On the Generalization Ability of Online Learning Algorithms for Pairwise Loss Functions

Purushottam Kar PURUSHOT@CSE.IITK.AC.IN
Department of Computer Science and Engineering, Indian Institute of Technology, Kanpur, UP 208 016, INDIA.

Bharath K Sriperumbudur BS493@STATSLAB.CAM.AC.UK
Statistical Laboratory, Centre for Mathematical Sciences, Wilberforce Road, Cambridge, CB3 0WB, ENGLAND.

Prateek Jain PRAJAIN@MICROSOFT.COM
Microsoft Research India, "Vigyan", #9, Lavelle Road, Bangalore, KA 560 001, INDIA.

Harish C Karnick HK@CSE.IITK.AC.IN
Department of Computer Science and Engineering, Indian Institute of Technology, Kanpur, UP 208 016, INDIA.

Abstract

In this paper, we study the generalization properties of online learning based stochastic methods for supervised learning problems where the loss function is dependent on more than one training sample (e.g., metric learning, ranking). We present a generic decoupling technique that enables us to provide Rademacher complexity-based generalization error bounds. Our bounds are in general tighter than those obtained by Wang et al. (2012) for the same problem. Using our decoupling technique, we are further able to obtain fast convergence rates for strongly convex pairwise loss functions. We are also able to analyze a class of memory efficient online learning algorithms for pairwise learning problems that use only a bounded subset of past training samples to update the hypothesis at each step. Finally, in order to complement our generalization bounds, we propose a novel memory efficient online learning algorithm for higher order learning problems with bounded regret guarantees.

1. Introduction

Several supervised learning problems involve working with pairwise or higher order loss functions, i.e., loss functions that depend on more than one training sample. Take for example the *metric learning* problem (Jin et al., 2009), where the goal is to learn a metric M that brings points of a similar label together while keeping differently labeled points apart. In this case the loss function used is a pairwise loss function $\ell(M, (\mathbf{x}, y), (\mathbf{x}', y')) = \phi\left(yy'\left(1 - M(\mathbf{x}, \mathbf{x}')\right)\right)$ where ϕ is the hinge loss function. In general, a pairwise loss function is of the form $\ell : \mathcal{H} \times \mathcal{X} \times \mathcal{X} \to \mathbb{R}^+$ where $\mathcal{H}$ is the hypothesis space and $\mathcal{X}$ is the input domain. Other examples include preference learning (Xing et al., 2002), ranking (Agarwal & Niyogi, 2009), AUC maximization (Zhao et al., 2011) and multiple kernel learning (Kumar et al., 2012).

In practice, algorithms for such problems use intersecting pairs of training samples to learn. Hence the training data pairs are not i.i.d. and consequently, standard generalization error analysis techniques do not apply to these algorithms. Recently, the analysis of *batch* algorithms learning from such coupled samples has received much attention (Cao et al., 2012; Clémençon et al., 2008; Brefeld & Scheffer, 2005) where a dominant idea has been to use an alternate representation of the U-statistic and provide uniform convergence bounds. Another popular approach has been to use algorithmic stability (Agarwal & Niyogi, 2009; Jin et al., 2009) to obtain algorithm-specific results.

While batch algorithms for pairwise (and higher-order) learning problems have been studied well theoretically, online learning based stochastic algorithms are more popular in practice due to their scalability. However, their generalization properties were not studied until recently. Wang et al. (2012) provided the first generalization error analysis of online learning methods

Proceedings of the 30th International Conference on Machine Learning, Atlanta, Georgia, USA, 2013. JMLR: W&CP volume 28.

applied to pairwise loss functions. In particular, they showed that such higher-order online learning methods also admit online to batch conversion bounds (similar to those for first-order problems (Cesa-Bianchi et al., 2001)) which can be combined with regret bounds to obtain generalization error bounds. However, due to their proof technique and dependence on L_∞ covering numbers of function classes, their bounds are not tight and have a strong dependence on the dimensionality of the input space.

In literature, there are several instances where Rademacher complexity based techniques achieve sharper bounds than those based on covering numbers (Kakade et al., 2008). However, the coupling of different input pairs in our problem does not allow us to use such techniques directly.

In this paper we introduce a generic technique for analyzing online learning algorithms for higher order learning problems. Our technique, that uses an extension of Rademacher complexities to higher order function classes (instead of covering numbers), allows us to give bounds that are tighter than those of (Wang et al., 2012) and that, for several learning scenarios, have no dependence on input dimensionality at all.

Key to our proof is a technique we call *Symmetrization of Expectations* which acts as a decoupling step and allows us to reduce excess risk estimates to Rademacher complexities of function classes. (Wang et al., 2012), on the other hand, perform a symmetrization with probabilities which, apart from being more involved, yields suboptimal bounds. Another advantage of our technique is that it allows us to obtain *fast* convergence rates for learning algorithms that use *strongly convex* loss functions. Our result, that uses a novel two stage proof technique, extends a similar result in the first order setting by Kakade & Tewari (2008) to the pairwise setting.

Wang et al. (2012) (and our results mentioned above) assume an online learning setup in which a stream of points $\mathbf{z}_1, \ldots, \mathbf{z}_n$ is observed and the penalty function used at the t^{th} step is $\hat{\mathcal{L}}_t(h) = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \ell(h, \mathbf{z}_t, \mathbf{z}_\tau)$. Consequently, the results of Wang et al. (2012) expect regret bounds with respect to these *all-pairs* penalties $\hat{\mathcal{L}}_t$. This requires one to use/store all previously seen points which is computationally/storagewise expensive and hence in practice, learning algorithms update their hypotheses using only a bounded subset of the past samples (Zhao et al., 2011).

In the above mentioned setting, we are able to give generalization bounds that only require algorithms to give regret bounds with respect to *finite-buffer* penalty functions such as $\hat{\mathcal{L}}_t^{\text{buf}}(h) = \frac{1}{|B|}\sum_{\mathbf{z}\in B} \ell(h, \mathbf{z}_t, \mathbf{z})$ where B is a *buffer* that is updated at each step. Our proofs hold for any *stream oblivious* buffer update policy including FIFO and the widely used reservoir sampling policy (Vitter, 1985; Zhao et al., 2011)[1].

To complement our online to batch conversion bounds, we also provide a memory efficient online learning algorithm that works with bounded buffers. Although our algorithm is constrained to observe and learn using the *finite-buffer* penalties $\hat{\mathcal{L}}_t^{\text{buf}}$ alone, we are still able to provide high confidence regret bounds with respect to the *all-pairs* penalty functions $\hat{\mathcal{L}}_t$. We note that Zhao et al. (2011) also propose an algorithm that uses finite buffers and claim an *all-pairs* regret bound for the same. However, their regret bound does not hold due to a subtle mistake in their proof.

We also provide empirical validation of our proposed online learning algorithm on AUC maximization tasks and show that our algorithm performs competitively with that of (Zhao et al., 2011), in addition to being able to offer theoretical regret bounds.

Our Contributions:

(a) We provide a generic online-to-batch conversion technique for higher-order supervised learning problems offering bounds that are sharper than those of (Wang et al., 2012).

(b) We obtain fast convergence rates when loss functions are *strongly convex*.

(c) We analyze online learning algorithms that are constrained to learn using a finite buffer.

(d) We propose a novel online learning algorithm that works with finite buffers but is able to provide a high confidence regret bound with respect to the *all-pairs* penalty functions.

2. Problem Setup

For ease of exposition, we introduce an online learning model for higher order supervised learning problems in this section; concrete learning instances such as AUC maximization and metric learning are given in Section 6. For sake of simplicity, we restrict ourselves to pairwise problems in this paper; our techniques can be readily extended to higher order problems as well.

For pairwise learning problems, our goal is to learn a

[1] Independently, Wang et al. (2013) also extended their proof to give similar guarantees. However, their bounds hold only for the FIFO update policy and have worse dependence on dimensionality in several cases (see Section 5).

real valued *bivariate* function $h^* : \mathcal{X} \times \mathcal{X} \to \mathcal{Y}$, where $h^* \in \mathcal{H}$, under some loss function $\ell : \mathcal{H} \times \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}^+$ where $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$.

The online learning algorithm is given sequential access to a stream of elements $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n$ chosen i.i.d. from the domain $\mathcal{Z}$. Let $Z^t := \{\mathbf{z}_1, \ldots, \mathbf{z}_t\}$. At each time step $t = 2 \ldots n$, the algorithm posits a hypothesis $h_{t-1} \in \mathcal{H}$ upon which the element $\mathbf{z}_t$ is revealed and the algorithm incurs the following penalty:

$$\hat{\mathcal{L}}_t(h_{t-1}) = \frac{1}{t-1} \sum_{\tau=1}^{t-1} \ell(h_{t-1}, \mathbf{z}_t, \mathbf{z}_\tau). \tag{1}$$

For any $h \in \mathcal{H}$, we define its expected risk as:

$$\mathcal{L}(h) := \mathop{\mathbb{E}}_{\mathbf{z},\mathbf{z}'} \left[\!\left[\ell(h, \mathbf{z}, \mathbf{z}') \right]\!\right]. \tag{2}$$

Our aim is to present an ensemble $h_1, \ldots, h_{n-1}$ such that the expected risk of the ensemble is small. More specifically, we desire that, for some small $\epsilon > 0$,

$$\frac{1}{n-1} \sum_{t=2}^{n} \mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \epsilon,$$

where $h^* = \mathop{\arg\min}_{h \in \mathcal{H}} \mathcal{L}(h)$ is the population risk minimizer. Note that this allows us to do hypothesis selection in a way that ensures small expected risk. Specifically, if one chooses a hypothesis as $\hat{h} := \frac{1}{(n-1)} \sum_{t=2}^{n} h_{t-1}$ (for convex ℓ) or $\hat{h} := \mathop{\arg\min}_{t=2,\ldots,n} \mathcal{L}(h_t)$ then we have $\mathcal{L}(\hat{h}) \leq \mathcal{L}(h^*) + \epsilon$.

Since the model presented above requires storing all previously seen points, it becomes unusable in large scale learning scenarios. Instead, in practice, a *sketch* of the stream is maintained in a buffer B of capacity s. At each step, the penalty is now incurred only on the pairs $\{(\mathbf{z}_t, \mathbf{z}) : \mathbf{z} \in B_t\}$ where B_t is the state of the buffer at time t. That is,

$$\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) = \frac{1}{|B_t|} \sum_{\mathbf{z} \in B_t} \ell(h_{t-1}, \mathbf{z}_t, \mathbf{z}). \tag{3}$$

We shall assume that the buffer is updated at each step using some *stream oblivious* policy such as FIFO or Reservoir sampling (Vitter, 1985) (see Section 5).

In Section 3, we present online-to-batch conversion bounds for online learning algorithms that give regret bounds w.r.t. penalty functions given by (1). In Section 4, we extend our analysis to algorithms using strongly convex loss functions. In Section 5 we provide generalization error bounds for algorithms that give regret bounds w.r.t. *finite-buffer* penalty functions given by (3). Finally in section 7 we present a novel memory efficient online learning algorithm with regret bounds.

3. Online to Batch Conversion Bounds for Bounded Loss Functions

We now present our generalization bounds for algorithms that provide regret bounds with respect to the *all-pairs* loss functions (see Eq. (1)). Our results give tighter bounds and have a much better dependence on input dimensionality than the bounds given by Wang et al. (2012). See Section 3.1 for a detailed comparison.

As was noted by (Wang et al., 2012), the generalization error analysis of online learning algorithms in this setting does not follow from existing techniques for first-order problems (such as (Cesa-Bianchi et al., 2001; Kakade & Tewari, 2008)). The reason is that the terms $V_t = \hat{\mathcal{L}}_t(h_{t-1})$ do not form a martingale due to the intersection of training samples in V_t and $V_\tau, \tau < t$.

Our technique, that aims to utilize the Rademacher complexities of function classes in order to get tighter bounds, faces yet another challenge at the *symmetrization* step, a precursor to the introduction of Rademacher complexities. It turns out that, due to the coupling between the "head" variable $\mathbf{z}_t$ and the "tail" variables $\mathbf{z}_\tau$ in the loss function $\hat{\mathcal{L}}_t$, a standard symmetrization between true $\mathbf{z}_\tau$ and ghost $\tilde{\mathbf{z}}_\tau$ samples does not succeed in generating Rademacher averages and instead yields complex looking terms.

More specifically, suppose we have *true* variables $\mathbf{z}_t$ and *ghost* variables $\tilde{\mathbf{z}}_t$ and are in the process of bounding the expected excess risk by analyzing expressions of the form

$$E_{\text{orig}} = \ell(h_{t-1}, \mathbf{z}_t, \mathbf{z}_\tau) - \ell(h_{t-1}, \tilde{\mathbf{z}}_t, \tilde{\mathbf{z}}_\tau).$$

Performing a traditional symmetrization of the variables $\mathbf{z}_\tau$ with $\tilde{\mathbf{z}}_\tau$ would give us expressions of the form

$$E_{\text{symm}} = \ell(h_{t-1}, \mathbf{z}_t, \tilde{\mathbf{z}}_\tau) - \ell(h_{t-1}, \tilde{\mathbf{z}}_t, \mathbf{z}_\tau).$$

At this point the analysis hits a barrier since unlike first order situations, we cannot relate E_{symm} to E_{orig} by means of introducing Rademacher variables.

We circumvent this problem by using a technique that we call *Symmetrization of Expectations*. The technique allows us to use standard symmetrization to obtain Rademacher complexities. More specifically, we analyze expressions of the form

$$E'_{\text{orig}} = \mathop{\mathbb{E}}_{\mathbf{z}} \left[\!\left[\ell(h_{t-1}, \mathbf{z}, \mathbf{z}_\tau) \right]\!\right] - \mathop{\mathbb{E}}_{\mathbf{z}} \left[\!\left[\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}_\tau) \right]\!\right]$$

which upon symmetrization yield expressions such as

$$E'_{\text{symm}} = \mathop{\mathbb{E}}_{\mathbf{z}} \left[\!\left[\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}_\tau) \right]\!\right] - \mathop{\mathbb{E}}_{\mathbf{z}} \left[\!\left[\ell(h_{t-1}, \mathbf{z}, \mathbf{z}_\tau) \right]\!\right]$$

which allow us to introduce Rademacher variables since $E'_{\text{symm}} = -E'_{\text{orig}}$. This idea is exploited by the

lemma given below that relates the expected risk of the ensemble to the penalties incurred during the online learning process. In the following we use the following extension of Rademacher averages (Kakade et al., 2008) to bivariate function classes:

$$\mathcal{R}_n(\mathcal{H}) = \mathbb{E}\left[\sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{\tau=1}^{n} \epsilon_\tau h(\mathbf{z}, \mathbf{z}_\tau)\right]$$

where the expectation is over ϵ_τ, $\mathbf{z}$ and $\mathbf{z}_\tau$. We shall denote composite function classes as follows : $\ell \circ \mathcal{H} := \{(\mathbf{z}, \mathbf{z}') \mapsto \ell(h, \mathbf{z}, \mathbf{z}'), h \in \mathcal{H}\}$.

Lemma 1. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a bounded loss function $\ell : \mathcal{H} \times \mathcal{Z} \times \mathcal{Z} \to [0, B]$. Then for any $\delta > 0$, we have with probability at least $1 - \delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) \leq \frac{1}{n-1}\sum_{t=2}^{n} \hat{\mathcal{L}}_t(h_{t-1}) + \frac{2}{n-1}\sum_{t=2}^{n} \mathcal{R}_{t-1}(\ell \circ \mathcal{H}) + 3B\sqrt{\frac{\log\frac{n}{\delta}}{n-1}}.$$

The proof of the lemma involves decomposing the excess risk term into a martingale difference sequence and a residual term in a manner similar to (Wang et al., 2012). The martingale sequence, being a bounded one, is shown to converge using the Azuma-Hoeffding inequality. The residual term is handled using uniform convergence techniques involving Rademacher averages. The complete proof of the lemma is given in the Appendix A.

Similar to Lemma 1, the following converse relation between the population and empirical risk of the population risk minimizer h^* can also be shown.

Lemma 2. *For any $\delta > 0$, we have with probability at least $1 - \delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n} \hat{\mathcal{L}}_t(h^*) \leq \mathcal{L}(h^*) + \frac{2}{n-1}\sum_{t=2}^{n} \mathcal{R}_{t-1}(\ell \circ \mathcal{H}) + 3B\sqrt{\frac{\log\frac{1}{\delta}}{n-1}}.$$

An online learning algorithm will be said to have an *all-pairs* regret bound $\mathfrak{R}_n$ if it presents an ensemble $h_1, \ldots, h_{n-1}$ such that

$$\sum_{t=2}^{n} \hat{\mathcal{L}}_t(h_{t-1}) \leq \inf_{h\in\mathcal{H}} \sum_{t=2}^{n} \hat{\mathcal{L}}_t(h) + \mathfrak{R}_n.$$

Suppose we have an online learning algorithm with a regret bound $\mathfrak{R}_n$. Then combining Lemmata 1 and 2 gives us the following online to batch conversion bound:

Theorem 3. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a B-bounded loss function ℓ that guarantees a regret bound of $\mathfrak{R}_n$. Then for any $\delta > 0$, we have with probability at least $1 - \delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{4}{n-1}\sum_{t=2}^{n} \mathcal{R}_{t-1}(\ell \circ \mathcal{H}) + \frac{\mathfrak{R}_n}{n-1} + 6B\sqrt{\frac{\log\frac{n}{\delta}}{n-1}}.$$

As we shall see in Section 6, for several learning problems, the Rademacher complexities behave as $\mathcal{R}_{t-1}(\ell \circ \mathcal{H}) \leq C_d \cdot \mathcal{O}\left(\frac{1}{\sqrt{t-1}}\right)$ where C_d is a constant dependent only on the dimension d of the input space and the $\mathcal{O}(\cdot)$ notation hides constants dependent on the domain size and the loss function. This allows us to bound the excess risk as follows:

$$\frac{\sum_{t=2}^{n} \mathcal{L}(h_{t-1})}{n-1} \leq \mathcal{L}(h^*) + \frac{\mathfrak{R}_n}{n-1} + \mathcal{O}\left(\frac{C_d + \sqrt{\log(n/\delta)}}{\sqrt{n-1}}\right).$$

Here, the error decreases with n at a standard $1/\sqrt{n}$ rate (up to a $\sqrt{\log n}$ factor), similar to that obtained by Wang et al. (2012). However, for several problems the above bound can be significantly tighter than those offered by covering number based arguments. We provide below a detailed comparison of our results with those of Wang et al. (2012).

