 2$ and $n' \leq n-5$, there exists a real valued witness $w > 0$ and a labeling instance S of partially labeled n points, such that for $0 \leq i \leq n'/2 - 1$, $l_{A(G(r),L)} < w$ for $r \in (r_{2i}, r_{2i+1})$, and $l_{A(G(r),L)} > w$ for $r \in (r_{2i+1}, r_{2i+2})$ (where r_0 and $r_{n'+1}$ correspond to immediate left and right neighborhoods respectively of r_1 and $r_{n'}$).*

Proof. We first present a sketch of the construction. We will use binary labels a and b. We further have three points labeled a (namely a_1, a_2, a_3) and two points labeled b (say b_1, b_2). At some intial $r = r_0$, all the like-labeled points are connected in $G(r_0)$ and all the unlabeled points (namely $u_1, \ldots, u_{n'}$) are connected to a_1 as shown in Figure 4a. The algorithm $A(G(r), L)$ labels everything a and gets exactly half the labels right. As r is increased to r_i, u_i gets connected to b_1 and b_2 (Figure 4b). If the sequence u_i is alternately labeled, the loss increases and decreases alternately as all the predicted labels turn to b as r is increased to $r_{n'}$. Further increasing r may connect all the unlabeled points with true label a to a_2 and a_3 (Figure 4c), although this is not crucial to our argument. The rest of the proof gives concrete values of r and verifies that the construction is indeed feasible.

We will ensure all the pairwise distances are between 1 and 2, so that triangle inequality is always satisfied. It may also be readily verified that $O(\log n)$ dimensions suffice for our construction to exist. We start by defining some useful constants. We pick $r_-, r_+, r_{\max} \in (1, 2)$ such that $r_- < r_1 < \cdots < r_{n'} < r_+ < r_{\max}$,

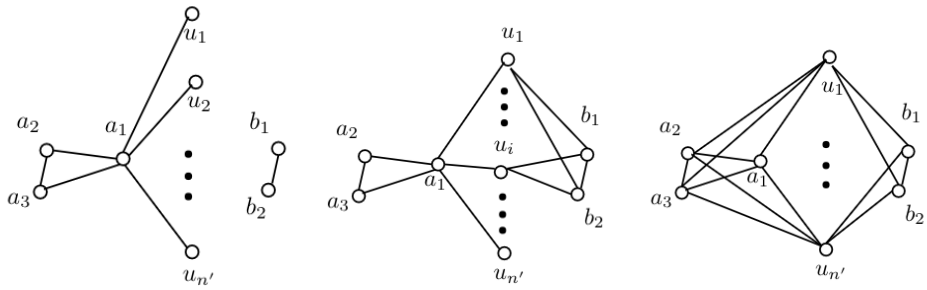
$$r_- = \frac{1 + r_1}{2},$$
$$r_+ = 1 + \frac{r_{n'}}{2},$$
$$r_{\max} = 1 + \frac{r_+}{2}.$$

We will now specify the distances of the labeled points. The points with the same label are close together and away from the oppositely labeled points.

$$\begin{aligned} d(a_i, a_j) &= r_-, && 1 \leq i < j \leq 3, \\ d(b_1, b_2) &= r_-, \\ d(a_i, b_j) &= r_{\max}, && 1 \leq i \leq 3, 1 \leq j \leq 2. \end{aligned}$$

Further, the unlabeled points are located as follows

$$\begin{aligned} d(a_1, u_k) &= r_-, && 1 \leq k \leq n', \\ d(b_i, u_k) &= r_k, && 1 \leq k \leq n', 1 \leq i \leq 2, \\ d(a_i, u_k) &= r_+, && 1 \leq k \leq n', 2 \leq i \leq 3, \\ d(u_i, u_j) &= r_{\max}, && 1 \leq i < j \leq n'. \end{aligned}$$



(a) $G(r_-)$ (b) $G(r_i)$ (c) $G(r_+)$

Figure 4: Graphs $G(r)$ as r is varied, for lower bound construction for pseudodimension of $\mathcal{H}_r$.

That is, all unknown points are closest to a_1, followed by b_i's, remaining a_i's and other u_i's in order. Further let the true labels of the unlabeled nodes be alternating with the index, i.e. u_k is a if and only if k is even.

