Optimal Margin Distribution Learning in Dynamic Environments

Teng Zhang,[1] **Peng Zhao,**[2] **Hai Jin**[1*]
[1]National Engineering Research Center for Big Data Technology and System
Services Computing Technology and System Lab, Cluster and Grid Computing Lab
School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan, 430074, China
[2]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, 210023, China
{tengzhang, hjin}@hust.edu.cn, zhaop@lamda.nju.edu.cn

Abstract

Recently a promising research direction of statistical learning has been advocated, i.e., the optimal margin distribution learning with the central idea that instead of the minimal margin, the margin distribution is more crucial to the generalization performance. Although the superiority of this new learning paradigm has been verified under batch learning settings, it remains open for online learning settings, in particular, the dynamic environments in which the underlying decision function varies over time. In this paper, we propose the dynamic optimal margin distribution machine and theoretically analyze its regret. Although the obtained bound has the same order with the best known one, our method can significantly relax the restrictive assumption that the function variation should be given ahead of time, resulting in better applicability in practical scenarios. We also derive an excess risk bound for the special case when the underlying decision function only evolves several discrete changes rather than varying continuously. Extensive experiments on both synthetic and real data sets demonstrate the superiority of our method.

Introduction

Support vector machines (SVMs) and Boosting have always been two mainstream learning approaches during the past decades. The former (Cortes and Vapnik 1995) roots in the statistical learning theory (Vapnik 1995) which aims to search a large margin separator, while the latter also enjoys a long history of being explained by margin theory (Freund and Schapire 1995; Schapire et al. 1998), due to it is resistant to over-fitting empirically (Reyzin and Schapire 2006; Wang et al. 2011; Zhou 2012).

Recently the margin theory for Boosting has finally been defended (Gao and Zhou 2013), and disclosed that instead of optimizing a single margin, margin distribution is more crucial to the generalization performance. Similar conclusions have also been obtained for SVM-style learning approaches. Specifically, Zhang and Zhou (2014; 2017; 2019) proposed the optimal margin distribution machines (ODMs) for binary and multi-class classification problems respectively, and achieved superior generalization performance to the traditional large margin based methods. After that, the optimal margin distribution learning has turned into a promising research direction and attracted a lot of attentions, just to name a few, Zhou and Zhou (2016) exploit it to handle unequal misclassification cost; Cheng, Zhang, and Wen (2016) use it for imbalanced data; Ou et al. (2017) applies it to feature elimination; Zhang and Zhou (2018a; 2018b) extends it to clustering and semi-supervised learning settings.

Although this new learning paradigm has achieved many successes under batch learning settings, it remains open for online learning settings (Rosenblatt 1958; Freund and Schapire 1999; Li and Long 1999; Kivinen, Smola, and Williamson 2001; Zhao, Hoi, and Jin 2011), in particular, the changing environments in which the underlying decision function varies over time (Hazan and Seshadhri 2009; Besbes, Gur, and Zeevi 2015). In this paper, we proposes the dynamic optimal margin distribution machine, whose basic idea is to simultaneously maintain a set of candidate learners, dynamically adjust their weights according to the performances, and finally perform a majority vote by combining their outputs. We also theoretically analyze its dynamic regret and derives an $\widetilde{O}(\sqrt{TV_T})$ [1] bound, where V_T is the *function variation* which can capture the change intensity of the underlying decision function. Although this bound has the same order with the best known one (Besbes, Gur, and Zeevi 2015), our method can significantly relax the restrictive assumption that the complexity measure V_T should be given ahead of time, leading to better applicability in practical scenarios. For the special case when the underlying decision function only evolves several discrete changes rather than varying continuously, i.e., they stay the same most of time, we also derive an excess risk bound for the proposed method. Extensive experiments on both synthetic and real data sets demonstrate the superiority of our method.

The rest of the paper is organized as follows. We first briefly introduce some preliminaries, and then detail the proposed method. After that we discuss some theoretical analyses, followed by empirical studies. Finally we conclude the paper with future work.

[*]This research was supported by the National Key Research & Development Program of China (2018YFB1004002).

[1] We use the notation $\widetilde{O}(\cdot)$ to suppress poly-logarithmic dependence on T.