3.1. Discussion on the nature of our bounds

As mentioned above, our proof enables us to use Rademacher complexities which are typically easier to analyze and provide tighter bounds (Kakade et al., 2008). In particular, as shown in Section 6, for L_2 regularized learning formulations, the Rademacher complexities are dimension independent i.e. $C_d = 1$. Consequently, unlike the bounds of (Wang et al., 2012) that have a linear dependence on d, our bound becomes independent of the input space dimension. For sparse learning formulations with L_1 or trace norm regularization, we have $C_d = \sqrt{\log d}$ giving us a mild dependence on the input dimensionality.

Our bounds are also tighter that those of (Wang et al., 2012) in general. Whereas we provide a confidence bound of $\delta < \exp\left(-n\epsilon^2 + \log n\right)$, (Wang et al., 2012) offer a weaker bound $\delta < (1/\epsilon)^d \exp\left(-n\epsilon^2 + \log n\right)$.

An artifact of the proof technique of (Wang et al., 2012) is that their proof is required to exclude a constant fraction of the ensemble $(h_1, \ldots, h_{cn})$ from the

analysis, failing which their bounds turn vacuous. Our proof on the other hand is able to give guarantees for the *entire* ensemble.

In addition to this, as the following sections show, our proof technique enjoys the flexibility of being extendable to give fast convergence guarantees for strongly convex loss functions as well as being able to accommodate learning algorithms that use finite buffers.

4. Fast Convergence Rates for Strongly Convex Loss Functions

In this section we extend results of the previous section to give *fast* convergence guarantees for online learning algorithms that use strongly convex loss functions of the following form: $\ell(h, \mathbf{z}, \mathbf{z}') = g(\langle h, \phi(\mathbf{z}, \mathbf{z}')\rangle) + r(h)$, where g is a convex function and $r(h)$ is a σ-strongly convex regularizer (see Section 6 for examples) i.e. $\forall h_1, h_2 \in \mathcal{H}$ and $\alpha \in [0, 1]$, we have

$$r(\alpha h_1 + (1-\alpha)h_2) \leq \alpha r(h_1) + (1-\alpha) r(h_2) - \frac{\sigma}{2}\alpha(1-\alpha)\left\|h_1 - h_2\right\|^2.$$

For any norm $\|\cdot\|$, let $\|\cdot\|_*$ denote its dual norm. Our analysis reduces the pairwise problem to a first order problem and a martingale convergence problem. We require the following *fast* convergence bound in the standard first order *batch* learning setting:

Theorem 4. *Let $\mathcal{F}$ be a closed and convex set of functions over $\mathcal{X}$. Let $\wp(f, \mathbf{x}) = p(\langle f, \phi(\mathbf{x})\rangle) + r(f)$, for a σ-strongly convex function r, be a loss function with $\mathcal{P}$ and $\hat{\mathcal{P}}$ as the associated population and empirical risk functionals and f^* as the population risk minimizer. Suppose $\wp$ is L-Lipschitz and $\|\phi(\mathbf{x})\|_* \leq R, \forall \mathbf{x} \in \mathcal{X}$. Then w.p. $1-\delta$, for any $\epsilon > 0$, we have for all $f \in \mathcal{F}$,*

$$\mathcal{P}(f) - \mathcal{P}(f^*) \leq (1+\epsilon)\left(\hat{\mathcal{P}}(f) - \hat{\mathcal{P}}(f^*)\right) + \frac{C_\delta}{\epsilon\sigma n}$$

where $C_\delta = C_d^2 \cdot (4(1+\epsilon)LR)^2\,(32 + \log(1/\delta))$ and C_d is the dependence of the Rademacher complexity of the class $\mathcal{F}$ on the input dimensionality d.

The above theorem is a minor modification of a similar result by Sridharan et al. (2008) and the proof (given in Appendix B) closely follows their proof as well. We can now state our online to batch conversion result for strongly convex loss functions.

Theorem 5. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a B-bounded, L-Lipschitz and σ-strongly convex loss function ℓ. Further suppose the learning algorithm guarantees a regret bound of $\mathfrak{R}_n$. Let $\mathfrak{V}_n = \max\left\{\mathfrak{R}_n, 2C_d^2 \log n \log(n/\delta)\right\}$ Then for any $\delta > 0$, we have with probability at least $1-\delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{\mathfrak{R}_n}{n-1} + C_d \cdot \mathcal{O}\left(\frac{\sqrt{\mathfrak{V}_n \log n \log(n/\delta)}}{n-1}\right),$$

where the $\mathcal{O}(\cdot)$ notation hides constants dependent on domain size and the loss function such as L, B and σ.

The decomposition of the excess risk in this case is not made explicitly but rather emerges as a side-effect of the proof progression. The proof starts off by applying Theorem 4 to the hypothesis in each round with the following loss function $\wp(h, \mathbf{z}') := \mathop{\mathbb{E}}_{\mathbf{z}}[\ell(h, \mathbf{z}, \mathbf{z}')]$. Applying the regret bound to the resulting expression gives us a martingale difference sequence which we then bound using Bernstein-style inequalities and a proof technique from (Kakade & Tewari, 2008). The complete proof is given in Appendix C.

We now note some properties of this result. The effective dependence of the above bound on the input dimensionality is C_d^2 since the expression $\sqrt{\mathfrak{V}_n}$ hides a C_d term. We have $C_d^2 = 1$ for non sparse learning formulations and $C_d^2 = \log d$ for sparse learning formulations. We note that our bound matches that of Kakade & Tewari (2008) (for *first*-order learning problems) up to a logarithmic factor.

5. Analyzing Online Learning Algorithms that use Finite Buffers

In this section, we present our online to batch conversion bounds for algorithms that work with *finite-buffer* loss functions $\hat{\mathcal{L}}_t^{\text{buf}}$. Recall that an online learning algorithm working with finite buffers incurs a loss $\hat{\mathcal{L}}_t^{\text{buf}}(h) = \frac{1}{|B_t|}\sum_{\mathbf{z}\in B_t}\ell(h_{t-1}, \mathbf{z}_t, \mathbf{z})$ at each step where B_t is the state of the buffer at time t.

An online learning algorithm will be said to have a *finite-buffer* regret bound $\mathfrak{R}_n^{\text{buf}}$ if it presents an ensemble $h_1, \ldots, h_{n-1}$ such that

$$\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \inf_{h\in\mathcal{H}}\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{\text{buf}}(h) \leq \mathfrak{R}_n^{\text{buf}}.$$

For our guarantees to hold, we require the buffer update policy used by the learning algorithm to be *stream oblivious*. More specifically, we require the buffer update rule to decide upon the inclusion of a particular point $\mathbf{z}_i$ in the buffer based only on its stream index $i \in [n]$. Popular examples of stream oblivious policies include Reservoir sampling (Vitter, 1985) (referred to

as **RS** henceforth) and FIFO. Stream oblivious policies allow us to decouple buffer construction randomness from training sample randomness which makes analysis easier; we leave the analysis of *stream aware* buffer update policies as a topic of future research.

In the above mentioned setting, we can prove the following online to batch conversion bounds:

Theorem 6. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a finite buffer of capacity s and a B-bounded loss function ℓ. Moreover, suppose that the algorithm guarantees a regret bound of $\mathfrak{R}_n^{buf}$. Then for any $\delta > 0$, we have with probability at least $1-\delta$,*

$$\frac{\sum_{t=2}^{n} \mathcal{L}(h_{t-1})}{n-1} \leq \mathcal{L}(h^*) + \frac{\mathfrak{R}_n^{buf}}{n-1} + \mathcal{O}\left(\frac{C_d}{\sqrt{s}} + B\sqrt{\frac{\log \frac{n}{\delta}}{s}}\right)$$

If the loss function is Lipschitz and strongly convex as well, then with the same confidence, we have

$$\frac{\sum_{t=2}^{n} \mathcal{L}(h_{t-1})}{n-1} \leq \mathcal{L}(h^*) + \frac{\mathfrak{R}_n^{buf}}{n-1} + C_d \cdot \mathcal{O}\left(\sqrt{\frac{\mathfrak{W}_n \log \frac{n}{\delta}}{sn}}\right)$$

where $\mathfrak{W}_n = \max\left\{\mathfrak{R}_n^{buf}, \frac{2C_d^2 n \log(n/\delta)}{s}\right\}$ and C_d is the dependence of $\mathcal{R}_n(\mathcal{H})$ on the input dimensionality d.

The above bound guarantees an excess error of $\tilde{\mathcal{O}}(1/s)$ for algorithms (such as Follow-the-leader (Hazan et al., 2006)) that offer logarithmic regret $\mathfrak{R}_n^{\text{buf}} = \mathcal{O}(\log n)$. We stress that this theorem is not a direct corollary of our results for the *infinite buffer* case (Theorems 3 and 5). Instead, our proofs require a more careful analysis of the excess risk in order to accommodate the finiteness of the buffer and the randomness (possibly) used in constructing it.

More specifically, care needs to be taken to handle randomized buffer update policies such as **RS** which introduce additional randomness into the analysis. A naive application of techniques used to prove results for the unbounded buffer case would result in bounds that give non trivial generalization guarantees only for large buffer sizes such as $s = \omega(\sqrt{n})$. Our bounds, on the other hand, only require $s = \tilde{\omega}(1)$.

Key to our proofs is a conditioning step where we first analyze the conditional excess risk by conditioning upon randomness used by the buffer update policy. Such conditioning is made possible by the stream-oblivious nature of the update policy and thus, stream-obliviousness is required by our analysis. Subsequently, we analyze the excess risk by taking expectations over randomness used by the buffer update policy. The complete proofs of both parts of Theorem 6 are given in Appendix D.

Note that the above results only require an online learning algorithm to provide regret bounds w.r.t. the *finite-buffer* penalties $\hat{\mathcal{L}}_t^{\text{buf}}$ and do not require any regret bounds w.r.t the *all-pairs* penalties $\hat{\mathcal{L}}_t$.

For instance, the finite buffer based online learning algorithms OAM_{seq} and OAM_{gra} proposed in (Zhao et al., 2011) are able to provide a regret bound w.r.t. $\hat{\mathcal{L}}_t^{\text{buf}}$ (Zhao et al., 2011, Lemma 2) but are not able to do so w.r.t the *all-pairs* loss function (see Section 7 for a discussion). Using Theorem 6, we are able to give a generalization bound for OAM_{seq} and OAM_{gra} and hence explain the good empirical performance of these algorithms as reported in (Zhao et al., 2011). Note that Wang et al. (2013) are not able to analyze OAM_{seq} and OAM_{gra} since their analysis is restricted to algorithms that use the (deterministic) FIFO update policy whereas OAM_{seq} and OAM_{gra} use the (randomized) **RS** policy of Vitter (1985).

6. Applications

In this section we make explicit our online to batch conversion bounds for several learning scenarios and also demonstrate their dependence on input dimensionality by calculating their respective Rademacher complexities. Recall that our definition of Rademacher complexity for a pairwise function class is given by,

$$\mathcal{R}_n(\mathcal{H}) = \mathbb{E}\left[\!\!\left[\sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{\tau=1}^{n} \epsilon_\tau h(\mathbf{z}, \mathbf{z}_\tau)\right]\!\!\right].$$

For our purposes, we would be interested in the Rademacher complexities of *composition classes* of the form $\ell \circ \mathcal{H} := \{(\mathbf{z}, \mathbf{z}') \mapsto \ell(h, \mathbf{z}, \mathbf{z}'), h \in \mathcal{H}\}$ where ℓ is some Lipschitz loss function. Frequently we have $\ell(h, \mathbf{z}, \mathbf{z}') = \phi\left(h(\mathbf{x}, \mathbf{x}')Y(y, y')\right)$ where $Y(y, y') = y - y'$ or $Y(y, y') = yy'$ and $\phi : \mathbb{R} \to \mathbb{R}$ is some margin loss function (Steinwart & Christmann, 2008). Suppose ϕ is L-Lipschitz and $Y = \sup_{y,y'\in\mathcal{Y}} |Y(y, y')|$. Then we have

Theorem 7. $\mathcal{R}_n(\ell \circ \mathcal{H}) \leq LY\mathcal{R}_n(\mathcal{H})$.

The proof uses standard contraction inequalities and is given in Appendix E. This reduces our task to computing the values of $\mathcal{R}_n(\mathcal{H})$ which we do using a two stage proof technique (see Appendix F). For any subset X of a Banach space and any norm $\|\cdot\|_p$, we define $\|X\|_p := \sup_{\mathbf{x}\in X} \|\mathbf{x}\|_p$. Let the domain $\mathcal{X} \subset \mathbb{R}^d$.

AUC maximization (Zhao et al., 2011): the goal here is to maximize the area under the ROC curve for a linear classification problem where the hypothesis space $\mathcal{W} \subset \mathbb{R}^d$. We have $h_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') = \mathbf{w}^\top\mathbf{x} - \mathbf{w}^\top\mathbf{x}'$ and $\ell(h_{\mathbf{w}}, \mathbf{z}, \mathbf{z}') = \phi\left((y - y')h_{\mathbf{w}}(\mathbf{x}, \mathbf{x}')\right)$ where ϕ is the

hinge loss. In case our classifiers are L_p regularized for $p > 1$, we can show that $\mathcal{R}_n(\mathcal{W}) \leq 2 \|\mathcal{X}\|_q \|\mathcal{W}\|_p \sqrt{\frac{q-1}{n}}$ where $q = p/(p-1)$. Using the sparsity promoting L_1 regularizer gives us $\mathcal{R}_n(\mathcal{W}) \leq 2 \|\mathcal{X}\|_\infty \|\mathcal{W}\|_1 \sqrt{\frac{e \log d}{n}}$. Note that we obtain dimension independence, for example when the classifiers are L_2 regularized which allows us to bound the Rademacher complexities of kernelized function classes for bounded kernels as well.

Metric learning (Jin et al., 2009): the goal here is to learn a Mahalanobis metric $M_{\mathbf{W}}(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^\top \mathbf{W}(\mathbf{x} - \mathbf{x}')$ using the loss function $\ell(\mathbf{W}, \mathbf{z}, \mathbf{z}') = \phi\left(yy'\left(1 - M^2_{\mathbf{W}}(\mathbf{x}, \mathbf{x}')\right)\right)$ for a hypothesis class $\mathcal{W} \subset \mathbb{R}^{d \times d}$. In this case it is possible to use a variety of mixed norm $\|\cdot\|_{p,q}$ and Schatten norm $\|\cdot\|_{S(p)}$ regularizations on matrices in the hypothesis class. In case we use trace norm regularization on the matrix class, we get $\mathcal{R}_n(\mathcal{W}) \leq \|\mathcal{X}\|_2^2 \|\mathcal{W}\|_{S(1)} \sqrt{\frac{e \log d}{n}}$. The $(2,2)$-norm regularization offers a dimension independent bound $\mathcal{R}_n(\mathcal{W}) \leq \|\mathcal{X}\|_2^2 \|\mathcal{W}\|_{2,2} \sqrt{\frac{1}{n}}$. The mixed $(2,1)$-norm regularization offers $\mathcal{R}_n(\mathcal{W}) \leq \|\mathcal{X}\|_2 \|\mathcal{X}\|_\infty \|\mathcal{W}\|_{2,1} \sqrt{\frac{e \log d}{n}}$.

Multiple kernel learning (Kumar et al., 2012): the goal here is to improve the SVM classification algorithm by learning a *good* kernel K that is a positive combination of *base* kernels $K_1, \ldots, K_p$ i.e. $K_{\boldsymbol{\mu}}(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^p \boldsymbol{\mu}_i K_i(\mathbf{x}, \mathbf{x}')$ for some $\boldsymbol{\mu} \in \mathbb{R}^p, \boldsymbol{\mu} \geq 0$. The base kernels are bounded, i.e. for all i, $|K_i(\mathbf{x}, \mathbf{x}')| \leq \kappa^2$ for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ The notion of goodness used here is the one proposed by Balcan & Blum (2006) and involves using the loss function $\ell(\boldsymbol{\mu}, \mathbf{z}, \mathbf{z}') = \phi\left(yy' K_{\boldsymbol{\mu}}(\mathbf{x}, \mathbf{x}')\right)$ where $\phi(\cdot)$ is a margin loss function meant to encode some notion of alignment. The two hypothesis classes for the combination vector $\boldsymbol{\mu}$ that we study are the L_1 regularized unit simplex $\Delta(1) = \{\boldsymbol{\mu} : \|\boldsymbol{\mu}\|_1 = 1, \boldsymbol{\mu} \geq 0\}$ and the L_2 regularized unit sphere $\mathcal{S}_2(1) = \{\boldsymbol{\mu} : \|\boldsymbol{\mu}\|_2 = 1, \boldsymbol{\mu} \geq 0\}$. We are able to show the following Rademacher complexity bounds for these classes: $\mathcal{R}_n(\mathcal{S}_2(1)) \leq \kappa^2 \sqrt{\frac{p}{n}}$ and $\mathcal{R}_n(\Delta(1)) \leq \kappa^2 \sqrt{\frac{e \log p}{n}}$.

The details of the Rademacher complexity derivations for these problems and other examples such as similarity learning can be found in Appendix F.

7. OLP : Online Learning with Pairwise Loss Functions

In this section, we present an online learning algorithm for learning with pairwise loss functions in a finite buffer setting. The key contribution in this section is a buffer update policy that when combined with a variant of the GIGA algorithm (Zinkevich, 2003) allows us to give high probability regret bounds.