We will now compute the loss for the soft labeling algorithm $A(G(r), L)$ of Zhu et al. [2003] as r varies from r_- to r_+, starting with $r = r_0 = r_-$. We note that our construction also works for other algorithms as well, for example the min-cut based approach of Blum and Chawla [2001], but omit the details.

For the graph $G(r_-)$, $A(G, L)$ labels each unknown node as a since each unknown point is a leaf node connected to a_1. Indeed if $f(a_1) = 1$, the quadratic objective attains the minimum of 0 for exactly $f(u_k) = 1$ for each $1 \leq k \leq n'$. This results in half the labels in the dataset being incorrectly labeled since we stipulate that half the unknown labels are of each category. This results in loss $l_{A(G(r_-),L)} =: l_{\text{high}}$ say.

Now as r is increased to r_1, the edges (b_i, u_1), $i = 1, 2$ are added with b_i labeled as $f(b_i) = 0$. This results in a fractional label of $\frac{1}{3}$ for $f(u_1)$ while $f(u_k) = 1$ for $k \neq 1$. Indeed the terms involving $f(u_1)$ in the objective are $(1 - f(u_1))^2 + 2f(u_1)^2$, which is minimized at $\frac{1}{3}$. Since u_1 has true label b, this results in a slightly smaller loss of $l_{A(G(r_1),L)} =: l_{\text{low}}$. This happens when A uses rounding, or in expectation if A uses randomized prediction with probability $f(u)$.

At the next critical point r_2, u_2 gets connected to b_i's and gets incorrectly classified as b. This increases the loss again to l_{high}. The loss function thus alternates as r is varied through the specified values, between l_{high} and l_{low}. We therefore set the witness w to something in between.

$$w = \frac{l_{\text{low}} + l_{\text{high}}}{2}.$$

□

Continued Proof of Theorem 10 We will now use Lemma 29 to prove our lower bound. Arbitarily choose $n' = n - 5$ (assumed to be a power of 2 just for convenient presentation) real numbers $r_{[000...01]} < r_{[000...10]} < \cdots < r_{[111...11]}$ in $(1, 2)$. The indices are increasing binary numbers of length $m = \log n'$. We create labeling instances using Lemma 29 which can be shattered by these r values. Instance $S_i = (G_i, L_i)$ corresponds to fluctuation of i-th bit b_i in our r_b sequence, where $b = (b_1, \ldots, b_m) \in \{0, 1\}^m$, i.e., we apply the lemma by using a subset of the r_b values which correspond to the bit flips in the i-th binary digit. For example, S_1 just needs a single bit flip (at $r_{[100...00]}$). The lemma gives us both the instances and corresponding witnesses w_i.

This construction ensures $\text{sign}(l_{A(G_i(r_b),L_i)} - w_i) = b_i$, i.e. the set of instances is shattered. Thus the pseudodimension is at least $\log(n - 5) = \Omega(\log n)$. □

Theorem 11. *The pseudo-dimension of $\mathcal{H}_\sigma$ is $\Theta(n)$.*

Proof. The upper bound trivially follows by noting that we have n vertices and therefore only 2^n possible labelings in an instance. Thus, for m problems, $2^m \leq m2^n$ gives $m = O(n)$. The rest of the proof deals with the lower bound.

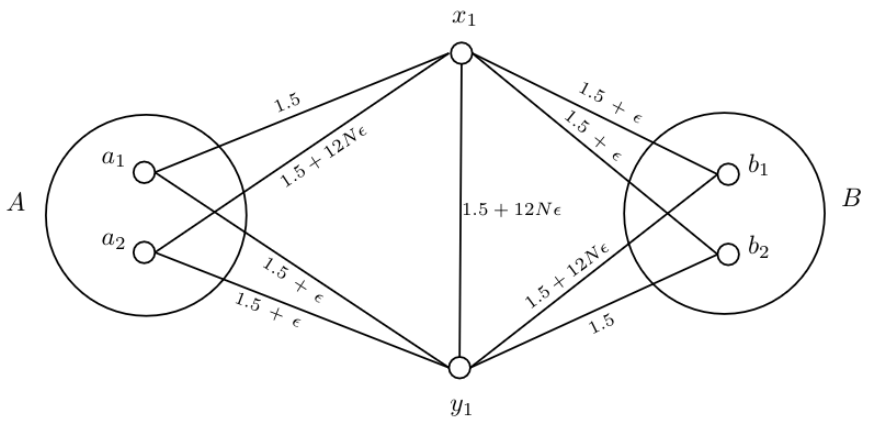


Figure 5: The base case of our inductive construction.