Preliminaries

For convenience, we first introduce some notation conventions. Throughout the paper, we denote scalars with lower case letters (e.g., y), and vectors with bold face letters (e.g., $\boldsymbol{x}$). Sets are designated by upper case letters with mathcal font (e.g., $\mathcal{W}$). Let $\mathcal{X} \subseteq \mathbb{R}^n$ and $\mathcal{Y} = \{1, -1\}$ denote the input and output spaces respectively. For any $m \geq 1$, the set of integers $\{1, \ldots, m\}$ is denoted by $[m]$. Given an argument π, we use the $\mathbf{1}_{[\pi]}$ to denote the indicator function which outputs 1 if π holds and 0 otherwise. The hinge loss function is denoted by $[a]_+ = \max\{0, a\}$.

Margin Distribution

It is well-known that SVMs employ the large margin principle to select the decision boundary. As a result, the final classification hyperplane only consists of a small amount of instances and the rest are totally ignored (Schölkopf and Smola 2001), which may be misleading in some situations. See Figure 1 for an illustration (Zhou 2014).

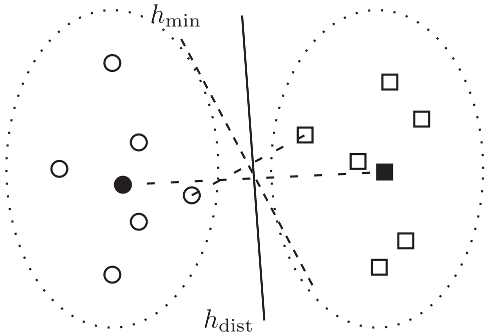

Figure 1: Dotted ellipses are two underlying distributions from which circles and squares are instances sampled. Solid circle and square are mean instances (not necessarily real instances in the training data). $h_{\min}$ and h_{dist} are classification hyperplanes obtained by optimizing the minimum margin and margin distribution respectively.

Instead of maximizing the minimal margin, one more robust strategy is to optimize the margin distribution, which can exploit the whole data and prevent from being cheated by noisy instances. To characterize the margin distribution, the most straightforward way is to employ the first- and second-order statistics. Moreover, as suggested in (Gao and Zhou 2013), maximizing the margin mean and minimizing the margin variance simultaneously can yield a tighter generalization bound, so we achieve the following formulation:

$$\begin{aligned} \min_{\boldsymbol{w}, \bar{\gamma}, \xi_i, \epsilon_i} \ & \frac{1}{2}\|\boldsymbol{w}\|^2 - \eta\bar{\gamma} + \frac{\lambda}{m}\sum_{i \in [m]} (\xi_i^2 + \epsilon_i^2) \\ \text{s.t.} \ & \gamma(\boldsymbol{x}_i, y_i) \geq \bar{\gamma} - \xi_i, \\ & \gamma(\boldsymbol{x}_i, y_i) \leq \bar{\gamma} + \epsilon_i, \ \forall i \in [m], \end{aligned} \tag{1}$$

where $\gamma(\boldsymbol{x}_i, y_i) = y_i \boldsymbol{w}^\top \boldsymbol{x}_i$ is the margin of labeled instance $(\boldsymbol{x}_i, y_i)$, $\bar{\gamma}$ is the margin mean, η and λ are the trading-off parameters. Note that ξ_i and ϵ_i are deviations from the margin mean, so the last term $\sum_{i \in [m]}(\xi_i^2 + \epsilon_i^2)/m$ is exactly the margin variance.

Since scaling $\boldsymbol{w}$ does not affect the final classification results, Eqn. (1) can be simplified by fixing the margin mean as 1. Besides, introducing different weights for the two different kinds of deviations, i.e., ξ_i and ϵ_i, can make the model more flexible so as to characterize the margin distributions more adaptively. Finally, to achieve a sparse solution, we tolerate the deviation smaller than the given threshold θ as the insensitive loss used in support vector regression. Putting the above all together, we come up with the final formulation of the optimal margin distribution machine (ODM):

$$\begin{aligned} \min_{\boldsymbol{w}, \xi_i, \epsilon_i} \ & F(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{\lambda}{m}\sum_{i \in [m]} \frac{\xi_i^2 + \mu\epsilon_i^2}{(1-\theta)^2} \\ \text{s.t.} \ & y_i \boldsymbol{w}^\top \boldsymbol{x}_i \geq 1 - \theta - \xi_i, \\ & y_i \boldsymbol{w}^\top \boldsymbol{x}_i \leq 1 + \theta + \epsilon_i, \ \forall i \in [m], \end{aligned}$$

where μ is the parameter for trading-off the two deviations, and $(1-\theta)^2$ in the denominator is to scale the second term as a surrogate loss for 0-1 loss.