Algorithm 1 RS-x : Stream Subsampling with Replacement

Input: Buffer B, new point $\mathbf{z}_t$, buffer size s, timestep t.

```
if |B| < s then                                   //There is space
    B ← B ∪ {z_t}
else                                              //Overflow situation
    if t = s + 1 then                             //Repopulation step
        TMP ← B ∪ {z_t}
        Repopulate B with s points sampled uniformly with replacement from TMP.
    else                                          //Normal update step
        Independently, replace each point of B with z_t with probability 1/t.
    end if
end if
```

Algorithm 2 OLP : Online Learning with Pairwise Loss Functions

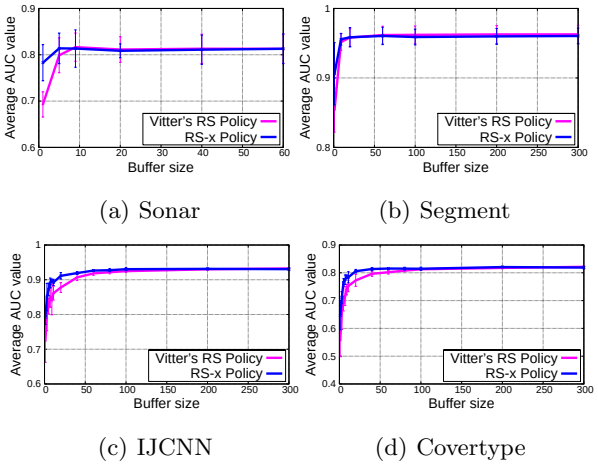
Input: Step length scale η, Buffer size s

Output: An ensemble $\mathbf{w}_2, \ldots, \mathbf{w}_n \in \mathcal{W}$ with low regret

```
w_0 ← 0, B ← φ
for t = 1 to n do
    Obtain a training point z_t
    Set step length η_t ← η/√t
    w_t ← Π_W [ w_{t-1} + (η_t/|B|) Σ_{z∈B} ∇_w ℓ(w_{t-1}, z_t, z) ]
                                                  //Π_W projects onto the set W
    B ← Update-buffer(B, z_t, s, t)               //using RS-x
end for
return w_2, ..., w_n
```

In previous work, Zhao et al. (2011) presented an online learning algorithm that uses finite buffers with the **RS** policy and proposed an *all-pairs* regret bound. The **RS** policy ensures, over the randomness used in buffer updates, that at any given time, the buffer contains a uniform sample from the preceding stream. Using this property, (Zhao et al., 2011, Lemma 2) claimed that $\mathbb{E}\left[\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1})\right]\!\right] = \hat{\mathcal{L}}_t(h_{t-1})$ where the expectation is taken over the randomness used in buffer construction. However, a property such as $\mathbb{E}\left[\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h)\right]\!\right] = \hat{\mathcal{L}}_t(h)$ holds only for functions h that are either fixed or obtained independently of the random variables used in buffer updates (over which the expectation is taken). Since h_{t-1} is learned from points in the buffer itself, the above property, and consequently the regret bound, does not hold.

We remedy this issue by showing a relatively weaker claim; we show that with high probability we have $\hat{\mathcal{L}}_t(h_{t-1}) \leq \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) + \epsilon$. At a high level, this claim is similar to showing uniform convergence bounds for $\hat{\mathcal{L}}_t^{\text{buf}}$. However, the reservoir sampling algorithm is not particularly well suited to prove such uniform conver-



(a) Sonar (b) Segment

(c) IJCNN (d) Covertype

Figure 1. Performance of **OLP** (using **RS-x**) and OAM_{gra} (using **RS**) by (Zhao et al., 2011) on AUC maximization tasks with varying buffer sizes.

gence bounds as it essentially performs sampling without replacement (see Appendix G for a discussion). We overcome this hurdle by proposing a new buffer update policy **RS-x** (see Algorithm 1) that, at each time step, guarantees s i.i.d. samples from the preceding stream (see Appendix H for a proof).

Our algorithm uses this buffer update policy in conjunction with an online learning algorithm **OLP** (see Algorithm 2) that is a variant of the well-known GIGA algorithm (Zinkevich, 2003). We provide the following *all-pairs* regret guarantee for our algorithm:

Theorem 8. *Suppose the **OLP** algorithm working with an s-sized buffer generates an ensemble $\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}$. Then with probability at least $1-\delta$,*

$$\frac{\mathfrak{R}_n}{n-1} \leq \mathcal{O}\left(C_d\sqrt{\frac{\log\frac{n}{\delta}}{s}} + \sqrt{\frac{1}{n-1}}\right)$$

See Appendix I for the proof. A drawback of our bound is that it offers sublinear regret only for buffer sizes $s = \omega(\log n)$. A better regret bound for constant s or a lower-bound on the regret is an open problem.

8. Experimental Evaluation

In this section we present experimental evaluation of our proposed **OLP** algorithm. We stress that the aim of this evaluation is to show that our algorithm, that enjoys high confidence regret bounds, also performs competitively in practice with respect to the OAM_{gra} algorithm proposed by Zhao et al. (2011) since our results in Section 5 show that OAM_{gra} does enjoy good generalization guarantees despite the lack of an *all-pairs* regret bound.

In our experiments, we adapted the **OLP** algorithm to the AUC maximization problem and compared it with OAM_{gra} on 18 different benchmark datasets. We used 60% of the available data points up to a maximum of 20000 points to train both algorithms. We refer the reader to Appendix J for a discussion on the implementation of the **RS-x** algorithm. Figure 1 presents the results of our experiments on 4 datasets across 5 random training/test splits. Results on other datasets can be found in Appendix K. The results demonstrate that **OLP** performs competitively to OAM_{gra} while in some cases having slightly better performance for small buffer sizes.

9. Conclusion

In this paper we studied the generalization capabilities of online learning algorithms for pairwise loss functions from several different perspectives. Using the method of *Symmetrization of Expectations*, we first provided sharp online to batch conversion bounds for algorithms that offer *all-pairs* regret bounds. Our results for bounded and strongly convex loss functions closely match their first order counterparts. We also extended our analysis to algorithms that are only able to provide *finite-buffer* regret bounds using which we were able to explain the good empirical performance of some existing algorithms. Finally we presented a new memory-efficient online learning algorithm that is able to provide *all-pairs* regret bounds in addition to performing well empirically.

Several interesting directions can be pursued for future work, foremost being the development of online learning algorithms that can guarantee sub-linear regret at constant buffer sizes or else a regret lower bound for finite buffer algorithms. Secondly, the idea of a *stream-aware* buffer update policy is especially interesting both from an empirical as well as theoretical point of view and would possibly require novel proof techniques for its analysis. Lastly, scalability issues that arise when working with higher order loss functions also pose an interesting challenge.

Acknowledgment

The authors thank the anonymous referees for comments that improved the presentation of the paper. PK is supported by the Microsoft Corporation and Microsoft Research India under a Microsoft Research India Ph.D. fellowship award.

References

Agarwal, Shivani and Niyogi, Partha. Generalization Bounds for Ranking Algorithms via Algorithmic Stability. *JMLR*, 10:441–474, 2009.

Balcan, Maria-Florina and Blum, Avrim. On a Theory of Learning with Similarity Functions. In *ICML*, pp. 73–80, 2006.

Bellet, Aurélien, Habrard, Amaury, and Sebban, Marc. Similarity Learning for Provably Accurate Sparse Linear Classification. In *ICML*, 2012.

Brefeld, Ulf and Scheffer, Tobias. AUC Maximizing Support Vector Learning. In *ICML workshop on ROC Analysis in Machine Learning*, 2005.

Cao, Qiong, Guo, Zheng-Chu, and Ying, Yiming. Generalization Bounds for Metric and Similarity Learning, 2012. arXiv:1207.5437.

Cesa-Bianchi, Nicoló and Gentile, Claudio. Improved Risk Tail Bounds for On-Line Algorithms. *IEEE Trans. on Inf. Theory*, 54(1):286–390, 2008.

Cesa-Bianchi, Nicoló, Conconi, Alex, and Gentile, Claudio. On the Generalization Ability of On-Line Learning Algorithms. In *NIPS*, pp. 359–366, 2001.

Clémençon, Stéphan, Lugosi, Gábor, and Vayatis, Nicolas. Ranking and empirical minimization of U-statistics. *Annals of Statistics*, 36:844–874, 2008.

Cortes, Corinna, Mohri, Mehryar, and Rostamizadeh, Afshin. Generalization Bounds for Learning Kernels. In *ICML*, pp. 247–254, 2010a.

Cortes, Corinna, Mohri, Mehryar, and Rostamizadeh, Afshin. Two-Stage Learning Kernel Algorithms. In *ICML*, pp. 239–246, 2010b.

Cristianini, Nello, Shawe-Taylor, John, Elisseeff, André, and Kandola, Jaz S. On Kernel-Target Alignment. In *NIPS*, pp. 367–373, 2001.

Freedman, David A. On Tail Probabilities for Martingales. *Annals of Probability*, 3(1):100–118, 1975.

Hazan, Elad, Kalai, Adam, Kale, Satyen, and Agarwal, Amit. Logarithmic Regret Algorithms for Online Convex Optimization. In *COLT*, pp. 499–513, 2006.

Jin, Rong, Wang, Shijun, and Zhou, Yang. Regularized Distance Metric Learning: Theory and Algorithm. In *NIPS*, pp. 862–870, 2009.

Kakade, Sham M. and Tewari, Ambuj. On the Generalization Ability of Online Strongly Convex Programming Algorithms. In *NIPS*, pp. 801–808, 2008.

Kakade, Sham M., Sridharan, Karthik, and Tewari, Ambuj. On the Complexity of Linear Prediction: Risk Bounds, Margin Bounds, and Regularization. In *NIPS*, 2008.

Kakade, Sham M., Shalev-Shwartz, Shai, and Tewari, Ambuj. Regularization Techniques for Learning with Matrices. *JMLR*, 13:1865–1890, 2012.

Kumar, Abhishek, Niculescu-Mizil, Alexandru, Kavukcuoglu, Koray, and III, Hal Daumé. A Binary Classification Framework for Two-Stage Multiple Kernel Learning. In *ICML*, 2012.

Ledoux, Michel and Talagrand, Michel. *Probability in Banach Spaces: Isoperimetry and Processes*. Springer, 2002.

Sridharan, Karthik, Shalev-Shwartz, Shai, and Srebro, Nathan. Fast Rates for Regularized Objectives. In *NIPS*, pp. 1545–1552, 2008.

Steinwart, Ingo and Christmann, Andreas. *Support Vector Machines*. Information Science and Statistics. Springer, 2008.

Vitter, Jeffrey Scott. Random Sampling with a Reservoir. *ACM Trans. on Math. Soft.*, 11(1):37–57, 1985.

Wang, Yuyang, Khardon, Roni, Pechyony, Dmitry, and Jones, Rosie. Generalization Bounds for Online Learning Algorithms with Pairwise Loss Functions. *JMLR - Proceedings Track*, 23:13.1–13.22, 2012.

Wang, Yuyang, Khardon, Roni, Pechyony, Dmitry, and Jones, Rosie. Online Learning with Pairwise Loss Functions, 2013. arXiv:1301.5332.

Xing, Eric P., Ng, Andrew Y., Jordan, Michael I., and Russell, Stuart J. Distance Metric Learning with Application to Clustering with Side-Information. In *NIPS*, pp. 505–512, 2002.

Zhao, Peilin, Hoi, Steven C. H., Jin, Rong, and Yang, Tianbao. Online AUC Maximization. In *ICML*, pp. 233–240, 2011.

Zinkevich, Martin. Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In *ICML*, pp. 928–936, 2003.

A. Proof of Lemma 1

Lemma 9 (Lemma 1 restated)**.** *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a bounded loss function $\ell : \mathcal{H} \times \mathcal{Z} \times \mathcal{Z} \to [0, B]$. Then for any $\delta > 0$, we have with probability at least $1 - \delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \frac{1}{n-1}\sum_{t=2}^{n}\hat{\mathcal{L}}_t(h_{t-1}) + \frac{2}{n-1}\sum_{t=2}^{n}\mathcal{R}_{t-1}(\ell\circ\mathcal{H}) + 3B\sqrt{\frac{\log\frac{n}{\delta}}{n-1}}.$$

Proof. As a first step, we decompose the excess risk in a manner similar to (Wang et al., 2012). For any $h \in \mathcal{H}$ let

$$\tilde{\mathcal{L}}_t(h) := \mathop{\mathbb{E}}_{\mathbf{z}_t}\left[\!\!\left[\hat{\mathcal{L}}_t(h)\middle| Z^{t-1}\right]\!\!\right].$$

This allows us to decompose the excess risk as follows:

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) - \hat{\mathcal{L}}_t(h_{t-1}) = \frac{1}{n-1}\left(\sum_{t=2}^{n}\underbrace{\mathcal{L}(h_{t-1}) - \tilde{\mathcal{L}}_t(h_{t-1})}_{P_t} + \underbrace{\tilde{\mathcal{L}}(h_{t-1}) - \hat{\mathcal{L}}_t(h_{t-1})}_{Q_t}\right).$$

By construction, we have $\mathop{\mathbb{E}}_{\mathbf{z}_t}\left[\!\!\left[Q_t\middle| Z^{t-1}\right]\!\!\right] = 0$ and hence the sequence $Q_2, \ldots, Q_n$ forms a martingale difference sequence. Since $|Q_t| \leq B$ as the loss function is bounded, an application of the Azuma-Hoeffding inequality shows that with probability at least $1 - \delta$

$$\frac{1}{n-1}\sum_{t=2}^{n}Q_t \leq B\sqrt{\frac{2\log\frac{1}{\delta}}{n-1}}. \tag{4}$$

We now analyze each term P_t individually. By linearity of expectation, we have for a ghost sample $\tilde{Z}^{t-1} = \{\tilde{\mathbf{z}}_1, \ldots, \tilde{\mathbf{z}}_{t-1}\}$,

$$\mathcal{L}(h_{t-1}) = \mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\!\left[\frac{1}{t-1}\sum_{\tau=1}^{t-1}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\!\left[\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}_\tau)\right]\!\!\right]\right]\!\!\right]. \tag{5}$$

The expression of $\mathcal{L}(h_{t-1})$ as a nested expectation is the precursor to performing symmetrization with expectations and plays a crucial role in overcoming coupling problems. This allows us to write P_t as

$$P_t = \mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\!\left[\frac{1}{t-1}\sum_{\tau=1}^{t-1}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\!\left[\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}_\tau)\right]\!\!\right]\right]\!\!\right] - \tilde{\mathcal{L}}_t(h_{t-1}) \leq \underbrace{\sup_{h\in\mathcal{H}}\left[\mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\!\left[\frac{1}{t-1}\sum_{\tau=1}^{t-1}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\!\left[\ell(h, \mathbf{z}, \tilde{\mathbf{z}}_\tau)\right]\!\!\right]\right]\!\!\right] - \tilde{\mathcal{L}}_t(h)\right]}_{g_t(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1})}.$$

Since $\tilde{\mathcal{L}}_t(h) = \mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\!\left[\frac{1}{t-1}\sum_{\tau=1}^{t-1}\ell(h, \mathbf{z}, \mathbf{z}_\tau)\middle| Z^{t-1}\right]\!\!\right]$ and ℓ is bounded, the expression $g_t(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1})$ can have a variation of at most $B/(t-1)$ when changing any of its $(t-1)$ variables. Hence an application of McDiarmid's inequality gives us, with probability at least $1 - \delta$,

$$g_t(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1}) \leq \mathop{\mathbb{E}}_{Z^{t-1}}\left[\!\!\left[g_t(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1})\right]\!\!\right] + B\sqrt{\frac{\log\frac{1}{\delta}}{2(t-1)}}.$$

For any $h \in \mathcal{H}, \mathbf{z}' \in \mathbb{Z}$, let $\wp(h, \mathbf{z}') := \frac{1}{t-1}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\!\left[\ell(h, \mathbf{z}, \mathbf{z}')\right]\!\!\right]$. Then we can write $\mathop{\mathbb{E}}_{Z^{t-1}}\left[\!\!\left[g(\mathbf{z}_1, \ldots, \mathbf{z}_{t-1})\right]\!\!\right]$ as

$$\begin{aligned}
&\mathop{\mathbb{E}}_{Z^{t-1}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\!\left[\sum_{\tau=1}^{t-1}\wp(h, \tilde{\mathbf{z}}_\tau)\right]\!\!\right] - \sum_{\tau=1}^{t-1}\wp(h, \mathbf{z}_\tau)\right]\right]\!\!\right] \\
&\leq \mathop{\mathbb{E}}_{Z^{t-1}, \tilde{Z}^{t-1}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\sum_{\tau=1}^{t-1}\wp(h, \tilde{\mathbf{z}}_\tau) - \sum_{\tau=1}^{t-1}\wp(h, \mathbf{z}_\tau)\right]\right]\!\!\right] \\
&= \mathop{\mathbb{E}}_{Z^{t-1}, \tilde{Z}^{t-1}, \{\epsilon_\tau\}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\sum_{\tau=1}^{t-1}\epsilon_\tau\left(\wp(h, \tilde{\mathbf{z}}_\tau) - \wp(h, \mathbf{z}_\tau)\right)\right]\right]\!\!\right] \\
&\leq \frac{2}{t-1}\mathop{\mathbb{E}}_{Z^{t-1}, \{\epsilon_\tau\}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\sum_{\tau=1}^{t-1}\epsilon_\tau\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\!\left[\ell(h, \mathbf{z}, \mathbf{z}_\tau)\right]\!\!\right]\right]\right]\!\!\right] \\
&\leq \frac{2}{t-1}\mathop{\mathbb{E}}_{\mathbf{z}, Z^{t-1}, \{\epsilon_\tau\}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\sum_{\tau=1}^{t-1}\epsilon_\tau\ell(h, \mathbf{z}, \mathbf{z}_\tau)\right]\right]\!\!\right] \\
&= 2\mathcal{R}_{t-1}(\ell\circ\mathcal{H}).
\end{aligned}$$

Note that in the third step, the symmetrization was made possible by the decoupling step in Eq. (5) where we decoupled the "head" variable $\mathbf{z}_t$ from the "tail" variables by absorbing it inside an expectation. This allowed us to symmetrize the true and ghost samples $\mathbf{z}_\tau$ and $\tilde{\mathbf{z}}_\tau$ in a standard manner. Thus we have, with probability at least $1 - \delta$,

$$P_t \leq 2\mathcal{R}_{t-1}(\ell\circ\mathcal{H}) + B\sqrt{\frac{\log\frac{1}{\delta}}{2(t-1)}}.$$

Applying a union bound on the bounds for $P_t, t = 2, \ldots, n$ gives us with probability at least $1 - \delta$,

$$\frac{1}{n-1}\sum_{t=2}^{n}P_t \leq \frac{2}{n-1}\sum_{t=2}^{n}\mathcal{R}_{t-1}(\ell\circ\mathcal{H}) + B\sqrt{\frac{2\log\frac{n}{\delta}}{n-1}}. \tag{6}$$

Adding Equations (4) and (6) gives us the result. □

B. Proof of Theorem 4

Theorem 10 (Theorem 4 restated)**.** *Let $\mathcal{F}$ be a closed and convex set of functions over $\mathcal{X}$. Let $\wp(f, \mathbf{x}) = p(\langle f, \phi(\mathbf{x})\rangle) + r(f)$, for a σ-strongly convex function*

r, be a loss function with $\mathcal{P}$ and $\hat{\mathcal{P}}$ as the associated population and empirical risk functionals and f^* as the population risk minimizer. Suppose $\wp$ is L-Lipschitz and $\|\phi(\mathbf{x})\|_* \leq R, \forall \mathbf{x} \in \mathcal{X}$. Then w.p. $1-\delta$, for any $\epsilon > 0$, we have for all $f \in \mathcal{F}$,

$$\mathcal{P}(f) - \mathcal{P}(f^*) \leq (1+\epsilon)\left(\hat{\mathcal{P}}(f) - \hat{\mathcal{P}}(f^*)\right) + \frac{C_\delta}{\epsilon\sigma n}$$

where $C_\delta = C_d^2 \cdot (4(1+\epsilon)LR)^2 (32 + \log(1/\delta))$ and C_d is the dependence of the Rademacher complexity of the class $\mathcal{F}$ on the input dimensionality d.