The plan for the proof is to first construct a graph where the edge weights are carefully selected, so that we have 2^N oscillations in the loss function with σ for $N = \Omega(n)$. Then we use this construction to create $\Theta(n)$ instances, each having a subset of the oscillations so that each interval leads to a unique labeling of the instances, for a total of 2^N labelings, which would imply pseudodimension is $\Omega(n)$. We will present our discussion in terms of the min-cut objective, for simplicity of presentation.

Graph construction: First a quick rough overview. We start with a pair of labeled nodes of each class, and a pair of unlabeled nodes which may be assigned either label depending on σ. We then build the graph in $N = (n-4)/2$ stages, adding two new nodes at each step with carefully chosen distances from existing nodes. Before adding the ith pair x_i, y_i of nodes, there will be 2^{i-1} intervals of σ such that the intervals correspond to distinct min-cuts which result in all possible labelings of $\{x_1, \ldots, x_{i-1}\}$. Moreover, y_j will be labeled differently from x_j in each of these intervals. The edges to the new nodes will ensure that the cuts that differ exactly in x_i will divide each of these intervals giving us 2^i intervals where distinct mincuts give all labelings of $\{x_1, \ldots, x_i\}$, and allowing an inductive proof. The challenge is that we only get to set $O(i)$ edges but seek properties about 2^i cuts, so we must do this carefully.

Let $\varsigma = e^{-1/\sigma^2}$. Notice $\varsigma \in (0, 1)$, and bijectively corresponds to $\sigma \in (0, \infty)$ (due to monotonicity) and therefore it suffices to specify intervals of ς corresponding to different labelings. Further we can specify distances $d(u, v)$ between pairs of nodes u, v by specifying the squared distance $d(u, v)^2$. For the remainder of this proof we will refer to $\delta(u, v) = d(u, v)^2$ by *distance* and set values in $[1.5, 1.6]$. Consequently, $d(u, v) \in (1.22, 1.27)$ and therefore the triangle inequality is always satisfied. Notice that with this notation, the graph weights will be $w(u, v) = \varsigma^{\delta(u,v)}$.

We now provide details of the construction. We have four labeled nodes as follows. a_1, a_2 are labeled 0 and are collectively denoted by $A = \{a_1, a_2\}$, similarly b_1, b_2 are labeled 1 and $B = \{b_1, b_2\}$. Note that edges between these nodes are on all or no cut separating A, B, we set the distances to 1.6 and call this graph G_0. We further add unlabeled nodes in pairs (x_j, y_j) in *rounds* $1 \leq j \leq N$. In round i, we construct graph G_i by adding nodes (x_i, y_i) to G_{i-1}. The distances are set to ensure that for G_N there are 2^N unique values of ς corresponding to distinct min-cuts, each giving a unique labeling for $\{x_1, \ldots, x_n\}$ (and the complementary labeling for $\{y_1, \ldots, y_n\}$). Moreover subsets of these points also obtain the unique labeling for $\{x_1, \ldots, x_i\}$ for each G_i.

We set the distances in round 1 such that there are intervals $I_0 = (\varsigma_0, \varsigma_0') \subset (0, 1)$ and $I_1 = (\varsigma_1, \varsigma_1') \subset (0, 1)$ such that $\varsigma_0' < \varsigma_1$ and (x_1, y_1) are labeled $(l, 1-l)$ in interval I_l. In general, there will be 2^{i-1} intervals at the end of round $i-1$, any interval $I^{(i-1)}$ will be split into disjoint intervals $I_0^{(i)}, I_1^{(i)} \subset I^{(i-1)}$ where labelings of $\{x_1, \ldots, x_{i-1}\}$ match that of $I^{(i-1)}$ and (x_i, y_i) are labeled $(l, 1-l)$ in $I_l^{(i)}$.