Dynamic Environments

The ODM was originally proposed for batch learning setting, under which we assume that there is an unknown (underlying) distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. Given the labeled training set S drawn identically and independently (i.i.d.) according to $\mathcal{D}$ and the hypothesis set $\mathcal{H} = \{h : \mathcal{X} \mapsto \mathcal{Y}\}$, the goal is to learn a classifier $h \in \mathcal{H}$ such that the generalization error $\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\mathbf{1}_{h(\boldsymbol{x}) \neq y}]$ is small.

However, in some practical applications, the assumption of batch learning is too limited to be met. Considering the spam filtering system, emails arrive into the system and need be classified as spam/valid repeatedly. In addition, no i.i.d. sampling can be assumed since the data can be even adversarially generated. As opposed to batch learning, the common tool for handling this kind of sequential data is online learning (Shalev-Shwartz 2012), which can be formulated as a repeated game between a learner and an adversary (Cesa-Bianchi and Lugosi 2006). At the t-th round, the learner selects a decision function parameterized by $\boldsymbol{w}_t$ from some convex set $\mathcal{W} \subseteq \mathbb{R}^n$ and the adversary chooses a convex function $f_t : \mathcal{W} \mapsto \mathbb{R}$. Then the function is revealed to the learner who suffers a loss $f_t(\boldsymbol{w}_t)$. Since no distributional assumption is made, there is no notion of generalization. Instead the performance of an algorithm is measured by competing with the optimal fixed decision function in hindsight, i.e., the *regret* (Zinkevich 2003), defined as the difference between their cumulative loss:

$$\text{Regret}_T = \sum_{t \in [T]} f_t(\boldsymbol{w}_t) - \min_{\boldsymbol{w} \in \mathcal{W}} \sum_{t \in [T]} f_t(\boldsymbol{w}). \tag{2}$$

During the past decades, various online algorithms have been proposed to yield sublinear regret under different scenarios (Shalev-Shwartz, Singer, and Srebro 2007; Hazan, Agarwal, and Kale 2007). The latest summary of this line of research can be found in (Hazan 2016).

Although equipped with rich theories, the notion of regret defined in Eqn. (2) is not always the right objective to minimize, especially in dynamic environments where the underlying decision function can vary over time (Zhao, Cai, and Zhou 2019; Zhao et al. 2019b) and there is no single fixed decision function doing well overall. To overcome this limitation, it is natural to consider a more stringent measure, i.e., *dynamic regret* (Hall and Willett 2013; Besbes, Gur, and Zeevi 2015; Mokhtari et al. 2016; Yang et al. 2016; Zhang et al. 2017; Zhao et al. 2019a), defined as the difference between the cumulative loss of the learner and that of a sequence of local minimizers:

$$\begin{aligned} \text{D-Regret}_T &= \sum_{t\in[T]} f_t(\boldsymbol{w}_t) - \sum_{t\in[T]} \min_{\boldsymbol{w}\in\mathcal{W}} f_t(\boldsymbol{w}) \\ &= \sum_{t\in[T]} f_t(\boldsymbol{w}_t) - \sum_{t\in[T]} f_t(\boldsymbol{w}_t^\star). \end{aligned}$$

Note that the dynamic regret competes with the optimal $\boldsymbol{w}_t^\star$ at each round, which allows the comparator changing over time. This key characteristic makes it more favored in dynamic environments. Moreover, the original regret defined in Eqn. (2) is also called static regret since only one static comparator is used.