Proof. We begin with a lemma implicit in the proof of Theorem 1 in (Sridharan et al., 2008). For the function class $\mathcal{F}$ and loss function $\wp$ as above, define a new loss function $\mu : (f, \mathbf{x}) \mapsto \wp(f, \mathbf{x}) - \wp(f^*, \mathbf{x})$ with $\mathcal{M}$ and $\hat{\mathcal{M}}$ as the associated population and empirical risk functionals. Let $r = \frac{4L^2R^2C_d^2(32+\log(1/\delta))}{\sigma n}$. Then we have the following

Lemma 11. *For any $\epsilon > 0$, with probability at least $1-\delta$, the following happens*

1. *For all $f \in \mathcal{F}$ such that $\mathcal{M}(f) \leq 16\left(1+\frac{1}{\epsilon}\right)^2 r$, we have $\mathcal{M}(f) \leq \hat{\mathcal{M}}(f) + 4\left(1+\frac{1}{\epsilon}\right) r$.*
2. *For all $f \in \mathcal{F}$ such that $\mathcal{M}(f) > 16\left(1+\frac{1}{\epsilon}\right)^2 r$, we have $\mathcal{M}(f) \leq (1+\epsilon)\hat{\mathcal{M}}(f)$.*

The difference in our proof technique lies in the way we combine these two cases. We do so by proving the following two simple results.

Lemma 12. *For all f s.t. $\mathcal{M}(f) \leq 16\left(1+\frac{1}{\epsilon}\right)^2 r$, we have $\mathcal{M}(f) \leq (1+\epsilon)\left(\hat{\mathcal{M}}(f) + 4\left(1+\frac{1}{\epsilon}\right) r\right)$.*

Proof. We notice that for all $f \in \mathcal{F}$, we have $\mathcal{M}(f) = \mathcal{P}(f) - \mathcal{P}(f^*) \geq 0$. Thus, using Lemma 11, Part 1, we have $\hat{\mathcal{M}}(f) + 4\left(1+\frac{1}{\epsilon}\right) r \geq \mathcal{M}(f) \geq 0$. Since for any $a, \epsilon > 0$, we have $a \leq (1+\epsilon)a$, the result follows. □

Lemma 13. *For all f s.t. $\mathcal{M}(f) > 16\left(1+\frac{1}{\epsilon}\right)^2 r$, we have $\mathcal{M}(f) \leq (1+\epsilon)\left(\hat{\mathcal{M}}(f) + 4\left(1+\frac{1}{\epsilon}\right) r\right)$.*

Proof. We use the fact that $r > 0$ and thus $4(1+\epsilon)\left(1+\frac{1}{\epsilon}\right) r > 0$ as well. The result then follows from an application of Part 2 of Lemma 11. □

From the definition of the loss function μ, we have for any $f \in \mathcal{F}$, $\mathcal{M}(f) = \mathcal{P}(f) - \mathcal{P}(f^*)$ and $\hat{\mathcal{M}}(f) = \hat{\mathcal{P}}(f) - \hat{\mathcal{P}}(f^*)$. Combining the above lemmata with this observation completes the proof. □

C. Proof of Theorem 5

Theorem 14 (Theorem 5 restated)**.** *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a B-bounded, L-Lipschitz and σ-strongly convex loss function ℓ. Further suppose the learning algorithm guarantees a regret bound of $\mathfrak{R}_n$. Let $\mathfrak{V}_n = \max\left\{\mathfrak{R}_n, 2C_d^2 \log n \log(n/\delta)\right\}$ Then for any $\delta > 0$, we have with probability at least $1-\delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{\mathfrak{R}_n}{n-1} + C_d \cdot \mathcal{O}\left(\frac{\sqrt{\mathfrak{V}_n \log n \log(n/\delta)}}{n-1}\right),$$

where the $\mathcal{O}(\cdot)$ notation hides constants dependent on domain size and the loss function such as L, B and σ.

Proof. The decomposition of the excess risk shall not be made explicitly in this case but shall emerge as a side-effect of the proof progression. Consider the loss function $\wp(h, \mathbf{z}') := \mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h, \mathbf{z}, \mathbf{z}')]\!]$ with $\mathcal{P}$ and $\hat{\mathcal{P}}$ as the associated population and empirical risk functionals. Clearly, if ℓ is L-Lipschitz and σ-strongly convex then so is $\wp$. As Equation (5) shows, for any $h \in \mathcal{H}$, $\mathcal{P}(h) = \mathcal{L}(h)$. Also it is easy to see that for any Z^{t-1}, $\hat{\mathcal{P}}(h) = \tilde{\mathcal{L}}_t(h)$. Applying Theorem 4 on h_{t-1} with the loss function $\wp$ gives us w.p. $1-\delta$,

$$\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*) \leq (1+\epsilon)\left(\tilde{\mathcal{L}}_t(h_{t-1}) - \tilde{\mathcal{L}}_t(h^*)\right) + \frac{C_\delta}{\epsilon\sigma(t-1)}$$

which, upon summing across time steps and taking a union bound, gives us with probability at least $1-\delta$,

$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{C_{(\delta/n)} \log n}{\epsilon\sigma(n-1)} + \frac{1+\epsilon}{n-1}\sum_{t=2}^{n}\left(\tilde{\mathcal{L}}_t(h_{t-1}) - \tilde{\mathcal{L}}_t(h^*)\right).$$

Let $\xi_t := \left(\tilde{\mathcal{L}}_t(h_{t-1}) - \tilde{\mathcal{L}}_t(h^*)\right) - \left(\hat{\mathcal{L}}_t(h_{t-1}) - \hat{\mathcal{L}}_t(h^*)\right)$. Then using the regret bound $\mathfrak{R}_n$ we can write,

$$\frac{1}{n-1}\sum_{t=2}^{n} \mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{1+\epsilon}{n-1}\left(\mathfrak{R}_n + \sum_{t=2}^{n}\xi_t\right) + \frac{C_{(\delta/n)} \log n}{\epsilon\sigma(n-1)}.$$

We now use Bernstein type inequalities to bound the sum $\sum_{t=2}^{n}\xi_t$ using a proof technique used in (Kakade & Tewari, 2008; Cesa-Bianchi & Gentile, 2008). We first note some properties of the sequence below.

Lemma 15. *The sequence $\xi_2, \ldots, \xi_n$ is a bounded martingale difference sequence with bounded conditional variance.*

Proof. That ξ_t is a martingale difference sequence follows by construction: we can decompose the term $\xi_t = \phi_t - \psi_t$ where $\phi_t = \tilde{\mathcal{L}}_t(h_{t-1}) - \hat{\mathcal{L}}_t(h_{t-1})$ and $\psi_t = \tilde{\mathcal{L}}_t(h^*) - \hat{\mathcal{L}}_t(h^*)$, both of which are martingale difference sequences with respect to the common filtration $\mathcal{F} = \{\mathcal{F}_n : n = 0, 1, \ldots\}$ where $\mathcal{F}_n = \sigma(\mathbf{z}_i : i = 1, \ldots, n)$.

Since the loss function takes values in $[0, B]$, we have $|\xi_t| \leq 2B$ which proves that our sequence is bounded.

To prove variance bounds for the sequence, we first use the Lipschitz properties of the loss function to get

$$\begin{aligned}\xi_t &= \left(\tilde{\mathcal{L}}_t(h_{t-1}) - \tilde{\mathcal{L}}_t(h^*)\right) - \left(\hat{\mathcal{L}}_t(h_{t-1}) - \hat{\mathcal{L}}_t(h^*)\right) \\ &\leq 2L \left\| h_{t-1} - h^* \right\|.\end{aligned}$$

Recall that the hypothesis space is embedded in a Banach space equipped with the norm $\|\cdot\|$. Thus we have $\mathbb{E}\left[\!\left[\xi_t^2 \middle| Z^{t-1} \right]\!\right] \leq 4L^2 \left\| h_{t-1} - h^* \right\|^2$. Now using σ-strong convexity of the loss function we have

$$\begin{aligned}\frac{\mathcal{L}(h_{t-1}) + \mathcal{L}(h^*)}{2} &\geq \mathcal{L}\left(\frac{h_{t-1} + h^*}{2}\right) + \frac{\sigma}{8} \left\| h_{t-1} - h^* \right\|^2 \\ &\geq \mathcal{L}(h^*) + \frac{\sigma}{8} \left\| h_{t-1} - h^* \right\|^2.\end{aligned}$$

Let $\sigma_t^2 := \frac{16L^2}{\sigma} \left(\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*)\right)$. Combining the two inequalities we get $\mathbb{E}\left[\!\left[\xi_t^2 \middle| Z^{t-1} \right]\!\right] \leq \sigma_t^2$. □

We note that although (Kakade & Tewari, 2008) state their result with a requirement that the loss function be strongly convex in a point wise manner, i.e., for all $\mathbf{z}, \mathbf{z}' \in \mathcal{Z}$, the function $\ell(h, \mathbf{z}, \mathbf{z}')$ be strongly convex in h, they only require the result in expectation. More specifically, our notion of strong convexity where we require the population risk functional $\mathcal{L}(h)$ to be strongly convex actually suits the proof of (Kakade & Tewari, 2008) as well.

We now use a Bernstein type inequality for martingales proved in (Kakade & Tewari, 2008). The proof is based on a fundamental result on martingale convergence due to Freedman (1975).

Theorem 16. *Given a martingale difference sequence $X_t, t = 1 \ldots n$ that is uniformly B-bounded and has conditional variance $\mathbb{E}\left[\!\left[X_t^2 | X_1, \ldots, X_{t-1} \right]\!\right] \leq \sigma_t^2$, we have for any $\delta < 1/e$ and $n \geq 3$, with probability at least $1 - \delta$,*

$$\sum_{t=1}^{n} X_t \leq \max\left\{ 2\sigma^*, 3B\sqrt{\log\frac{4\log n}{\delta}} \right\} \sqrt{\log\frac{4\log n}{\delta}},$$

where $\sigma^* = \sqrt{\sum_{t=1}^{n} \sigma_t^2}$.

Let $\mathfrak{D}_n = \sum_{t=2}^{n} \left(\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*)\right)$. Then we can write the variance bound as

$$\begin{aligned}\sigma^* &= \sqrt{\sum_{t=1}^{n} \sigma_t^2} = \sqrt{\sum_{t=1}^{n} \frac{16L^2}{\sigma} \left(\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*)\right)} \\ &= 4L\sqrt{\frac{\mathfrak{D}_n}{\sigma}}.\end{aligned}$$

Thus, with probability at least $1 - \delta$, we have

$$\sum_{t=1}^{n} \xi_t \leq \max\left\{ 8L\sqrt{\frac{\mathfrak{D}_n}{\sigma}}, 6B\sqrt{\log\frac{4\log n}{\delta}} \right\} \sqrt{\log\frac{4\log n}{\delta}}.$$

Denoting $\Delta = \sqrt{\log\frac{4\log n}{\delta}}$ for notational simplicity and using the above bound in the online to batch conversion bound gives us

$$\begin{aligned}\frac{\mathfrak{D}_n}{n-1} \leq \frac{1+\epsilon}{n-1} &\left(\mathfrak{R}_n + \max\left\{ 8L\sqrt{\frac{\mathfrak{D}_n}{\sigma}}, 6B\Delta \right\} \Delta \right) \\ &+ \frac{C_{(\delta/n)} \log n}{\epsilon\sigma(n-1)}.\end{aligned}$$

Solving this quadratic inequality is simplified by a useful result given in (Kakade & Tewari, 2008, Lemma 4)

Lemma 17. *For any $s, r, d, b, \Delta > 0$ such that*

$$s \leq r + \max\left\{ 4\sqrt{ds}, 6b\Delta \right\} \Delta,$$

we also have

$$s \leq r + 4\sqrt{dr}\Delta + \max\{16d, 6b\}\, \Delta^2.$$

Using this result gives us a rather nasty looking expression which we simplify by absorbing constants inside the $\mathcal{O}(\cdot)$ notation. We also make a simplifying ad-hoc assumption that we shall only set $\epsilon \in (0, 1]$. The resulting expression is given below:

$$\begin{aligned}\mathfrak{D}_n \leq\ & (1+\epsilon)\, \mathfrak{R}_n + \mathcal{O}\left(\frac{C_d^2 \log n \log(n/\delta)}{\epsilon} + \log\frac{\log n}{\delta} \right) \\ &+ \mathcal{O}\left(\sqrt{\left(\mathfrak{R}_n + \frac{C_d^2 \log n \log(n/\delta)}{\epsilon} \right) \log\frac{\log n}{\delta}} \right).\end{aligned}$$

Let $\mathfrak{V}_n = \max\left\{ \mathfrak{R}_n, 2C_d^2 \log n \log(n/\delta) \right\}$. Concentrating only on the portion of the expression involving ϵ and ignoring the constants, we get

$$\begin{aligned}&\epsilon\mathfrak{R}_n + \frac{C_d^2 \log n \log(n/\delta)}{\epsilon} + \sqrt{\frac{C_d^2 \log n \log(n/\delta)}{\epsilon} \log\frac{\log n}{\delta}} \\ &\leq \epsilon\mathfrak{R}_n + \frac{2C_d^2 \log n \log(n/\delta)}{\epsilon} \leq \epsilon\mathfrak{V}_n + \frac{2C_d^2 \log n \log(n/\delta)}{\epsilon} \\ &\leq 2C_d\sqrt{2\mathfrak{V}_n \log n \log(n/\delta)},\end{aligned}$$

where the second step follows since $\epsilon \leq 1$ and the fourth step follows by using $\epsilon = \sqrt{\frac{2C_d^2 \log n \log(n/\delta)}{\mathfrak{V}_n}} \leq 1$. Putting this into the excess risk expression gives us

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{\mathfrak{R}_n}{n-1} + C_d \cdot \mathcal{O}\left(\frac{\sqrt{\mathfrak{V}_n \log n \log(n/\delta)}}{n-1}\right)$$

which finishes the proof. □

D. Generalization Bounds for Finite Buffer Algorithms

In this section we present online to batch conversion bounds for learning algorithms that work with finite buffers and are able to provide regret bounds $\mathfrak{R}_n^{\text{buf}}$ with respect to *finite-buffer* loss functions $\hat{\mathcal{L}}_t^{\text{buf}}$.

Although due to lack of space, Theorem 6 presents these bounds for bounded as well as strongly convex functions together, we prove them separately for sake of clarity. Moreover, the techniques used to prove these two results are fairly different which further motivates this. Before we begin, we present the problem setup formally and introduce necessary notation.

In our finite buffer online learning model, one observes a stream of elements $\mathbf{z}_1, \ldots, \mathbf{z}_n$. A *sketch* of these elements is maintained in a buffer B of size s, i.e., at each step $t = 2, \ldots, n$, the buffer contains a subset of the elements Z^{t-1} of size at most s. At each step $t = 2 \ldots n$, the online learning algorithm posits a hypothesis $h_{t-1} \in \mathcal{H}$, upon which the element $\mathbf{z}_t$ is revealed and the algorithm incurs the loss

$$\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) = \frac{1}{|B_t|}\sum_{\mathbf{z}\in B_t}\ell(h_{t-1}, \mathbf{z}_t, \mathbf{z}),$$

where B_t is the state of the buffer at time t. Note that $|B_t| \leq s$. We would be interested in algorithms that are able to give a *finite-buffer* regret bound, i.e., for which, the proposed ensemble $h_1, \ldots, h_{n-1}$ satisfies

$$\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \inf_{h\in\mathcal{H}}\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{\text{buf}}(h) \leq \mathfrak{R}_n^{\text{buf}}.$$

We assume that the buffer is updated after each step in a *stream-oblivious* manner. For randomized buffer update policies (such as reservoir sampling (Vitter, 1985)), we assume that we are supplied at each step with some fresh randomness r_t (see examples below) along with the data point $\mathbf{z}_t$. Thus the data received at time t is a tuple $\mathbf{w}_t = (\mathbf{z}_t, r_t)$. We shall refer to the random variables r_t as *auxiliary* variables. It is important to note that stream obliviousness dictates that r_t as a random variable is independent of z_t. Let $W^{t-1} := \{\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}\}$ and $R^{t-1} := \{r_1, \ldots, r_{t-1}\}$. Note that R^{t-1} completely decides the indices present in the buffer B_t at step t independent of Z^{t-1}. For any $h \in \mathcal{H}$, define

$$\tilde{\mathcal{L}}_t^{\text{buf}} := \mathop{\mathbb{E}}_{\mathbf{z}_t}\left[\!\left[\hat{\mathcal{L}}_t^{\text{buf}} \middle| W^{t-1}\right]\!\right].$$

D.1. Examples of Stream Oblivious Policies

Below we give some examples of stream oblivious policies for updating the buffer:

1. **FIFO**: in this policy, the data point $\mathbf{z}_t$ arriving at time $t > s$ is inducted into the buffer by evicting the data point $\mathbf{z}_{(t-s)}$ from the buffer. Since this is a non-randomized policy, there is no need for auxiliary randomness and we can assume that r_t follows the trivial law $r_t \sim \mathbb{1}_{\{r=1\}}$.
2. **RS** : the Reservoir Sampling policy was introduced by Vitter (1985). In this policy, at time $t > s$, the incoming data point $\mathbf{z}_t$ is inducted into the buffer with probability s/t. If chosen to be induced, it results in the eviction of a random element of the buffer. In this case the auxiliary random variable is 2-tuple that follows the law
$$r_t = (r_t^1, r_t^2) \sim \left(\text{Bernoulli}\left(\frac{s}{t}\right), \frac{1}{s}\sum_{i=1}^{s}\mathbb{1}_{\{r_2=i\}}\right).$$
3. **RS-x** (see Algorithm 1): in this policy, the incoming data point $\mathbf{z}_t$ at time $t > s$, replaces each data point in the buffer independently with probability $1/t$. Thus the incoming point has the potential to evict multiple buffer points while establishing multiple copies of itself in the buffer. In this case, the auxiliary random variable is defined by a Bernoulli process: $r_t = (r_t^1, r_t^2 \ldots, r_t^s) \sim \left(\text{Bernoulli}\left(\frac{1}{t}\right), \text{Bernoulli}\left(\frac{1}{t}\right), \ldots, \text{Bernoulli}\left(\frac{1}{t}\right)\right)$.
4. **RS-x^2** (see Algorithm 3): this is a variant of **RS-x** in which the number of evictions is first decided by a Binomial trial and then those many random points in the buffer are replaced by the incoming data point. This can be implemented as follows: $r_t = (r_t^1, r_t^2) \sim \left(\text{Binomial}\left(s, \frac{1}{t}\right), \text{Perm}(s)\right)$ where $\text{Perm}(s)$ gives a random permutation of s elements.