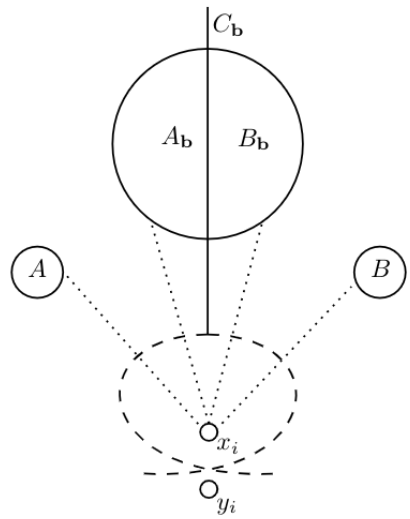


Figure 6: The inductive step in our lower bound construction for pseudodimension of $\mathcal{H}_\sigma$. The min-cut $C_{\mathbf{b}}$ is extended to two new min-cuts (depicted by dashed lines) for which labels of x_i, y_i are flipped, at controlled parameter intervals.

Now we set up the edges to achieve these properties. In round 1, we set the distances as follows.

$$\begin{aligned}\delta(x_1, a_1) = \delta(y_1, b_2) &= 1.5,\\ \delta(x_1, a_2) = \delta(y_1, b_1) = \delta(x_1, y_1) &= 1.5 + 12N\epsilon,\\ \delta(x_1, b_1) = \delta(x_1, b_2) = \delta(y_1, a_1) = \delta(y_1, a_2) &= 1.5 + \epsilon.\end{aligned}$$

where ϵ is a small positive quantity such that the largest distance $1.5 + 12N\epsilon < 1.6$. It is straightforward to verify that for $I_0 = (0, \frac{1}{2}^{1/\epsilon})$ we have that (x_1, y_1) are labeled $(0, 1)$ by determining the values of ς for which the corresponding cut is the min-cut (Figure 5). Indeed, we seek ς such that $w_{C01} = w(x_1, b_1) + w(x_1, b_2) + w(x_1, y_1) + w(y_1, a_1) + w(y_1, a_2)$ satisfies

$$\begin{aligned}w_{C01} \le w_{C00} &= w(x_1, b_1) + w(x_1, b_2) + w(y_1, b_1) + w(y_1, b_2),\\ w_{C01} \le w_{C11} &= w(x_1, a_1) + w(x_1, a_2) + w(y_1, a_1) + w(y_1, a_2),\\ w_{C01} \le w_{C10} &= w(x_1, a_1) + w(x_1, a_2) + w(x_1, y_1) + w(y_1, b_1) + w(y_1, b_2),\end{aligned}$$

which simultaneously hold for $\varsigma < \frac{1}{2}^{1/\epsilon}$.

Moreover, we can similarly conclude that (x_1, y_1) are labeled $(1, 0)$ for the interval $I_1 = (\varsigma_1, \varsigma_1')$ where $\varsigma_1 < \varsigma_1'$ are given by the two positive roots of the equation

$$1 - 2\varsigma^{\epsilon} + 2\varsigma^{12N\epsilon} = 0.$$

We now consider the inductive step, to set the distances and obtain an inductive proof of the claim above. In round i, the distances are as specified.

$$\begin{aligned}\delta(x_i, a_1) = \delta(y_i, b_2) &= 1.5,\\ \delta(x_i, a_2) = \delta(y_i, b_1) = \delta(x_i, y_i) &= 1.5 + 12N\epsilon,\\ \delta(x_i, b_1) = \delta(x_i, b_2) = \delta(y_i, a_1) = \delta(y_i, a_2) &= 1.5 + \epsilon,\\ \delta(x_i, y_j) = \delta(y_i, x_j) &= 1.5 + 6(2j-1)\epsilon \quad (1 \le j \le i-1),\\ \delta(x_i, x_j) = \delta(y_i, y_j) &= 1.5 + 12j\epsilon \qquad\quad (1 \le j \le i-1).\end{aligned}$$