It is proved that a sublinear dynamic regret can not be achievable unless some complexity measures are imposed (Besbes, Gur, and Zeevi 2015), among which a commonly used one is the *function variation* characterized by:

$$V_T = \sum_{t\in[T]} \sup_{\boldsymbol{w}\in\mathcal{W}} |f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})|, \tag{3}$$

which accumulates the maximum variation between two consecutive functions f_{t+1} and f_t for any feasible $\boldsymbol{w} \in \mathcal{W}$. The brief summaries of other complexity measures, e.g., (squared) path-length can be found in (Zhang et al. 2017). Since these complexity measures are largely compatible and yield qualitatively comparable results, we only consider the function variation in this paper.

The Proposed Method

In this section, we detail the proposed method whose basic idea is to simultaneously maintain a set of candidate learners to cope with the dynamic environments. The weights of them are repeatedly adjusted according to the performances, followed by a majority vote by combining their outputs.

Candidate Learner

Follow the Regularized Leader (FTRL) is one of the most general online learning framework which can induce low-regret algorithms. At the t-th round, its update is formed as:

$$\boldsymbol{w}_{t+1} = \operatorname*{argmin}_{\boldsymbol{w}\in\mathcal{W}} \left\{ \sum_{i\in[t]} f_i(\boldsymbol{w}) + R(\boldsymbol{w}) \right\}, \tag{4}$$

where the first summation term aims to select $\boldsymbol{w}$ which has minimal loss on all past rounds as the Follow the Leader (FTL) framework, and $R(\boldsymbol{w})$ is the regularization term to stabilize the FTL solutions and obtain better regret bound.

One can solve the Eqn. (4) with standard convex optimization methods. However, it turns out that it is without loss of generality to assume $f_t(\boldsymbol{w})$ is a linear function because by its convexity the regret can be bounded as

$$\begin{aligned} \text{Regret}_T &= \max_{\boldsymbol{w}\in\mathcal{W}} \sum_{t\in[T]} (f_t(\boldsymbol{w}_t) - f_t(\boldsymbol{w})) \\ &\le \max_{\boldsymbol{w}\in\mathcal{W}} \sum_{t\in[T]} \nabla f_t(\boldsymbol{w}_t)^\top (\boldsymbol{w}_t - \boldsymbol{w}). \end{aligned}$$

Therefore, we can imagine that the loss function is actually a linear function $\nabla f_t(\boldsymbol{w}_t)^\top \boldsymbol{w}$, and a regret bound for this linear problem is clearly also a regret bound for the original problem. With this reduction and by picking the Euclidean regularization $R(\boldsymbol{w}) = \|\boldsymbol{w}\|^2/(2\eta)$, FTRL turns into

$$\begin{aligned} \boldsymbol{w}_{t+1} &= \operatorname*{argmin}_{\boldsymbol{w}\in\mathcal{W}} \left\{ \sum_{i\in[t]} \nabla f_i(\boldsymbol{w}_i)^\top \boldsymbol{w} + \frac{1}{2\eta}\|\boldsymbol{w}\|^2 \right\} \\ &= \operatorname*{argmin}_{\boldsymbol{w}\in\mathcal{W}} \left\| \boldsymbol{w} + \eta \sum_{i\in[t]} \nabla f_i(\boldsymbol{w}_i) \right\|^2, \end{aligned}$$

which means $\boldsymbol{w}_{t+1}$ is the projection of $-\eta \sum_{i\in[t]} \nabla f_i(\boldsymbol{w}_i)$ onto the convex set $\mathcal{W}$.

Note that ODM can be equivalently rewritten into the following unconstrained form:

$$\min_{\boldsymbol{w}} \; F(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \frac{\lambda}{m} \sum_{i\in[m]} \ell(\boldsymbol{w}; \boldsymbol{x}_i),$$

where $\ell(\boldsymbol{w}; \boldsymbol{x}_i)$ is the ODM loss function defined as

$$\ell(\boldsymbol{w}; \boldsymbol{x}_i) = \frac{[1-\theta-y_i\boldsymbol{w}^\top\boldsymbol{x}_i]_+^2 + \mu[y_i\boldsymbol{w}^\top\boldsymbol{x}_i - 1 - \theta]_+^2}{(1-\theta)^2}.$$