D.2. Finite Buffer Algorithms with Bounded Loss Functions

We shall prove the result in two steps. In the first step we shall prove the following uniform convergence style

result

Lemma 18. *Let $h_1, \ldots, h_{n-1}$ be an ensemble of hypotheses generated by an online learning algorithm working with a B-bounded loss function ℓ and a finite buffer of capacity s. Then for any $\delta > 0$, we have with probability at least $1-\delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \frac{1}{n-1}\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{buf}(h_{t-1}) + B\sqrt{\frac{2\log\frac{n}{\delta}}{s}} + \frac{2}{n-1}\sum_{t=2}^{n}\mathcal{R}_{\min\{t-1,s\}}(\ell\circ\mathcal{H}).$$

At a high level, our proof progression shall follow that of Lemma 1. However, the execution of the proof will have to be different in order to accommodate the finiteness of the buffer and randomness used to construct it. Similarly, we shall also be able to show the following result.

Lemma 19. *For any $\delta > 0$, we have with probability at least $1-\delta$,*

$$\frac{1}{n-1}\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{buf}(h^*) \leq \mathcal{L}(h^*) + 3B\sqrt{\frac{\log\frac{n}{\delta}}{s}} + \frac{2}{n-1}\sum_{t=2}^{n}\mathcal{R}_{\min\{t-1,s\}}(\ell\circ\mathcal{H}).$$

Note that for classes whose Rademacher averages behave as $\mathcal{R}_n(\mathcal{H}) \leq C_d\cdot\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$, applying Lemma 7 gives us $\mathcal{R}_n(\ell\circ\mathcal{H}) \leq C_d\cdot\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$ as well which allows us to show

$$\frac{2}{n-1}\sum_{t=2}^{n}\mathcal{R}_{\min\{t-1,s\}}(\ell\circ\mathcal{H}) = C_d\cdot\mathcal{O}\left(\frac{1}{\sqrt{s}}\right).$$

Combining Lemmata 18 and 19 along with the definition of bounded buffer regret $\mathfrak{R}_n^{\text{buf}}$ gives us the first part of Theorem 6. We prove Lemma 18 below:

Proof (of Lemma 18). We first decompose the excess risk term as before

$$\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) - \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) = \sum_{t=2}^{n}\underbrace{\mathcal{L}(h_{t-1}) - \tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1})}_{P_t} + \underbrace{\tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1})}_{Q_t}.$$

By construction, the sequence Q_t forms a martingale difference sequence, i.e., $\mathop{\mathbb{E}}_{\mathbf{z}_t}\left[\!\left[Q_t \middle| Z^{t-1}\right]\!\right] = 0$ and hence by an application of Azuma Hoeffding inequality we have

$$\frac{1}{n-1}\sum_{t=2}^{n}Q_t \leq B\sqrt{\frac{2\log\frac{1}{\delta}}{n-1}}. \tag{7}$$

We now analyze each term P_t individually. To simplify the analysis a bit we assume that the buffer update policy keeps admitting points into the buffer as long as there is space so that for $t \leq s+1$, the buffer contains an exact copy of the preceding stream. This is a very natural assumption satisfied by FIFO as well as reservoir sampling. We stress that our analysis works even without this assumption but requires a bit more work. In case we do make this assumption, the analysis of Lemma 1 applies directly and we have, for any $t \leq s+1$, with probability at least $1-\delta$,

$$P_t \leq \mathcal{R}_{t-1}(\ell\circ\mathcal{H}) + B\sqrt{\frac{\log\frac{1}{\delta}}{2(t-1)}}$$

For $t > s+1$, for an independent ghost sample $\{\tilde{\mathbf{w}}_1, \ldots, \tilde{\mathbf{w}}_{t-1}\}$ we have,

$$\mathop{\mathbb{E}}_{\tilde{W}^{t-1}}\left[\!\left[\tilde{\mathcal{L}}_t^{\text{buf}}\right]\!\right] = \mathop{\mathbb{E}}_{\tilde{W}^{t-1}}\left[\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}})\right]\!\right]\right]\!\right]$$
$$= \mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\!\left[\mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}})\right]\!\right] \middle| \tilde{R}^{t-1}\right]\!\right]\right]\!\right].$$

The conditioning performed above is made possible by stream obliviousness. Now suppose that given $\tilde{R}^{t-1}$ the indices $\tilde{\tau}_1, \ldots, \tilde{\tau}_s$ are present in the buffer $\tilde{B}_t$ at time t. Recall that this choice of indices is independent of $\tilde{Z}^{t-1}$ because of stream obliviousness. Then we can write the above as

$$\mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\!\left[\mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}})\right]\!\right] \middle| \tilde{R}^{t-1}\right]\!\right]\right]\!\right]$$
$$= \mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\!\left[\mathop{\mathbb{E}}_{\tilde{Z}^{t-1}}\left[\!\left[\frac{1}{s}\sum_{j=1}^{s}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}}_{\tilde{\tau}_j})\right]\!\right]\right]\!\right]\right]\!\right]$$
$$= \mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\!\left[\mathop{\mathbb{E}}_{\tilde{\mathbf{z}}_1,\ldots,\tilde{\mathbf{z}}_s}\left[\!\left[\frac{1}{s}\sum_{j=1}^{s}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}}_j)\right]\!\right]\right]\!\right]\right]\!\right]$$
$$= \mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\!\left[\mathcal{L}(h_{t-1})\right]\!\right] = \mathcal{L}(h_{t-1}).$$

We thus have

$$\mathop{\mathbb{E}}_{\tilde{W}^{t-1}}\left[\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t}\mathop{\mathbb{E}}_{\mathbf{z}}\left[\!\left[\ell(h_{t-1},\mathbf{z},\tilde{\mathbf{z}})\right]\!\right]\right]\!\right] = \mathcal{L}(h_{t-1}). \tag{8}$$

We now upper bound P_t as

$$\begin{aligned} P_t &= \mathcal{L}(h_{t-1}) - \tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \\ &= \mathop{\mathbb{E}}_{\tilde{W}^{t-1}} \left[\left[\frac{1}{s} \sum_{\tilde{\mathbf{z}} \in \tilde{B}_t} \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h_{t-1}, \mathbf{z}, \tilde{\mathbf{z}}) \right] \right] \right] - \tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \\ &\leq \underbrace{\sup_{h \in \mathcal{H}} \left[\mathop{\mathbb{E}}_{\tilde{W}^{t-1}} \left[\left[\frac{1}{s} \sum_{\tilde{\mathbf{z}} \in \tilde{B}_t} \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \tilde{\mathbf{z}}) \right] \right] \right] - \tilde{\mathcal{L}}_t^{\text{buf}}(h) \right]}_{g_t(\mathbf{w}_1, \ldots, \mathbf{w}_{t-1})}. \end{aligned}$$

Now it turns out that applying McDiarmid's inequality to $g_t(\mathbf{w}_1, \ldots, \mathbf{w}_{t-1})$ directly would yield a very loose bound. This is because of the following reason: since $\hat{\mathcal{L}}_t^{\text{buf}}(h) = \frac{1}{|B_t|} \sum_{\mathbf{z} \in B_t} \ell(h, \mathbf{z}_t, \mathbf{z})$ depends only on s data points, changing any one of the $(t-1)$ variables $\mathbf{w}_i$ brings about a perturbation in g_t of magnitude at most $\mathcal{O}(1/s)$. The problem is that g_t is a function of $(t-1) \gg s$ variables and hence a direct application of McDiarmid's inequality would yield an excess error term of $\sqrt{\frac{t \log(1/\delta)}{s^2}}$ which would in the end require $s = \omega(\sqrt{n})$ to give any non trivial generalization bounds. In contrast, we wish to give results that would give non trivial bounds for $s = \tilde{\omega}(1)$.

In order to get around this problem, we need to reduce the number of variables in the statistic while applying McDiarmid's inequality. Fortunately, we observe that g_t *effectively* depends only on s variables, the data points that end up in the buffer at time t. This allows us to do the following. For any R^{t-1}, define

$$\delta(R^{t-1}) := \mathop{\mathbb{P}}_{Z^{t-1}} \left[g_t(\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}) > \epsilon \middle| R^{t-1} \right].$$

We will first bound $\delta(R^{t-1})$. This will allow us to show

$$\mathop{\mathbb{P}}_{W^{t-1}} \left[g_t(\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}) > \epsilon \right] \leq \mathop{\mathbb{E}}_{R^{t-1}} \left[\delta(R^{t-1}) \right],$$

where we take expectation over the distribution on R^{t-1} induced by the buffer update policy. Note that since we are oblivious to the nature of the distribution over R^{t-1}, our proof works for any stream oblivious buffer update policy. Suppose that given R^{t-1} the indices $\tau_1, \ldots, \tau_s$ are present in the buffer B_t at time t. Then we have

$$\begin{aligned} & g_t(\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}; R^{t-1}) \\ &= \sup_{h \in \mathcal{H}} \left[\mathop{\mathbb{E}}_{\tilde{W}^{t-1}} \left[\left[\frac{1}{s} \sum_{\tilde{\mathbf{z}} \in \tilde{B}_t} \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \tilde{\mathbf{z}}) \right] \right] \right] - \frac{1}{s} \sum_{j=1}^{s} \mathop{\mathbb{E}}_{\mathbf{z}} \left[\ell(h, \mathbf{z}, \mathbf{z}_{\tau_j}) \right] \right] \\ &=: \tilde{g}_t(\mathbf{z}_{\tau_1}, \ldots, \mathbf{z}_{\tau_s}). \end{aligned}$$

The function $\tilde{g}_t$ can be perturbed at most B/s due to a change in one of $\mathbf{z}_{\tau_j}$. Applying McDiarmid's inequality to the function $\tilde{g}_t$ we get with probability at least $1-\delta$,

$$\tilde{g}_t(\mathbf{z}_{\tau_1}, \ldots, \mathbf{z}_{\tau_s}) \leq \mathop{\mathbb{E}}_{Z^{t-1}} \left[\tilde{g}_t(\mathbf{z}_{\tau_1}, \ldots, \mathbf{z}_{\tau_s}) \right] + B \sqrt{\frac{\log \frac{1}{\delta}}{2s}}$$

We analyze $\mathop{\mathbb{E}}_{Z^{t-1}} \left[\tilde{g}_t(\mathbf{z}_{\tau_1}, \ldots, \mathbf{z}_{\tau_s}) \right]$ in Figure 2. In the third step in the calculations we symmetrize the true random variable $\mathbf{z}_{\tau_j}$ with the ghost random variable $\tilde{\mathbf{z}}_{\tilde{\tau}_j}$. This is contrasted with traditional symmetrization where we would symmetrize $\mathbf{z}_i$ with $\tilde{\mathbf{z}}_i$. In our case, we let the buffer construction dictate the matching at the symmetrization step. Thus we get, with probability at least $1-\delta$ over $\mathbf{z}_1, \ldots, \mathbf{z}_{t-1}$,

$$g_t(\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}; R^{t-1}) \leq 2\mathcal{R}_s(\ell \circ \mathcal{H}) + B \sqrt{\frac{\log \frac{1}{\delta}}{2s}}$$

which in turn, upon taking expectations with respect to R^{t-1}, gives us with probability at least $1-\delta$ over $\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}$,

$$P_t = g_t(\mathbf{w}_1, \ldots, \mathbf{w}_{t-1}) \leq 2\mathcal{R}_s(\ell \circ \mathcal{H}) + B \sqrt{\frac{\log \frac{1}{\delta}}{2s}}.$$

Applying a union bound on the bounds for $P_t, t = 2, \ldots, n$ gives us with probability at least $1-\delta$,

$$\begin{aligned} \frac{1}{n-1} \sum_{t=2}^{n} P_t \leq \frac{2}{n-1} \sum_{t=2}^{n} \mathcal{R}_{\min\{t-1,s\}}(\ell \circ \mathcal{H}) \\ + B \sqrt{\frac{\log \frac{n}{\delta}}{2s}}. \end{aligned} \tag{9}$$

Adding Equations (7) and (9) gives us the result. □

D.3. Finite Buffer Algorithms with Strongly Convex Loss Functions

In this section we prove faster convergence bounds for algorithms that offer *finite-buffer* regret bounds and use strongly convex loss functions. Given the development of the method of decoupling training and auxiliary random variables in the last section, we can proceed with the proof right away.

Our task here is to prove bounds on the following quantity

$$\frac{1}{n-1} \sum_{t=2}^{n} \mathcal{L}(h_{t-1}) - \mathcal{L}(h^*).$$

Proceeding as before, we will first prove the following result

$$\mathop{\mathbb{P}}_{Z^n} \left[\frac{1}{n-1} \sum_{t=2}^{n} \mathcal{L}(h_{t-1}) - \mathcal{L}(h^*) > \epsilon \middle| R^n \right] \leq \delta. \tag{10}$$

$$
\begin{aligned}
\mathop{\mathbb{E}}_{Z^{t-1}}[\![\tilde{g}_t(\mathbf{z}_{\tau_1},\ldots,\mathbf{z}_{\tau_s})]\!] &= \mathop{\mathbb{E}}_{Z^{t-1}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\mathop{\mathbb{E}}_{\tilde{W}^{t-1}}\left[\!\!\left[\frac{1}{s}\sum_{\tilde{\mathbf{z}}\in\tilde{B}_t}\mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h,\mathbf{z},\tilde{\mathbf{z}})]\!]\right]\!\!\right] - \frac{1}{s}\sum_{j=1}^{s}\mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h,\mathbf{z},\mathbf{z}_{\tau_j})]\!]\right]\right]\!\!\right] \\
&\le \mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\mathop{\mathbb{E}}_{Z^{t-1},\tilde{Z}^{t-1}}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\frac{1}{s}\sum_{j=1}^{s}\mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h,\mathbf{z},\tilde{\mathbf{z}}_{\tilde{\tau}_j})]\!] - \frac{1}{s}\sum_{j=1}^{s}\mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h,\mathbf{z},\mathbf{z}_{\tau_j})]\!]\right]\right]\!\!\right]\middle|\,\tilde{R}^{t-1}\right] \\
&= \mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\mathop{\mathbb{E}}_{Z^{t-1},\tilde{Z}^{t-1},\epsilon_j}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\frac{1}{s}\sum_{j=1}^{s}\epsilon_j\left(\mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h,\mathbf{z},\tilde{\mathbf{z}}_{\tilde{\tau}_j})]\!] - \mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h,\mathbf{z},\mathbf{z}_{\tau_j})]\!]\right)\right]\right]\!\!\right]\middle|\,\tilde{R}^{t-1}\right] \\
&\le 2\mathop{\mathbb{E}}_{\tilde{R}^{t-1}}\left[\mathop{\mathbb{E}}_{Z^{t-1},\epsilon_j}\left[\!\!\left[\sup_{h\in\mathcal{H}}\left[\frac{1}{s}\sum_{j=1}^{s}\epsilon_j\mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h,\mathbf{z},\mathbf{z}_{\tau_j})]\!]\right]\!\!\right]\middle|\,\tilde{R}^{t-1}\right] \\
&\le 2\mathop{\mathbb{E}}_{\tilde{R}^{t-1}}[\![\mathcal{R}_s(\ell\circ\mathcal{H})]\!] \le 2\mathcal{R}_s(\ell\circ\mathcal{H}).
\end{aligned}
$$

Figure 2. Decoupling training and auxiliary variables for Rademacher complexity-based analysis.

This will allow us, upon taking expectations over R^n, show the following

$$
\mathop{\mathbb{P}}_{W^n}\left[\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*) > \epsilon\right] \le \delta,
$$

which shall complete the proof.