We denote the (inductively hypothesized) 2^{i-1} ς-intervals at the end of round $i-1$ by $I_{\mathbf{b}}^{(i-1)}$, where $\mathbf{b} = \{b^{(1)}, \ldots, b^{(i-1)}\} \in \{0,1\}^{i-1}$ indicates the labels of $x_j, j \in [i-1]$ in $I_{\mathbf{b}}^{(i-1)}$. Min-cuts from round $i-1$ extend to min-cuts of round i depending on how the edges incident on (x_i, y_i) are set (Figure 6). It suffices to consider only those min-cuts where x_j and y_j have opposite labels for each

j. Consider an arbitrary such min-cut $C_{\mathbf{b}} = (A_{\mathbf{b}}, B_{\mathbf{b}})$ of G_{i-1} which corresponds to the interval $I_{\mathbf{b}}^{(i-1)}$, that is $A_{\mathbf{b}} = \{x_j \mid b^{(j)} = 0\} \cup \{y_j \mid b^{(j)} = 1\}$ and $B_{\mathbf{b}}$ contains the remaining unlabeled nodes of G_{i-1}. It extends to $C_{[\mathbf{b}\ 0]}$ and $C_{[\mathbf{b}\ 1]}$ for $\varsigma \in I_{\mathbf{b}}^{(i-1)}$ satisfying, respectively,

$$E_{\mathbf{b},0}(\varsigma) := 1 - 2\varsigma^{\epsilon} + F(C_{\mathbf{b}}; \varsigma) > 0,$$
$$E_{\mathbf{b},1}(\varsigma) := 1 - 2\varsigma^{\epsilon} + 2\varsigma^{12N\epsilon} + F(C_{\mathbf{b}}; \varsigma) < 0,$$

where $F(C_{\mathbf{b}}; \varsigma) = \sum_{z \in A_{\mathbf{b}}} \varsigma^{\delta(x_i, z)} - \sum_{z \in B_{\mathbf{b}}} \varsigma^{\delta(x_i, z)} = \sum_{z \in B_{\mathbf{b}}} \varsigma^{\delta(y_i, z)} - \sum_{z \in A_{\mathbf{b}}} \varsigma^{\delta(y_i, z)}$. If we show that the solutions of the above inequations have disjoint non-empty intersections with $\varsigma \in I_{\mathbf{b}}^{(i-1)}$, our induction step is complete. We will use an indirect approach for this.

For $1 \leq i \leq N$, given $\mathbf{b} = \{b^{(1)}, \ldots, b^{(i-1)}\} \in \{0,1\}^{i-1}$, let $E_{\mathbf{b},0}$ and $E_{\mathbf{b},1}$ denote the expressions (exponential polynomials in ς) in round i which determine labels of (x_i, y_i), in the case where for all $1 \leq j < i$, x_j is labeled $b^{(j)}$ (and let $E_{\phi,0}, E_{\phi,1}$ denote the expressions for round 1). Let $\varsigma_{\mathbf{b},i} \in (0,1)$ denote the smallest solution to $E_{\mathbf{b},i} = 0$. Then we need to show the $\varsigma_{\mathbf{b},i}$'s are well-defined and follow a specific ordering. This ordering is completely specified by two conditions:

(i) $\varsigma_{[\mathbf{b}\ 0],1} < \varsigma_{[\mathbf{b}],0} < \varsigma_{[\mathbf{b}],1} < \varsigma_{[\mathbf{b}\ 1],0}$, and

(ii) $\varsigma_{[\mathbf{b}\ 0\ \mathbf{c}],1} < \varsigma_{[\mathbf{b}\ 1\ \mathbf{d}],0}$

for all $\mathbf{b}, \mathbf{c}, \mathbf{d} \in \cup_{i<N}\{0,1\}^i$ and $|\mathbf{c}| = |\mathbf{d}|$.

First we make a quick observation that all $\varsigma_{\mathbf{b},i}$'s are well-defined and less than $(3/4)^{1/\epsilon}$. To do this, it will suffice to note that $E_{\mathbf{b},i}(0) = 1$ and $E_{\mathbf{b},i}(\frac{3}{4}^{1/\epsilon}) < 0$ for all $\mathbf{b}, i$, since the functions are continuous in $(0, \frac{3}{4}^{1/\epsilon})$. This holds because

$$\begin{aligned}
E_{\mathbf{b},0}\left(\frac{3}{4}^{1/\epsilon}\right) < E_{\mathbf{b},1}\left(\frac{3}{4}^{1/\epsilon}\right) &= 1 - \frac{3}{2} + \left(\frac{3}{4}\right)^{12N} + F\left(C_{\mathbf{b}}; \frac{3}{4}^{1/\epsilon}\right) \\
&\leq -\frac{1}{2} + \left(\frac{3}{4}\right)^{12N} + \sum_{j=1}^{|\mathbf{b}|} \left(\frac{3}{4}\right)^{6j} \left(1 - \left(\frac{3}{4}\right)^{6j}\right) \\
&< -\frac{1}{2} + \sum_{j=1}^{N} \left(\frac{3}{4}\right)^{6j} \\
&< 0
\end{aligned}$$