By setting $f_i(\boldsymbol{w}) = \|\boldsymbol{w}\|^2/2 + \lambda\ell(\boldsymbol{w}; \boldsymbol{x}_i)$ and noting that $\mathcal{W} = \mathbb{R}^n$, we can obtain

$$\begin{aligned} \boldsymbol{w}_{t+1} &= -\eta \sum_{i\in[t-1]} \nabla f_i(\boldsymbol{w}_i) - \eta\nabla f_t(\boldsymbol{w}_t) \\ &= \boldsymbol{w}_t - \eta(\boldsymbol{w}_t + \lambda\nabla\ell(\boldsymbol{w}_t; \boldsymbol{x}_t)). \end{aligned}$$

Actually this update can also be derived by applying the online gradient descent framework (Zinkevich 2003) to ODM, therefore we call it online ODM, and adopt it as the candidate learner to make use of the power of optimal margin distribution learning.

Dynamic ODM

Evidently each candidate learner should possess something special so as to distinguish from others. Otherwise it is completely unnecessary to maintain them because they will have the same behavior. On the other hand, each update of FTRL depends on all the past instances, which may be not very suitable for dynamic environments. When the underlying decision function changes, these old data can not provide useful information any more, or even only has negative effects when facing an adversary environment.

Based on these observations, we personalize each candidate learner by leveraging the restarting mechanism (Besbes, Gur, and Zeevi 2015). Specifically, we divide the time horizon into several epochs and different candidate learners use a personalized epoch size. In each epoch, the update is based on the instances seen in this epoch only. In other words, the candidate learners will empty their "memory" when an epoch is over, and start another epoch with a random initialization.

Now the key issue turns to the design of epoch sizes. Intuitively speaking, the diversity of the candidate learners is of importance, because they should be able to recover the underlying decision function with high accuracy no matter when and how it changes. To this end, we introduce the following set $\mathcal{P}$ which consists of N epoch sizes forming a geometrical sequence with a common ratio of 2:

$$\mathcal{P} = \{\Delta_i = 2^{i-1}\sqrt{1/(2C)} \mid i \in [N]\},$$

where Δ_i is the epoch size used by the i-th candidate learner, and C is a constant satisfying $|f_t(\boldsymbol{w})| \le C$ for any $\boldsymbol{w}$ and any $t \in [T]$. The number of candidate learners N is set as $\lceil \log(2CT/V_T)/2 \rceil + 1$ where V_T is the function variation defined in Eqn. (3). The reason for setting N in this manner will be discussed in the theoretical analysis part.

For the i-th candidate learner, it divides the time horizon T into $\lceil T/\Delta_i \rceil$ epochs with equal size Δ_i. In each epoch, it carries out the online ODM update for Δ_i times in total. Algorithm 1 summarizes the pseudo-code of the update for the i-th restarted online ODM.

Algorithm 1 Update for the i-th restarted online ODM

```
Require: Time horizon $T$, epoch size $\Delta_i$.
$j \leftarrow 1$;
while $j \le \lceil T/\Delta_i \rceil$ do
    $\tau \leftarrow (j-1)\Delta_i$;
    Randomly initialize $\boldsymbol{w}_\tau$;
    for $t = \tau, \dots, \tau + \Delta_i - 1$ do
        $\boldsymbol{w}^i_{t+1} \leftarrow (1-\eta)\boldsymbol{w}^i_t + \eta\lambda\nabla\ell(\boldsymbol{w}^i_t; \boldsymbol{x}_t)$;
    end for
    $j \leftarrow j + 1$;
end while
```

The weight for the i-th candidate learner at the t-th round is denoted by β^i_t and initialized as $1/N$. In other words, the algorithm begins with uniform weights over the N candidate learners. At the t-th round, the weight is adjusted according to the exponentially weighted average strategy (Cesa-Bianchi and Lugosi 2006):

$$\beta^i_{t+1} = \frac{\beta^i_t \exp(-\epsilon f_t(\boldsymbol{w}^i_t))}{\sum_{i\in[N]} \beta^i_t \exp(-\epsilon f_t(\boldsymbol{w}^i_t))},$$

where $\epsilon > 0$ is the learning rate to be specified later. Obviously, the larger the loss, the smaller the weight. The obtained classifier at the t-th round is a weighted combination of the outputs returned by all the candidates, i.e., $\boldsymbol{w}_t = \sum_{i\in[N]} \beta^i_t \boldsymbol{w}^i_t$. Algorithm 2 summarizes the pseudo-code of the whole algorithm.