In order to prove the statement given in Equation (10), we will use Theorem 4. As we did in the case of all-pairs loss functions, consider the loss function $\wp(h,\mathbf{z}') := \mathop{\mathbb{E}}_{\mathbf{z}}[\![\ell(h,\mathbf{z},\mathbf{z}')]\!]$ with $\mathcal{P}$ and $\hat{\mathcal{P}}$ as the associated population and empirical risk functionals. Clearly, if ℓ is L-Lipschitz and σ-strongly convex then so is $\wp$. By linearity of expectation, for any $h\in\mathcal{H}$, $\mathcal{P}(h) = \mathcal{L}(h)$. Suppose that given R^{t-1} the indices $\tau_1,\ldots,\tau_s$ are present in the buffer B_t at time t. Applying Theorem 4 on h_{t-1} at the t^{th} step with the loss function $\wp$ gives us that given R^{t-1}, with probability at least $1-\delta$ over the choice of Z^{t-1},

$$
\begin{aligned}
\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*) \le (1+\epsilon)\left(\tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \tilde{\mathcal{L}}_t^{\text{buf}}(h^*)\right) \\
+ \frac{C_\delta}{\epsilon\sigma(\min\{s,t-1\})},
\end{aligned}
$$

where we have again made the simplifying (yet optional) assumption that prior to time $t = s+1$, the buffer contains an exact copy of the stream. Summing across time steps and taking a union bound, gives us that given R^n, with probability at least $1-\delta$ over the choice of Z^n,

$$
\begin{aligned}
\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \le \mathcal{L}(h^*) + \frac{C_{(\delta/n)}}{\epsilon\sigma}\left(\frac{\log 2s}{n-1} + \frac{1}{s}\right) \\
+ \frac{1+\epsilon}{n-1}\sum_{t=2}^{n}\tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \tilde{\mathcal{L}}_t^{\text{buf}}(h^*).
\end{aligned}
$$

Let us define as before

$$
\xi_t := \left(\tilde{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \tilde{\mathcal{L}}_t^{\text{buf}}(h^*)\right) - \left(\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) - \hat{\mathcal{L}}_t^{\text{buf}}(h^*)\right).
$$

Then using the regret bound $\mathfrak{R}_n^{\text{buf}}$ we can write,

$$
\begin{aligned}
\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \le \mathcal{L}(h^*) + \frac{1+\epsilon}{n-1}\left(\mathfrak{R}_n^{\text{buf}} + \sum_{t=2}^{n}\xi_t\right) \\
+ \frac{C_{(\delta/n)}}{\epsilon\sigma}\left(\frac{\log 2s}{n-1} + \frac{1}{s}\right).
\end{aligned}
$$

Assuming $s < n/\log n$ simplifies the above expression to the following:

$$
\begin{aligned}
\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \le \mathcal{L}(h^*) + \frac{1+\epsilon}{n-1}\left(\mathfrak{R}_n^{\text{buf}} + \sum_{t=2}^{n}\xi_t\right) \\
+ \frac{2C_{(\delta/n)}}{\epsilon\sigma s}.
\end{aligned}
$$

Note that this assumption is neither crucial to our proof nor very harsh as for $s = \Omega(n)$, we can always apply the results from the *infinite-buffer* setting using Theorem 5. Moving forward, by using the Bernstein-style inequality from (Kakade & Tewari, 2008), one can show with that probability at least $1-\delta$, we have

$$
\sum_{t=1}^{n}\xi_t \le \max\left\{8L\sqrt{\frac{\mathfrak{D}_n}{\sigma}}, 6B\sqrt{\log\frac{4\log n}{\delta}}\right\}\sqrt{\log\frac{4\log n}{\delta}},
$$

where $\mathfrak{D}_n = \sum_{t=2}^{n}(\mathcal{L}(h_{t-1}) - \mathcal{L}(h^*))$. This gives us

$$
\begin{aligned}
\frac{\mathfrak{D}_n}{n-1} \le \frac{1+\epsilon}{n-1}\left(\mathfrak{R}_n^{\text{buf}} + \max\left\{8L\sqrt{\frac{\mathfrak{D}_n}{\sigma}}, 6B\Delta\right\}\Delta\right) \\
+ \frac{2C_{(\delta/n)}}{\epsilon\sigma s}.
\end{aligned}
$$

Using (Kakade & Tewari, 2008, Lemma 4) and absorbing constants inside the $\mathcal{O}(\cdot)$ notation we get:

$$\mathfrak{D}_n \leq (1+\epsilon)\,\mathfrak{R}_n^{\text{buf}} + \mathcal{O}\left(\frac{C_d^2 n\log(n/\delta)}{\epsilon s} + \log\frac{\log n}{\delta}\right) + \mathcal{O}\left(\sqrt{\left(\mathfrak{R}_n^{\text{buf}} + \frac{C_d^2 n\log(n/\delta)}{\epsilon s}\right)\log\frac{\log n}{\delta}}\right).$$

Let $\mathfrak{W}_n = \max\left\{\mathfrak{R}_n^{\text{buf}}, \frac{2C_d^2 n\log(n/\delta)}{s}\right\}$. Concentrating only on the portion of the expression involving ϵ and ignoring the constants, we get

$$\begin{aligned}&\epsilon\mathfrak{R}_n^{\text{buf}} + \frac{C_d^2 n\log(n/\delta)}{\epsilon s} + \sqrt{\frac{C_d^2 n\log(n/\delta)}{\epsilon s}\log\frac{\log n}{\delta}}\\ &\leq \epsilon\mathfrak{R}_n^{\text{buf}} + \frac{2C_d^2 n\log(n/\delta)}{\epsilon s} \leq \epsilon\mathfrak{W}_n + \frac{2C_d^2 n\log(n/\delta)}{\epsilon s}\\ &\leq 2C_d\sqrt{\frac{2\mathfrak{W}_n n\log(n/\delta)}{s}},\end{aligned}$$

where the second step follows since $\epsilon \leq 1$ and $s \leq n$ and the fourth step follows by using $\epsilon = \sqrt{\frac{2C_d^2 n\log(n/\delta)}{\mathfrak{W}_n s}} \leq 1$ Putting this into the excess risk expression gives us

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{\mathfrak{R}_n^{\text{buf}}}{n-1} + C_d\cdot\mathcal{O}\left(\sqrt{\frac{\mathfrak{W}_n\log(n/\delta)}{sn}}\right),$$

which finishes the proof. Note that in case $\mathfrak{W}_n = \mathfrak{R}_n^{\text{buf}}$, we get

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{\mathfrak{R}_n^{\text{buf}}}{n-1} + C_d\cdot\mathcal{O}\left(\sqrt{\frac{\mathfrak{R}_n^{\text{buf}}\log(n/\delta)}{sn}}\right).$$

On the other hand if $\mathfrak{R}_n^{\text{buf}} \leq \frac{2C_d^2 n\log(n/\delta)}{s}$, we get

$$\frac{1}{n-1}\sum_{t=2}^{n}\mathcal{L}(h_{t-1}) \leq \mathcal{L}(h^*) + \frac{\mathfrak{R}_n^{\text{buf}}}{n-1} + C_d^2\cdot\mathcal{O}\left(\frac{\log(n/\delta)}{s}\right).$$

E. Proof of Theorem 7

Recall that we are considering a composition classes of the form $\ell\circ\mathcal{H} := \{(\mathbf{z},\mathbf{z}') \mapsto \ell(h,\mathbf{z},\mathbf{z}'), h\in\mathcal{H}\}$ where ℓ is some Lipschitz loss function. We have $\ell(h,z_1,z_2) = \phi\left(h(x_1,x_2)Y(y_1,y_2)\right)$ where $Y(y_1,y_2) = y_1 - y_2$ or $Y(y_1,y_2) = y_1y_2$ and $\phi:\mathbb{R}\to\mathbb{R}$ involves some margin loss function. We also assume that ϕ is point wise L-Lipschitz. Let $Y = \sup_{y_1,y_2\in\mathcal{Y}}|Y(y_1,y_2)|$.

Theorem 20 (Theorem 7 restated)**.**

$$\mathcal{R}_n(\ell\circ\mathcal{H}) \leq LY\mathcal{R}_n(\mathcal{H})$$

Proof. Let $\tilde{\phi}(x) = \phi(x) - \phi(0)$. Note that $\tilde{\phi}(\cdot)$ is point wise L-Lipschitz as well as satisfies $\tilde{\phi}(0) = 0$. Let $Y = \sup_{y,y'\in\mathcal{Y}}|Y(y,y')|$.

We will require the following contraction lemma that we state below.

Theorem 21 (Implicit in proof of (Ledoux & Talagrand, 2002), Theorem 4.12)**.** *Let $\mathcal{H}$ be a set of bounded real valued functions from some domain $\mathcal{X}$ and let $\mathbf{x}_1,\ldots,\mathbf{x}_n$ be arbitrary elements from $\mathcal{X}$. Furthermore, let $\phi_i:\mathbb{R}\to\mathbb{R}$, $i=1,\ldots,n$ be L-Lipschitz functions such that $\phi_i(0)=0$ for all i. Then we have*

$$\mathbb{E}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\phi_i(h(\mathbf{x}_i))\right] \leq L\mathbb{E}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\epsilon_i h(\mathbf{x}_i)\right].$$

Using the above inequality we can state the following chain of (in)equalities:

$$\begin{aligned}\mathcal{R}_n(\ell\circ\mathcal{H}) &= \mathbb{E}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\ell(h,\mathbf{z},\mathbf{z}_i)\right]\\ &= \mathbb{E}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\phi\left(h(\mathbf{x},\mathbf{x}_i)Y(y,y_i)\right)\right]\\ &= \mathbb{E}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\tilde{\phi}\left(h(\mathbf{x},\mathbf{x}_i)Y(y,y_i)\right)\right]\\ &\quad + \phi(0)\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\right]\\ &= \mathbb{E}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\epsilon_i\tilde{\phi}\left(h(\mathbf{x},\mathbf{x}_i)Y(y,y_i)\right)\right]\\ &\leq LY\mathbb{E}\left[\sup_{h\in\mathcal{H}}\frac{1}{n}\sum_{i=1}^{n}\epsilon_i h(\mathbf{x},\mathbf{x}_i)\right]\\ &= LY\mathcal{R}_n(\mathcal{H}),\end{aligned}$$

where the fourth step follows from linearity of expectation. The fifth step is obtained by applying the contraction inequality to the functions $\psi_i : x\mapsto\tilde{\phi}(a_i x)$ where $a_i = Y(y,y_i)$. We exploit the fact that the contraction inequality is actually proven for the empirical Rademacher averages due to which we can take $a_i = Y(y,y_i)$ to be a constant dependent only on i,

use the inequality, and subsequently take expectations. We also have, for any i and any $x, y \in \mathbb{R}$,

$$\begin{aligned} |\psi_i(x) - \psi_i(y)| &= \left|\tilde{\phi}(a_i x) - \tilde{\phi}(a_i y)\right| \\ &\leq L\,|a_i x - a_i y| \\ &\leq L\,|a_i|\,|x - y| \\ &\leq LY\,|x - y|\,, \end{aligned}$$

which shows that every function $\psi_i(\cdot)$ is LY-Lipschitz and satisfies $\psi_i(0) = 0$. This makes an application of the contraction inequality possible on the empirical Rademacher averages which upon taking expectations give us the result. □

F. Applications

In this section we shall derive Rademacher complexity bounds for hypothesis classes used in various learning problems. Crucial to our derivations shall be the following result by (Kakade et al., 2008). Recall the usual definition of Rademacher complexity of a *univariate* function class $\mathcal{F} = \{f : \mathcal{X} \to \mathbb{R}\}$

$$\mathcal{R}_n(\mathcal{F}) = \mathbb{E}\left[\!\!\left[\sup_{f\in\mathcal{F}} \frac{1}{n}\sum_{i=1}^{n} \epsilon_i f(\mathbf{x}_i)\right]\!\!\right].$$

Theorem 22 ((Kakade et al., 2008), Theorem 1)**.** *Let $\mathcal{W}$ be a closed and convex subset of some Banach space equipped with a norm $\|\cdot\|$ and dual norm $\|\cdot\|_*$. Let $F : \mathcal{W} \to \mathbb{R}$ be σ-strongly convex with respect to $\|\cdot\|_*$. Assume $\mathcal{W} \subseteq \left\{\mathbf{w} : F(\mathbf{w}) \leq W_*^2\right\}$. Furthermore, let $\mathcal{X} = \{\mathbf{x} : \|\mathbf{x}\| \leq X\}$ and $\mathcal{F}_{\mathcal{W}} := \{\mathbf{w} \mapsto \langle \mathbf{w}, \mathbf{x}\rangle : \mathbf{w} \in \mathcal{W}, \mathbf{x} \in \mathcal{X}\}$. Then, we have*

$$\mathcal{R}_n(\mathcal{F}_{\mathcal{W}}) \leq XW_*\sqrt{\frac{2}{\sigma n}}.$$

We note that Theorem 22 is applicable only to first order learning problems since it gives bounds for univariate function classes. However, our hypothesis classes consist of bivariate functions which makes a direct application difficult. Recall our extension of Rademacher averages to bivariate function classes:

$$\mathcal{R}_n(\mathcal{H}) = \mathbb{E}\left[\!\!\left[\sup_{h\in\mathcal{H}} \frac{1}{n}\sum_{i=1}^{n} \epsilon_i h(\mathbf{z}, \mathbf{z}_i)\right]\!\!\right]$$

where the expectation is over ϵ_i, $\mathbf{z}$ and $\mathbf{z}_i$. To overcome the above problem we will use the following two step proof technique:

1. **Order reduction**: We shall cast our learning problems in a modified input domain where predictors behave linearly as univariate functions. More specifically, given a hypothesis class $\mathcal{H}$ and domain $\mathcal{X}$, we shall construct a modified domain $\tilde{\mathcal{X}}$ and a map $\psi : \mathcal{X} \times \mathcal{X} \to \tilde{\mathcal{X}}$ such that for any $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ and $h \in \mathcal{H}$, we have $h(\mathbf{x}, \mathbf{x}') = \langle h, \psi(\mathbf{x}, \mathbf{x}')\rangle$.
2. **Conditioning**: For every $\mathbf{x} \in \mathcal{X}$, we will create a function class $\mathcal{F}_{\mathbf{x}} = \{\mathbf{x}' \mapsto \langle h, \psi(\mathbf{x}, \mathbf{x}')\rangle : h \in \mathcal{H}\}$. Since $\mathcal{F}_{\mathbf{x}}$ is a univariate function class, we will use Theorem 22 to bound $\mathcal{R}_n(\mathcal{F}_{\mathbf{x}})$. Since $\mathcal{R}_n(\mathcal{H}) = \mathop{\mathbb{E}}_{\mathbf{x}}[\![\mathcal{R}_n(\mathcal{F}_{\mathbf{x}})]\!]$, we shall obtain Rademacher complexity bounds for $\mathcal{H}$.

Hypothesis class	Rademacher Complexity
$\mathcal{B}_q(\|\mathcal{W}\|_q)$	$2\,\|\mathcal{X}\|_p\,\|\mathcal{W}\|_q\,\sqrt{\frac{p-1}{n}}$
$\mathcal{B}_1(\|\mathcal{W}\|_1)$	$2\,\|\mathcal{X}\|_\infty\,\|\mathcal{W}\|_1\,\sqrt{\frac{e\log d}{n}}$

Table 1. Rademacher complexity bounds for AUC maximization. We have $1/p + 1/q = 1$ and $q > 1$.

We give below some examples of learning situations where these results may be applied.

As before, for any subset X of a Banach space and any norm $\|\cdot\|_p$, we define $\|X\|_p := \sup_{\mathbf{x}\in X} \|\mathbf{x}\|_p$. We also define norm bounded balls in the Banach space as $\mathcal{B}_p(r) := \left\{\mathbf{x} : \|\mathbf{x}\|_p \leq r\right\}$ for any $r > 0$. Let the domain $\mathcal{X}$ be a subset of $\mathbb{R}^d$.

For sake of convenience we present the examples using loss functions for classification tasks but the same can be extended to other learning problems such as regression, multi-class classification and ordinal regression.

F.1. AUC maximization for Linear Prediction

In this case the goal is to maximize the area under the ROC curve for a linear classification problem at hand. This translates itself to a learning situation where $\mathcal{W}, \mathcal{X} \subseteq \mathbb{R}^d$. We have $h_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') = \mathbf{w}^\top\mathbf{x} - \mathbf{w}^\top\mathbf{x}'$ and $\ell(h_{\mathbf{w}}, z_1, z_2) = \phi\left((y - y')h_{\mathbf{w}}(\mathbf{x}, \mathbf{x}')\right)$ where ϕ is the hinge loss or the exponential loss (Zhao et al., 2011).

In order to apply Theorem 22, we rewrite the hypothesis as $h_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') = \mathbf{w}^\top(\mathbf{x} - \mathbf{x}')$ and consider the input domain $\tilde{\mathcal{X}} = \{\mathbf{x} - \mathbf{x}' : \mathbf{x}, \mathbf{x}' \in \mathcal{X}\}$ and the map $\psi : (\mathbf{x}, \mathbf{x}') \mapsto \mathbf{x} - \mathbf{x}'$. Clearly if $\mathcal{X} \subseteq \{\mathbf{x} : \|\mathbf{x}\| \leq X\}$ then $\tilde{\mathcal{X}} \subseteq \{\mathbf{x} : \|\mathbf{x}\| \leq 2X\}$ and thus we have $\left\|\tilde{\mathcal{X}}\right\| \leq 2\,\|\mathcal{X}\|$ for any norm $\|\cdot\|$. It is now possible to regularize the hypothesis class $\mathcal{W}$ using a variety of norms.

If we wish to define our hypothesis class as $\mathcal{B}_q(\cdot), q > 1$, then in order to apply Theorem 22, we can use the regularizer $F(\mathbf{w}) = \|\mathbf{w}\|_q^2$. If we wish the sparse

Hypothesis Class	Rademacher Complexity
$\mathcal{B}_{2,2}(\|\mathcal{W}\|_{2,2})$	$\|\mathcal{X}\|_2^2 \|\mathcal{W}\|_{2,2} \sqrt{\frac{1}{n}}$
$\mathcal{B}_{2,1}(\|\mathcal{W}\|_{2,1})$	$\|\mathcal{X}\|_2 \|\mathcal{X}\|_\infty \|\mathcal{W}\|_{2,1} \sqrt{\frac{e \log d}{n}}$
$\mathcal{B}_{1,1}(\|\mathcal{W}\|_{1,1})$	$\|\mathcal{X}\|_\infty^2 \|\mathcal{W}\|_{1,1} \sqrt{\frac{2e \log d}{n}}$
$\mathcal{B}_{S(1)}(\|\mathcal{W}\|_{S(1)})$	$\|\mathcal{X}\|_2^2 \|\mathcal{W}\|_{S(1)} \sqrt{\frac{e \log d}{n}}$

Table 2. Rademacher complexity bounds for Similarity and Metric learning

hypotheses class, $\mathcal{B}_1(W_1)$, we can use the regularizer $F(\mathbf{w}) = \|\mathbf{w}\|_q^2$ with $q = \frac{\log d}{\log d - 1}$ as this regularizer is strongly convex with respect to the L_1 norm (Kakade et al., 2012). Table 1 gives a succinct summary of such possible regularizations and corresponding Rademacher complexity bounds.

Kernelized AUC maximization: Since the L_2 regularized hypothesis class has a dimension independent Rademacher complexity, it is possible to give guarantees for algorithms performing AUC maximization using kernel classifiers as well. In this case we have a Mercer kernel K with associated reproducing kernel Hilbert space $\mathcal{H}_K$ and feature map $\Phi_K : \mathcal{X} \to \mathcal{H}_K$. Our predictors lie in the RKHS, i.e., $\mathbf{w} \in \mathcal{H}_K$ and we have $h_{\mathbf{w}}(\mathbf{x}, \mathbf{x}') = \mathbf{w}^\top (\Phi_K(\mathbf{x}) - \Phi_K(\mathbf{x}'))$. In this case we will have to use the map $\psi : (\mathbf{x}, \mathbf{x}') \mapsto \Phi_K(\mathbf{x}) - \Phi_K(\mathbf{x}') \in \mathcal{H}_K$. If the kernel is bounded, i.e., for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$, we have $|K(\mathbf{x}, \mathbf{x}')| \leq \kappa^2$, then we can get a Rademacher average bound of $2\kappa \|\mathcal{W}\|_2 \sqrt{\frac{1}{n}}$.

F.2. Linear Similarity and Mahalanobis Metric learning

A variety of applications, such as in vision, require one to fine tune one's notion of proximity by learning a similarity or metric function over the input space. We consider some such examples below. In the following, we have $\mathbf{W} \in \mathbb{R}^{d \times d}$.