Let's now consider condition (i). We begin by showing $\varsigma_{[\mathbf{b}],0} < \varsigma_{[\mathbf{b}],1}$ for any $\mathbf{b}$. The exponential polynomials $E_{\mathbf{b},0}$ and $E_{\mathbf{b},1}$ both evaluate to 1 for $\varsigma = 0$ (since $|A_{\mathbf{b}}| = |B_{\mathbf{b}}| = |\mathbf{b}|$) and decrease monotonically (verified by elementary calculus) till their respective smallest zeros $\varsigma_{[\mathbf{b}],0}, \varsigma_{[\mathbf{b}],1}$. But then $E_{\mathbf{b},1}(\varsigma_{[\mathbf{b}],0}) = 2(\varsigma_{[\mathbf{b}],0})^{12N\epsilon} > 0$, which implies $\varsigma_{[\mathbf{b}],0} < \varsigma_{[\mathbf{b}],1}$. Now, to show $\varsigma_{[\mathbf{b}\ 0],1} < \varsigma_{[\mathbf{b}],0}$, note that $E_{[\mathbf{b}\ 0],1}(\varsigma) - E_{[\mathbf{b}],0}(\varsigma) = 2\varsigma^{12N\epsilon} + \varsigma^{12i\epsilon} - \varsigma^{(12i-6)\epsilon} = \varsigma^{(12i-6)\epsilon}(2\varsigma^{(12(N-i)+6)\epsilon} + \varsigma^{6\epsilon} - 1)$ where $1 \leq i = |\mathbf{b}| + 1 < N$. Since $\varsigma_{[\mathbf{b}],0} < \frac{3}{4}^{1/\epsilon}$, it follows that $E_{[\mathbf{b}\ 0],1}(\varsigma_{[\mathbf{b}],0}) < 0$, which implies $\varsigma_{[\mathbf{b}\ 0],1} < \varsigma_{[\mathbf{b}],0}$. Similarly, it is readily verified that $\varsigma_{[\mathbf{b}],1} < \varsigma_{[\mathbf{b}\ 1],0}$, establishing (i).

Finally, to show (ii), note that $E_{[\mathbf{b}\ 0\ \mathbf{c}],1}(\varsigma) - E_{[\mathbf{b}\ 0\ \mathbf{d}],0}(\varsigma) = 2\varsigma^{12N\epsilon} + \varsigma^{12i\epsilon} - \varsigma^{(12i-6)\epsilon} + \varsigma^{12i\epsilon}(F(C_{\mathbf{c}}; \varsigma) - F(C_{\mathbf{d}}; \varsigma)) = \varsigma^{(12i-6)\epsilon}(2\varsigma^{(12(N-i)+6)\epsilon} + \varsigma^{6\epsilon} - 1 + \varsigma^{6\epsilon}(F(C_{\mathbf{c}}; \varsigma) - F(C_{\mathbf{d}}; \varsigma)))$. Again, similar to above, we use $\varsigma_{[\mathbf{b}\ 0\ \mathbf{d}],0} < \frac{3}{4}^{1/\epsilon}$ in this expression to get $E_{[\mathbf{b}\ 0\ \mathbf{c}],1}(\varsigma_{[\mathbf{b}\ 0\ \mathbf{d}],0}) < 0$. Since the exponential polynomials decay monotonically with ς till their first roots, (ii) follows.