Algorithm 2 Dynamic ODM

```
Require: Time horizon $T$, learning rate $\epsilon$, set $\mathcal{P}$.
$\beta^i_1 \leftarrow 1/N$ for any $i \in [N]$;
for $t = 1, \dots, T$ do
    Receive $\boldsymbol{w}^i_t$ from the $i$-th candidate for any $i \in [N]$;
    Output the classifier $\boldsymbol{w}_t \leftarrow \sum_{i\in[N]} \beta^i_t \boldsymbol{w}^i_t$;
    Update the weight
        $\beta^i_{t+1} \leftarrow \frac{\beta^i_t \exp(-\epsilon f_t(\boldsymbol{w}^i_t))}{\sum_{i\in[N]} \beta^i_t \exp(-\epsilon f_t(\boldsymbol{w}^i_t))}$;
end for
```

Theoretical Analysis

In this section we analyze the dynamic regret and excess risk of the proposed method, respectively.

Dynamic Regret

The following lemma suggests that if the function variation V_T is available, by setting the epoch size $\Delta^\star = \sqrt{T/V_T}$, we can obtain an $\widetilde{O}(\sqrt{TV_T})$ dynamic regret bound.

Lemma 1. *Suppose $f_t(\boldsymbol{w})$ is α-strongly convex. Let Δ denote the epoch size. The dynamic regret of the restarted online ODM is upper bounded by*

$$\sum_{t\in[T]} f_t(\boldsymbol{w}_t) - \sum_{t\in[T]} f_t(\boldsymbol{w}^\star_t) \le \frac{G^2 T}{2\alpha\Delta}(\log\Delta + 1) + 2\Delta V_T,$$

where G is the upper bound on the norm of the gradient of $f_t(\boldsymbol{w})$, i.e., $\|\nabla f_t(\boldsymbol{w})\| \le G$ for any $\boldsymbol{w}$.

Proof. Let K denote the total number of epochs, i.e., $K = T/\Delta$, and $\mathcal{E}_k$ denote the k-th epoch for any $k \in [K]$. Then, the dynamic regret can be decomposed as

$$\sum_{k\in[K]} \left(\sum_{t\in\mathcal{E}_k} f_t(\boldsymbol{w}_t) - \sum_{t\in\mathcal{E}_k} f_t(\boldsymbol{w}^\star_t) \right) = \sum_{k\in[K]} (A_k + B_k),$$

where

$$A_k = \sum_{t\in\mathcal{E}_k} f_t(\boldsymbol{w}_t) - \min_{\boldsymbol{w}} \sum_{t\in\mathcal{E}_k} f_t(\boldsymbol{w}),$$
$$B_k = \min_{\boldsymbol{w}} \sum_{t\in\mathcal{E}_k} f_t(\boldsymbol{w}) - \sum_{t\in\mathcal{E}_k} f_t(\boldsymbol{w}^\star_t).$$

Note that A_k is the static regret of epoch $\mathcal{E}_k$, which can be upper bounded by $G^2(\log\Delta + 1)/2\alpha$ (Hazan, Agarwal, and Kale 2007), so it suffices to bound B_k.

Let $\widetilde{\boldsymbol{w}}_k$ denote the optimal solution of the epoch $\mathcal{E}_k$, and t_1 be the index of the first item in the epoch $\mathcal{E}_k$. Then

$$B_k = \sum_{t\in\mathcal{E}_k} f_t(\widetilde{\boldsymbol{w}}_k) - f_t(\boldsymbol{w}^\star_t) \le \sum_{t\in\mathcal{E}_k} f_t(\boldsymbol{w}^\star_{t_1}) - f_t(\boldsymbol{w}^\star_t).$$