1. *Mahalanobis metric learning*: in this case we wish to learn a metric $M_{\mathbf{W}}(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^\top \mathbf{W} (\mathbf{x} - \mathbf{x}')$ using the loss function $\ell(M_{\mathbf{W}}, \mathbf{z}, \mathbf{z}') = \phi\left(yy'\left(1 - M_{\mathbf{W}}^2(\mathbf{x}, \mathbf{x}')\right)\right)$ (Jin et al., 2009).
2. *Linear kernel learning*: in this case we wish to learn a linear kernel function $K_{\mathbf{W}}(\mathbf{x}, \mathbf{x}') = \mathbf{x}^\top \mathbf{W} \mathbf{x}', \mathbf{W} \succeq 0$. A variety of loss functions have been proposed to aid the learning process
   (a) *Kernel-target Alignment*: the loss function used is $\ell(K_{\mathbf{W}}, \mathbf{z}, \mathbf{z}') = \phi(yy' K_{\mathbf{W}}(\mathbf{x}, \mathbf{x}'))$ where ϕ is used to encode some notion of alignment (Cristianini et al., 2001; Cortes et al., 2010b).
   (b) *S-Goodness*: this is used in case one wishes to learn a *good* similarity function that need not be positive semi definite (Bellet et al., 2012; Balcan & Blum, 2006) by defining $\ell(K_{\mathbf{W}}, \mathbf{z}) = \phi\left(y \mathop{\mathbb{E}}_{(\mathbf{x}', y')} [\![y' K_{\mathbf{W}}(\mathbf{x}, \mathbf{x}')]\!]\right)$.

In order to apply Theorem 22, we will again rewrite the hypothesis and consider a different input domain. For the similarity learning problem, write the similarity function as $K_{\mathbf{W}}(\mathbf{x}, \mathbf{x}') = \left\langle \mathbf{W}, \mathbf{x}\mathbf{x}'^\top \right\rangle$ and consider the input space $\tilde{\mathcal{X}} = \left\{\mathbf{x}\mathbf{x}'^\top : \mathbf{x}, \mathbf{x}' \in \mathcal{X}\right\} \subseteq \mathbb{R}^{d \times d}$ along with the map $\psi : (\mathbf{x}, \mathbf{x}') \mapsto \mathbf{x}\mathbf{x}'^\top$. For the metric learning problem, rewrite the metric as $M_{\mathbf{W}}(\mathbf{x}, \mathbf{x}') = \left\langle \mathbf{W}, (\mathbf{x} - \mathbf{x}')(\mathbf{x} - \mathbf{x}')^\top \right\rangle$ and consider the input space $\tilde{\mathcal{X}} = \left\{(\mathbf{x} - \mathbf{x}')(\mathbf{x} - \mathbf{x}')^\top : \mathbf{x}, \mathbf{x}' \in \mathcal{X}\right\} \subseteq \mathbb{R}^{d \times d}$ along with the map $\psi : (\mathbf{x}, \mathbf{x}') \mapsto (\mathbf{x} - \mathbf{x}')(\mathbf{x} - \mathbf{x}')^\top$.

In this case it is possible to apply a variety of matrix norms to regularize the hypothesis class. We consider the following (mixed) matrix norms : $\|\cdot\|_{1,1}$, $\|\cdot\|_{2,1}$ and $\|\cdot\|_{2,2}$. We also consider the Schatten norm $\|\mathbf{X}\|_{S(p)} := \|\boldsymbol{\sigma}(\mathbf{X})\|_p$ that includes the widely used *trace norm* $\|\boldsymbol{\sigma}(\mathbf{X})\|_1$. As before, we define norm bounded balls in the Banach space as follows: $\mathcal{B}_{p,q}(r) := \left\{\mathbf{x} : \|\mathbf{x}\|_{p,q} \leq r\right\}$.

Using results on construction of strongly convex functions with respect to theses norms from (Kakade et al., 2012), it is possible to get bounds on the Rademacher averages of the various hypothesis classes. However these bounds involve norm bounds for the modified domain $\tilde{\mathcal{X}}$. We make these bounds explicit by expressing norm bounds for $\tilde{\mathcal{X}}$ in terms of those for $\mathcal{X}$. From the definition of $\tilde{\mathcal{X}}$ for the similarity learning problems, we get, for any $p, q \geq 1$, $\left\|\tilde{\mathcal{X}}\right\|_{p,q} \leq \|\mathcal{X}\|_p \|\mathcal{X}\|_q$. Also, since every element of $\tilde{\mathcal{X}}$ is of the form $\mathbf{x}\mathbf{x}'^\top$, it has only one non zero singular value $\|\mathbf{x}\|_2 \|\mathbf{x}'\|_2$ which gives us $\left\|\tilde{X}\right\|_{S(p)} \leq \|\mathcal{X}\|_2^2$ for any $p \geq 1$.

For the metric learning problem, we can similarly get $\left\|\tilde{X}\right\|_{p,q} \leq 4 \|\mathcal{X}\|_p \|\mathcal{X}\|_q$ and $\left\|\tilde{X}\right\|_{S(p)} \leq 4 \|\mathcal{X}\|_2^2$ for any $p \geq 1$ which allows us to get similar bounds as those for similarity learning but for an extra constant factor. We summarize our bounds in Table 2. We note that (Cao et al., 2012) devote a substantial amount of effort to calculate these values for the mixed norms on a case-by-case basis (and do not consider Schatten norms either) whereas, using results exploiting strong convexity and strong smoothness from (Kakade et al., 2012), we are able to get the same as simple corollaries.

Hypothesis Class	Rademacher Avg. Bound
$\mathcal{S}_2(1)$	$\kappa^2\sqrt{\frac{p}{n}}$
$\Delta(1)$	$\kappa^2\sqrt{\frac{e\log p}{n}}$

Table 3. Rademacher complexity bounds for Multiple kernel learning

F.3. Two-stage Multiple kernel learning

The analysis of the previous example can be replicated for learning non-linear Mercer kernels as well. Additionally, since all Mercer kernels yield Hilbertian metrics, these methods can be extended to learning Hilbertian metrics as well. However, since Hilbertian metric learning has not been very popular in literature, we restrict our analysis to kernel learning alone. We present this example using the framework proposed by (Kumar et al., 2012) due to its simplicity and generality.

We are given p Mercer kernels $K_1, \ldots, K_p$ that are bounded, i.e., for all i, $|K_i(\mathbf{x}, \mathbf{x}')| \leq \kappa^2$ for all $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ and our task is to find a combination of these kernels given by a vector $\boldsymbol{\mu} \in \mathbb{R}^p, \boldsymbol{\mu} \geq 0$ such that the kernel $K_{\boldsymbol{\mu}}(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{p} \boldsymbol{\mu}_i K_i(\mathbf{x}, \mathbf{x}')$ is a *good* kernel (Balcan & Blum, 2006). In this case the loss function used is $\ell(\boldsymbol{\mu}, \mathbf{z}, \mathbf{z}') = \phi\left(yy' K_{\boldsymbol{\mu}}(\mathbf{x}, \mathbf{x}')\right)$ where $\phi(\cdot)$ is meant to encode some notion of alignment. Kumar et al. (2012) take $\phi(\cdot)$ to be the hinge loss.

To apply Theorem 22, we simply use the "K-space" construction proposed in (Kumar et al., 2012). We write $K_{\boldsymbol{\mu}}(\mathbf{x}, \mathbf{x}') = \langle \boldsymbol{\mu}, z(\mathbf{x}, \mathbf{x}') \rangle$ where $z(\mathbf{x}, \mathbf{x}') = (K_1(\mathbf{x}, \mathbf{x}'), \ldots, K_p(\mathbf{x}, \mathbf{x}'))$. Consequently our modified input space looks like $\tilde{\mathcal{X}} = \{z(\mathbf{x}, \mathbf{x}') : \mathbf{x}, \mathbf{x}' \in \mathcal{X}\} \subseteq \mathbb{R}^p$ with the map $\psi : (\mathbf{x}, \mathbf{x}') \mapsto z(\mathbf{x}, \mathbf{x}')$. Popular regularizations on the kernel combination vector $\boldsymbol{\mu}$ include the sparsity inducing L_1 regularization that constrains $\boldsymbol{\mu}$ to lie on the unit simplex $\Delta(1) = \{\boldsymbol{\mu} : \|\boldsymbol{\mu}\|_1 = 1, \boldsymbol{\mu} \geq 0\}$ and L_2 regularization that restricts $\boldsymbol{\mu}$ to lie on the unit sphere $\mathcal{S}_2(1) = \{\boldsymbol{\mu} : \|\boldsymbol{\mu}\|_2 = 1, \boldsymbol{\mu} \geq 0\}$. Arguments similar to the one used to discuss the case of AUC maximization for linear predictors give us bounds on the Rademacher averages for these two hypothesis classes in terms of $\left\|\tilde{X}\right\|_2$ and $\left\|\tilde{X}\right\|_\infty$. Since $\left\|\tilde{X}\right\|_2 \leq \kappa^2\sqrt{p}$ and $\left\|\tilde{X}\right\|_\infty \leq \kappa^2$, we obtain explicit bounds on the Rademacher averages that are given in Table 3.

We note that for the L_1 regularized case, our bound has a similar dependence on the number of kernels, i.e., $\sqrt{\log p}$ as the bounds presented in (Cortes et al., 2010a). For the L_2 case however, we have a worse dependence of $\sqrt{p}$ than Cortes et al. (2010a) who get a $\sqrt[4]{p}$ dependence. However, it is a bit unfair to compare the two bounds since Cortes et al. (2010a) consider single stage kernel learning algorithms that try to learn the kernel combination as well as the classifier in a single step whereas we are dealing with a two-stage process where classifier learning is disjoint from the kernel learning step.

G. Regret Bounds for Reservoir Sampling Algorithms

The Reservoir Sampling algorithm (Vitter, 1985) essentially performs sampling without replacement which means that the samples present in the buffer are not i.i.d. samples from the preceding stream. Due to this, proving regret bounds by way of uniform convergence arguments becomes a bit more difficult. However, there has been a lot of work on analyzing learning algorithms that learn from non-i.i.d. data such as data generated by ergodic processes. Of particular interest is a result by Serfling [2] that gives Hoeffding style bounds for data generated from a finite population without replacement.

Although Serfling's result does provide a way to analyze the **RS** algorithm, doing so directly would require using arguments that involve covering numbers that offer bounds that are dimension dependent and that are not tight. It would be interesting to see if equivalents of the McDiarmid's inequality and Rademacher averages can be formulated for samples obtained without replacement to get tighter results. For our purposes, we remedy the situation by proposing a new sampling algorithm that gives us i.i.d. samples in the buffer allowing existing techniques to be used to obtain regret bounds (see Appendices H and I).

H. Analysis of the RS-x Algorithm

In this section we analyze the **RS-x** substream sampling algorithm and prove its statistical properties. Recall that the **RS-x** algorithm simply admits a point into the buffer if there is space. It performs a *Repopulation step* at the first instance of overflow which involves refilling the buffer by sampling with replacement from all the set of points seen so far (including the one that caused the overflow). In subsequent steps, a *Normal update step* is performed. The following theorem formalizes the properties of the sampling algorithm

Theorem 23. *Suppose we have a stream of elements* $\mathbf{z}_1, \ldots, \mathbf{z}_n$ *being sampled into a buffer* B *of size* s *using*

[2] R. J. Serfling, Probability Inequalities for the Sum in Sampling without Replacement, *The Annals of Statistics*, 2(1):39-48, 1974.

*the **RS-x** algorithm. Then at any time $t \geq s+2$, each element of B is an i.i.d. sample from the set Z^{t-1}.*

Proof. To prove the results, let us assume that the buffer contents are addressed using the variables $\zeta_1, \ldots, \zeta_s$. We shall first concentrate on a fixed element, say ζ_1 (which we shall call simply ζ for notational convenience) of the buffer and inductively analyze the probability law $\mathcal{P}_t$ obeyed by ζ at each time step $t \geq s+2$.

We will prove that the probability law obeyed by ζ at time t is $\mathcal{P}_t(\zeta) = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \mathbb{1}_{\{\zeta=\mathbf{z}_\tau\}}$. The law is interpreted as saying the following: for any $\tau \leq t-1$, $\mathbb{P}[\zeta = \mathbf{z}_\tau] = \frac{1}{t-1}$ and shows that the element ζ is indeed a uniform sample from the set Z^{t-1}. We would similarly be able to show this for all locations $\zeta_2, \ldots, \zeta_s$ which would prove that the elements in the buffer are indeed identical samples from the preceding stream. Since at each step, the **RS-x** algorithm updates all buffer locations independently, the random variables $\zeta_1, \ldots, \zeta_s$ are independent as well which would allow us to conclude that at each step we have s i.i.d. samples in the buffer as claimed.

We now prove the probability law for ζ. We note that the repopulation step done at time $t = s+1$ explicitly ensures that at step $t = s+2$, the buffer contains s i.i.d samples from Z^{s+1} i.e. $\mathcal{P}_{s+2}(\zeta) = \frac{1}{s+1}\sum_{\tau=1}^{s+1} \mathbb{1}_{\{\zeta=\mathbf{z}_\tau\}}$. This forms the initialization of our inductive argument. Now suppose that at the t^{th} time step, the claim is true and ζ obeys the law $\mathcal{P}_t(\zeta) = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \mathbb{1}_{\{\zeta=\mathbf{z}_\tau\}}$. At the t^{th} step, we would update the buffer by making the incoming element $\mathbf{z}_t$ replace the element present at the location indexed by ζ with probability $1/(t+1)$. Hence ζ would obey the following law after the update

$$\left(1 - \frac{1}{t}\right)\mathcal{P}_t(\zeta) + \frac{1}{t}\mathbb{1}_{\{\zeta=\mathbf{z}_t\}} = \frac{1}{t}\sum_{\tau=1}^{t} \mathbb{1}_{\{\zeta=\mathbf{z}_\tau\}}$$

which shows that at the $(t+1)^{\text{th}}$ step, ζ would follow the law $\mathcal{P}_{t+1}(\zeta) = \frac{1}{t}\sum_{\tau=1}^{t} \mathbb{1}_{\{\zeta=\mathbf{z}_\tau\}}$ which completes the inductive argument and the proof. □

I. Proof of Theorem 8

We now prove Theorem 8 that gives a high confidence regret bound for the **OLP** learning algorithm when used along with the **RS-x** buffer update policy. Our proof proceeds in two steps: in the first step we prove a uniform convergence type guarantee that would allow us to convert regret bounds with respect to the *finite-buffer* penalties $\hat{\mathcal{L}}_t^{\text{buf}}$ into regret bounds in in terms of the *all-pairs* loss functions $\hat{\mathcal{L}}_t$. In the second step we then prove a regret bound for **OLP** with respect to the *finite-buffer* penalties.

We proceed with the first step of the proof by proving the lemma given below. Recall that for any sequence of training examples $\mathbf{z}_1, \ldots, \mathbf{z}_n$, we define, for any $h \in \mathcal{H}$, the all-pairs loss function as $\hat{\mathcal{L}}_t(h) = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \ell(h, \mathbf{z}_t, \mathbf{z}_\tau)$. Moreover, if the online learning process uses a buffer, the we also define the *finite-buffer* loss function as $\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) = \frac{1}{|B_t|}\sum_{\mathbf{z}\in B_t} \ell(h_{t-1}, \mathbf{z}_t, \mathbf{z})$.

Lemma 24. *Suppose we have an online learning algorithm that incurs buffer penalties based on a buffer B of size s that is updated using the **RS-x** algorithm. Suppose further that the learning algorithm generates an ensemble $h_1, \ldots, h_{n-1}$. Then for any $t \in [1, n-1]$, with probability at least $1-\delta$ over the choice of the random variables used to update the buffer B until time t, we have*

$$\hat{\mathcal{L}}_t(h_{t-1}) \leq \hat{\mathcal{L}}_t^{buf}(h_{t-1}) + C_d \cdot \mathcal{O}\left(\sqrt{\frac{\log\frac{1}{\delta}}{s}}\right)$$

Proof. Suppose $t \leq s+1$, then since at that point the buffer stores the stream exactly, we have

$$\hat{\mathcal{L}}_t(h_{t-1}) = \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1})$$

which proves the result. Note that, as Algorithm 2 indicates, at step $t = s+1$ the buffer is updated (using the repopulation step) only after the losses have been calculated and hence step $t = s+1$ still works with a buffer that stores the stream exactly.

We now analyze the case $t > s+1$. At each step $\tau > s$, the **RS-x** algorithm uses s independent Bernoulli random variables (which we call *auxiliary random variables*) to update the buffer, call them $r_1^\tau, \ldots, r_s^\tau$ where r_j^τ is used to update the j^{th} item ζ_j in the buffer. Let $\mathfrak{r}_j^t := \{r_j^{s+1}, r_j^2, \ldots, r_j^t\} \in \{0,1\}^t$ denote an ensemble random variable composed of $t-s$ independent Bernoulli variables. It is easy to see that the element ζ_j is completely determined at the t^{th} step given $\mathfrak{r}_j^{t-1}$.