Problem instances: We will now show the graph instances and witnesses to establish the pseudodimension bound. Our graphs will be G_i from the above construction (padded appropriately such that the min-cut intervals do not change, if we insist each instance has exactly n nodes), and the shattering family σ_b ($b = (b_1, \ldots, b_N) \in \{0,1\}^N$) will be 2^N values of σ corresponding to the 2^N intervals of ς with distinct min-cuts in G_N described above. To obtain the witnesses, we set the labels so that only the last pair of nodes (x_i, y_i) have different labels (i.e. labels are same for all $(x_j, y_j), j < i$)

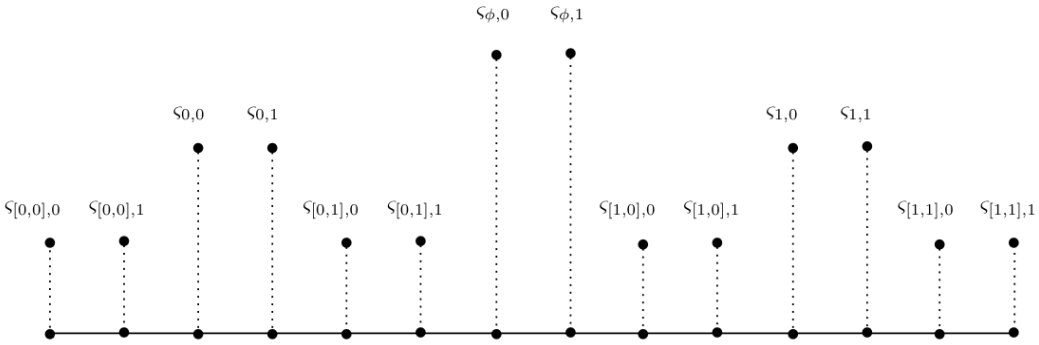
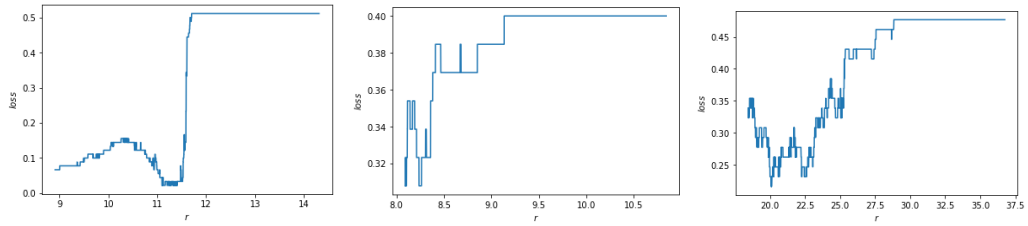


Figure 7: Relative positions of critical values of the parameter $\varsigma = e^{-1/\sigma^2}$.

and therefore the loss function oscillates 2^i times as (x_i, y_i) are correctly and incorrectly labeled in alternating intervals. The intervals of successive G_i are nested precisely so that σ_b shatter the instances for the above labelings/witnesses. Thus, we have shown that the pseudodimension is $\Omega(N) = \Omega((n-4)/2) = \Omega(n)$. □

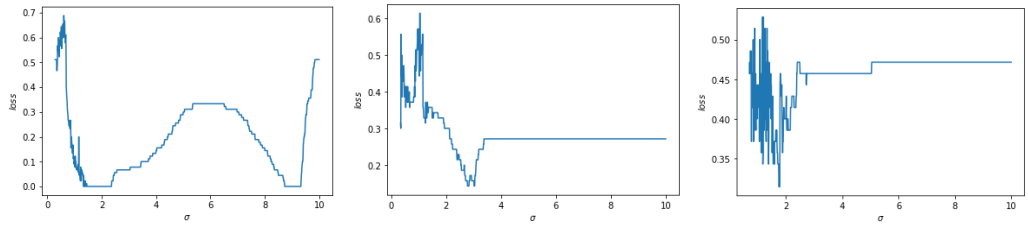
E Further experiments

We include plots for variation of loss function with graph hyperparameters r, σ for unweighted graphs $G(r)$ and weighted graphs $G(\sigma)$ for single instances of datasets drawn as described in Section 6. We examine the full variation of performance of graph-based semi-supervised learning for all possible graphs $G(r)$ ($r_{\min} < r < r_{\max}$) and $G(\sigma)$ for $\sigma \in [0, 10]$ (Figures 8, 9).



(a) MNIST (b) Omniglot (c) CIFAR-10

Figure 8: Loss for different unweighted graphs as a function of the threshold r.



(a) MNIST (b) Omniglot (c) CIFAR-10

Figure 9: Loss for different weighted graphs as a function of the parameter σ.