Theorem 23 shows, for any $t > s+1$, that the buffer contains s i.i.d. samples from the set Z^{t-1}. Thus, for any *fixed* function $h \in \mathcal{H}$, we have for any $j \in [s]$,

$$\mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1}}[\![\ell(h, \mathbf{z}_t, \zeta_j)]\!] = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \ell(h, \mathbf{z}_t, \mathbf{z}_\tau)$$

which in turn shows us that

$$\mathop{\mathbb{E}}_{\mathfrak{r}_1^{t-1},\ldots,\mathfrak{r}_s^{t-1}}\left[\!\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h)\right]\!\!\right] = \frac{1}{t-1}\sum_{\tau=1}^{t-1} \ell(h, \mathbf{z}_t, \mathbf{z}_\tau) = \hat{\mathcal{L}}_t(h)$$

Now consider a ghost sample of auxiliary random variables $\tilde{\mathfrak{r}}_1^{t-1}, \ldots, \tilde{\mathfrak{r}}_s^{t-1}$. Since our hypothesis h_{t-1} is independent of these ghost variables, we can write

$$\mathop{\mathbb{E}}_{\tilde{\mathfrak{r}}_1^{t-1},\ldots,\tilde{\mathfrak{r}}_s^{t-1}} \left[\!\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \right]\!\!\right] = \hat{\mathcal{L}}_t(h_{t-1})$$

We recall that error in the proof presented in Zhao et al. (2011) was to apply such a result on the *true* auxiliary variables upon which h_{t-1} is indeed dependent. Thus we have

$$\begin{aligned}
&\hat{\mathcal{L}}_t(h_{t-1}) - \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \\
&= \mathop{\mathbb{E}}_{\tilde{\mathfrak{r}}_1^{t-1},\ldots,\tilde{\mathfrak{r}}_s^{t-1}} \left[\!\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \right]\!\!\right] - \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \\
&\leq \underbrace{\sup_{h\in\mathcal{H}} \left[\mathop{\mathbb{E}}_{\tilde{\mathfrak{r}}_1^{t-1},\ldots,\tilde{\mathfrak{r}}_s^{t-1}} \left[\!\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h) \right]\!\!\right] - \hat{\mathcal{L}}_t^{\text{buf}}(h) \right]}_{g_t(\mathfrak{r}_1^{t-1},\ldots,\mathfrak{r}_s^{t-1})}
\end{aligned}$$

Now, the perturbation to any of the ensemble variables $\mathfrak{r}_j$ (a perturbation to an ensemble variable implies a perturbation to one or more variables forming that ensemble) can only perturb only the element ζ_j in the buffer. Since $\hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) = \frac{1}{s}\sum_{\mathbf{z}\in B_t} \ell(h_{t-1}, \mathbf{z}_t, \mathbf{z})$ and the loss function is B-bounded, this implies that a perturbation to any of the ensemble variables can only perturb $g(\mathfrak{r}_1^{t-1}, \ldots, \mathfrak{r}_s^{t-1})$ by at most B/s. Hence an application of McDiarmid's inequality gives us, with probability at least $1-\delta$,

$$g_t(\mathfrak{r}_1^{t-1}, \ldots, \mathfrak{r}_s^{t-1}) \leq \mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1}} \left[\!\!\left[g_t(\mathfrak{r}_1^{t-1}, \ldots, \mathfrak{r}_s^{t-1}) \right]\!\!\right] + B\sqrt{\frac{\log\frac{1}{\delta}}{2s}}$$

Analyzing the expectation term we get

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1}} \left[\!\!\left[g_t(\mathfrak{r}_1^{t-1}, \ldots, \mathfrak{r}_s^{t-1}) \right]\!\!\right] \\
&= \mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1}} \left[\!\!\left[\sup_{h\in\mathcal{H}} \left[\mathop{\mathbb{E}}_{\tilde{\mathfrak{r}}_1^{t-1},\ldots,\tilde{\mathfrak{r}}_s^{t-1}} \left[\!\!\left[\hat{\mathcal{L}}_t^{\text{buf}}(h) \right]\!\!\right] - \hat{\mathcal{L}}_t^{\text{buf}}(h) \right] \right]\!\!\right] \\
&\leq \mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1},\tilde{\mathfrak{r}}_j^{t-1}} \left[\!\!\left[\sup_{h\in\mathcal{H}} \left[\frac{1}{s}\sum_{j=1}^{s} \ell(h, \mathbf{z}_t, \tilde{\zeta}_j) - \ell(h, \mathbf{z}_t, \zeta_j) \right] \right]\!\!\right] \\
&= \mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1},\tilde{\mathfrak{r}}_j^{t-1},\epsilon_j} \left[\!\!\left[\sup_{h\in\mathcal{H}} \left[\frac{1}{s}\sum_{j=1}^{s} \epsilon_j \left(\ell(h, \mathbf{z}_t, \tilde{\zeta}_j) - \ell(h, \mathbf{z}_t, \zeta_j) \right) \right] \right]\!\!\right] \\
&\leq 2 \mathop{\mathbb{E}}_{\mathfrak{r}_j^{t-1},\tilde{\mathfrak{r}}_j^{t-1},\epsilon_j} \left[\!\!\left[\sup_{h\in\mathcal{H}} \left[\frac{1}{s}\sum_{j=1}^{s} \epsilon_j \ell(h, \mathbf{z}_t, \zeta_j) \right] \right]\!\!\right] \\
&\leq 2\mathcal{R}_s(\ell\circ\mathcal{H})
\end{aligned}$$

where in the third step we have used the fact that symmetrizing a pair of true and ghost ensemble variables is equivalent to symmetrizing the buffer elements they determine. In the last step we have exploited the definition of Rademacher averages with the (empirical) measure $\frac{1}{t-1}\sum_{\tau=1}^{t-1}\delta_{\mathbf{z}_\tau}$ imposed over the domain $\mathcal{Z}$.

For hypothesis classes for which we have $\hat{\mathcal{R}}_s(\ell\circ\mathcal{H}) = C_d\cdot\mathcal{O}\left(\sqrt{\frac{1}{s}}\right)$, this proves the claim. □

Using a similar proof progression we can also show the following:

Lemma 25. *For any fixed $h\in\mathcal{H}$ and any $t\in[1,n-1]$, with probability at least $1-\delta$ over the choice of the random variables used to update the buffer B until time t, we have*

$$\hat{\mathcal{L}}_t^{buf}(h) \leq \hat{\mathcal{L}}_t(h) + C_d\cdot\mathcal{O}\left(\sqrt{\frac{\log\frac{1}{\delta}}{s}}\right)$$

Combining Lemmata 24 and 25 and taking a union bound over all time steps, the following corollary gives us a *buffer to all-pairs* conversion bound.

Lemma 26. *Suppose we have an online learning algorithm that incurs buffer penalties based on a buffer B of size s that is updated using the **RS-x** algorithm. Suppose further that the learning algorithm generates an ensemble $h_1, \ldots, h_{n-1}$. Then with probability at least $1-\delta$ over the choice of the random variables used to update the buffer B, we have*

$$\mathfrak{R}_n \leq \mathfrak{R}_n^{buf} + C_d\,(n-1)\cdot\mathcal{O}\left(\sqrt{\frac{\log\frac{n}{\delta}}{s}}\right),$$

where we recall the definition of the all-pairs *regret as*

$$\mathfrak{R}_n := \sum_{t=2}^{n}\hat{\mathcal{L}}_t(h_{t-1}) - \inf_{h\in\mathcal{H}}\sum_{t=2}^{n}\hat{\mathcal{L}}_t(h)$$

and the finite-buffer *regret as*

$$\mathfrak{R}_n^{buf} := \sum_{t=2}^{n}\hat{\mathcal{L}}_t^{buf}(h_{t-1}) - \inf_{h\in\mathcal{H}}\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{buf}(h).$$

Proof. Let $\hat{h} := \arg\inf_{h\in\mathcal{H}} \sum_{t=2}^{n}\hat{\mathcal{L}}_t(h)$. Then Lemma 25 gives us, upon summing over t and taking a union bound,

$$\sum_{t=2}^{n}\hat{\mathcal{L}}_t^{\text{buf}}(\hat{h}) \leq \sum_{t=2}^{n}\hat{\mathcal{L}}_t(\hat{h}) + C_d(n-1)\cdot\mathcal{O}\left(\sqrt{\frac{\log\frac{n}{\delta}}{s}}\right), \tag{11}$$

whereas Lemma 24 similarly guarantees

$$\sum_{t=2}^{n} \hat{\mathcal{L}}_t(h_{t-1}) \leq \sum_{t=2}^{n} \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) + C_d(n-1) \cdot \mathcal{O}\left(\sqrt{\frac{\log \frac{n}{\delta}}{s}}\right) \tag{12}$$

where both results hold with high confidence. Adding the Equations (11) and (12) and using $\sum_{t=2}^{n} \hat{\mathcal{L}}_t^{\text{buf}}(h_{t-1}) \leq \inf_{h \in \mathcal{H}} \sum_{t=2}^{n} \hat{\mathcal{L}}_t^{\text{buf}}(\hat{h}) + \mathfrak{R}_n^{\text{buf}}$ completes the proof. □

As the final step of the proof, we give below a *finite-buffer* regret bound for the **OLP** algorithm.

Lemma 27. *Suppose the **OLP** algorithm working with an s-sized buffer generates an ensemble $\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}$. Further suppose that the loss function ℓ being used is L-Lipschitz and the space of hypotheses $\mathcal{W}$ is a compact subset of a Banach space with a finite diameter D with respect to the Banach space norm. Then we have*

$$\mathfrak{R}_n^{buf} \leq LD\sqrt{n-1}$$

Proof. We observe that the algorithm **OLP** is simply a variant of the GIGA algorithm (Zinkevich, 2003) being applied with the loss functions $\ell_t^{\text{GIGA}} : \mathbf{w} \mapsto \hat{\mathcal{L}}_t^{\text{buf}}(\mathbf{w})$. Since ℓ_t^{GIGA} inherits the Lipschitz constant of $\hat{\mathcal{L}}_t^{\text{buf}}$ which in turn inherits it from ℓ, we can use the analysis given by Zinkevich (2003) to conclude the proof. □

Combining Lemmata 26 and 27 gives us the following result:

Theorem 28 (Theorem 8 restated)**.** *Suppose the **OLP** algorithm working with an s-sized buffer generates an ensemble $\mathbf{w}_1, \ldots, \mathbf{w}_{n-1}$. Then with probability at least $1-\delta$,*

$$\frac{\mathfrak{R}_n}{n-1} \leq \mathcal{O}\left(C_d\sqrt{\frac{\log \frac{n}{\delta}}{s}} + \sqrt{\frac{1}{n-1}}\right)$$

J. Implementing the RS-x Algorithm

Although the **RS-x** algorithm presented in the paper allows us to give clean regret bounds, it suffers from a few drawbacks. From a theoretical point of view, the algorithm is inferior to Vitter's **RS** algorithm in terms of randomness usage. The **RS** algorithm (see (Zhao et al., 2011) for example) uses a Bernoulli random variable and a discrete uniform random variable at each time step. The discrete random variable takes values in $[s]$ as a result of which the algorithm uses a total of $\mathcal{O}(\log s)$ random bits at each step.

Algorithm 3 RS-x^2 : An Alternate Implementation of the **RS-x** Algorithm

Input: Buffer B, new point $\mathbf{z}_t$, buffer size s, timestep t
Output: Updated buffer B_{new}

if $|B| < s$ **then** //There is space
  $B_{\text{new}} \leftarrow B \cup \{\mathbf{z}_t\}$
else //Overflow situation
  if $t = s+1$ **then** //Repopulation step
    TMP $= B \cup \{\mathbf{z}_t\}$
    $B_{\text{new}} = \phi$
    for $i = 1$ **to** s **do**
      Select random $\mathbf{r} \in$ TMP with replacement
      $B_{\text{new}} \leftarrow B_{\text{new}} \cup \{\mathbf{r}\}$
    end for
  else //Normal update step
    $B_{\text{new}} \leftarrow B$
    Sample $k \sim \text{Binomial}(s, 1/t)$
    Remove k random elements from B_{new}
    $B_{\text{new}} \leftarrow B_{\text{new}} \cup \left(\coprod_{i=1}^{k} \{\mathbf{z}_t\}\right)$
  end if
end if
return B_{new}

The **RS-x** algorithm as proposed, on the other hand, uses s Bernoulli random variables at each step (to decide which buffer elements to replace with the incoming point) taking its randomness usage to $\mathcal{O}(s)$ bits. From a practical point of view this has a few negative consequences:

1. Due to increased randomness usage, the variance of the resulting algorithm increases.

2. At step t, the Bernoulli random variables required all have success probability $1/t$. This quantity drops down to negligible values for even moderate values of t. Note that Vitter's **RS** on the other hand requires a Bernoulli random variable with success probability s/t which dies down much more slowly.

3. Due to the requirement of such high precision random variables, the imprecisions of any pseudo random generator used to simulate this algorithm become apparent resulting in poor performance.

In order to ameliorate the situation, we propose an alternate implementation of the *normal* update step of the **RS-x** algorithm in Algorithm 3. We call this new sampling policy **RS-x^2** . We shall formally demonstrate the equivalence of the **RS-x** and the **RS-x^2** policies by showing that both policies result in a buffer whose each element is a uniform sample from the preceding stream with replacement. This shall be done by proving that the joint distribution of the buffer elements remains the same whether the **RS-x** *normal* update is applied or the **RS-x^2** *normal* step is ap-

plied (note that **RS-x** and **RS-x²** have identical *repopulation steps*). This will ensure that any learning algorithm will be unable to distinguish between the two update mechanisms and consequently, our regret guarantees shall continue to hold.

First we analyze the randomness usage of the **RS-x²** update step. The update step first samples a number $K_t \sim B(s, 1/t)$ from the binomial distribution and then replaces K_t random locations with the incoming point. Choosing k locations without replacement from a pool of s locations requires at most $k \log s$ bits of randomness. Since K_t is sampled from the binomial distribution $B(s, 1/t)$, we have $K_t = \mathcal{O}(1)$ in expectation (as well as with high probability) since $t > s$ whenever this step is applied. Hence our randomness usage per update is at most $\mathcal{O}(\log s)$ random bits which is much better than the randomness usage of **RS-x** and that actually matches that of Vitter's **RS** upto a constant.

To analyze the statistical properties of the **RS-x²** update step, let us analyze the state of the buffer after the update step. In the **RS-x** algorithm, the state of the buffer after an update is completely specified once we enumerate the locations that were replaced by the incoming point. Let the indicator variable R_i indicate whether the i^{th} location was replaced or not. Let $r \in \{0,1\}^s$ denote a *fixed* pattern of replacements. Then the original implementation of the update step of **RS-x** guarantees that

$$\mathop{\mathbb{P}}_{\textbf{RS-x}}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\right] = \left(\frac{1}{t}\right)^{\|r\|_1}\left(1-\frac{1}{t}\right)^{s-\|r\|_1}$$

To analyze the same for the alternate implementation of the **RS-x²** update step, we first notice that choosing k items from a pool of s without replacement is identical to choosing the first k locations from a random permutation of the s items. Let us denote $\|r\|_1 = k$. Then we have,

$$\begin{aligned}\mathop{\mathbb{P}}_{\textbf{RS-x}^2}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\right] &= \sum_{j=1}^{s}\mathbb{P}\left[\bigwedge_{i=1}^{s}(R_i = r_i) \wedge K_t = j\right] \\ &= \mathbb{P}\left[\bigwedge_{i=1}^{s}(R_i = r_i) \wedge K_t = k\right] \\ &= \mathbb{P}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\,\middle|\, K_t = k\right]\mathbb{P}\left[K_t = k\right]\end{aligned}$$

We have

$$\mathbb{P}\left[K_t = k\right] = \binom{s}{k}\left(\frac{1}{t}\right)^k\left(1-\frac{1}{t}\right)^{s-k}$$

The number of arrangements of s items such that some specific k items fall in the first k positions is $k!(s-k)!$.

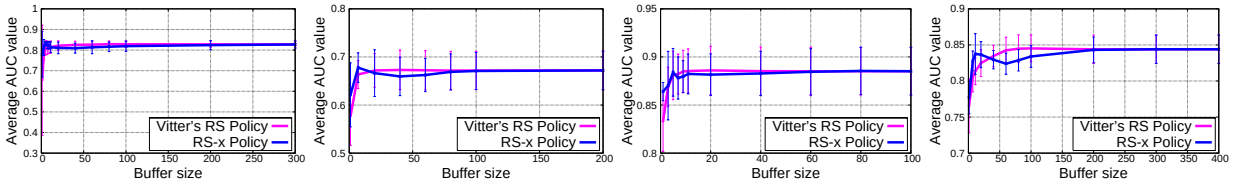
Thus we have

$$\begin{aligned}\mathop{\mathbb{P}}_{\textbf{RS-x}^2}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\right] &= \binom{s}{k}\left(\frac{1}{t}\right)^k\left(1-\frac{1}{t}\right)^{s-k}\frac{k!(s-k)!}{s!} \\ &= \left(\frac{1}{t}\right)^k\left(1-\frac{1}{t}\right)^{s-k} \\ &= \mathop{\mathbb{P}}_{\textbf{RS-x}}\left[\bigwedge_{i=1}^{s}(R_i = r_i)\right]\end{aligned}$$

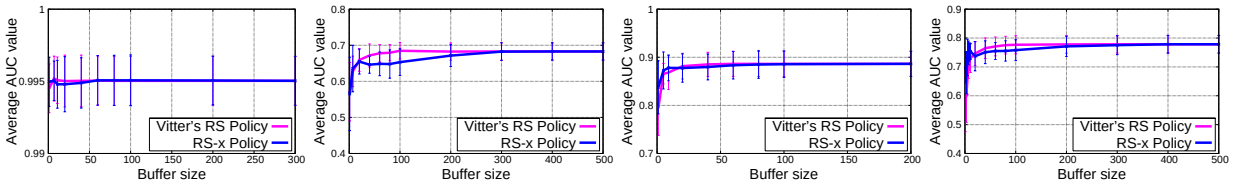
which completes the argument.

K. Additional Experimental Results

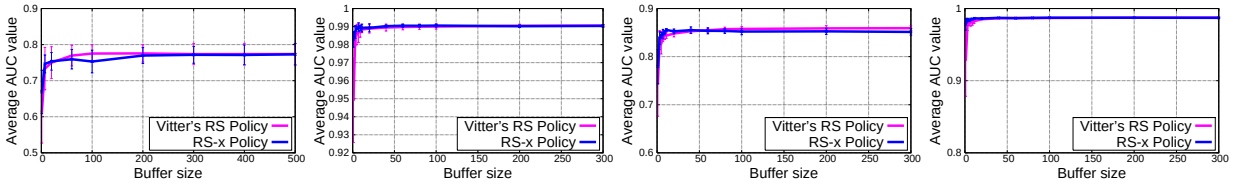
Here we present experimental results on 14 different benchmark datasets (refer to Figure 3) comparing the **OLP** algorithm using the **RS-x²** buffer policy with the OAM_{gra} algorithm using the **RS** buffer policy. We continue to observe the trend that **OLP** performs competitively to OAM_{gra} while enjoying a slight advantage in small buffer situations in most cases.



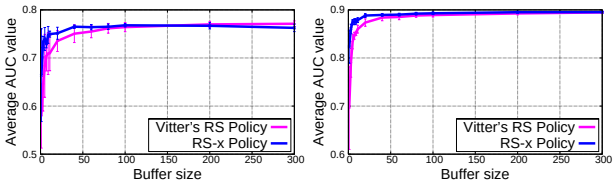
(a) Fourclass (b) Liver Disorders (c) Heart (d) Diabetes



(e) Breast Cancer (f) Vowel (g) Ionosphere (h) German



(i) SVMguide3 (j) SVMguide1 (k) Statlog (l) Cod RNA



(m) Letter (n) Adult

Figure 3. Comparison between OAM_{gra} (using **RS** policy) and **OLP** (using **RS-x** policy) on AUC maximization tasks.