The remaining is to show that for any $t \in \mathcal{E}_k$, we have

$$f_t(\boldsymbol{w}^\star_{t_1}) - f_t(\boldsymbol{w}^\star_t) \le 2V_k,$$

where V_k is the function variation in the epoch $\mathcal{E}_k$, i.e., $V_k = \sum_{t\in\mathcal{E}_k} \sup_{\boldsymbol{w}} |f_{t+1}(\boldsymbol{w}) - f_t(\boldsymbol{w})|$. The result holds by noticing

$$f_t(\boldsymbol{w}_{t_1}^\star) \leq f_{t_1}(\boldsymbol{w}_{t_1}^\star) + V_k,\ f_{t_1}(\boldsymbol{w}_t^\star) \leq f_t(\boldsymbol{w}_t^\star) + V_k,$$

together with $f_{t_1}(\boldsymbol{w}_{t_1}^\star) \leq f_{t_1}(\boldsymbol{w}_t^\star)$. Hence, B_k can be upper bounded by $2\Delta \sum_{k\in[K]} V_k = 2\Delta V_T$, and the proof is completed by combining the upper bounds of A_k and B_k. □

Next we prove that if the set $\mathcal{P}$ contains some epoch size Δ_j satisfying $\Delta^\star/2 \leq \Delta_j \leq \Delta^\star$, then the dynamic regret of our proposed method can be upper bounded by $\widetilde{O}(\sqrt{TV_T})$. The proof needs the following lemma (Cesa-Bianchi and Lugosi 2006, Theorem 2.2).

Lemma 2. *For any learning rate $\epsilon > 0$, we have*

$$\sum_{t\in[T]} f_t(\boldsymbol{w}_t) - \min_{i\in[N]} \sum_{t\in[T]} f_t(\boldsymbol{w}_t^i) \leq \frac{\epsilon CT}{8} + \frac{C\log N}{\epsilon}.$$

Now we can present the main theorem of this paper.

Theorem 1. *Suppose Δ_j satisfies $\Delta^\star/2 \leq \Delta_j \leq \Delta^\star$. The dynamic regret of dynamic ODM is upper bounded by*

$$\sum_{t\in[T]} f_t(\boldsymbol{w}_t) - \sum_{t\in[T]} f_t(\boldsymbol{w}_t^\star) = \widetilde{O}(\sqrt{TV_T}).$$

Proof. By setting $\epsilon = \sqrt{(8\log N)/T}$ in Lemma 2, we have

$$\begin{aligned}\sum_{t\in[T]} f_t(\boldsymbol{w}_t) &\leq \min_{i\in[N]} \sum_{t\in[T]} f_t(\boldsymbol{w}_t^i) + C\sqrt{(T/2)\log N} \\ &\leq \sum_{t\in[T]} f_t(\boldsymbol{w}_t^j) + C\sqrt{(T/2)\log(\log(2CT/V_T))}.\end{aligned}$$

According to Lemma 1, we have

$$\sum_{t\in[T]} (f_t(\boldsymbol{w}_t^j) - f_t(\boldsymbol{w}_t^\star)) \leq \frac{G^2 T}{2\alpha\Delta_j}(\log\Delta_j + 1) + 2\Delta_j V_T.$$

Combine the above two inequalities with $\Delta^\star/2 \leq \Delta_j \leq \Delta^\star$, and we can obtain

$$\begin{aligned}\sum_{t\in[T]} f_t(\boldsymbol{w}_t) &\leq \sum_{t\in[T]} f_t(\boldsymbol{w}_t^\star) + \frac{G^2T}{2\alpha\Delta_j}(\log\Delta_j + 1) \\ &\qquad + 2\Delta_j V_T + C\sqrt{(T/2)\log(\log(2CT/V_T))} \\ &\leq \sum_{t\in[T]} f_t(\boldsymbol{w}_t^\star) + \frac{G^2T}{\alpha\Delta^\star}(\log\Delta^\star + 1) + 2\Delta^\star V_T \\ &\qquad + C\sqrt{(T/2)\log(\log(2CT/V_T))} \\ &= \sum_{t\in[T]} f_t(\boldsymbol{w}_t^\star) + \sqrt{TV_T}\left(\frac{G^2}{2\alpha}\log\frac{T}{V_T} + \frac{G^2}{\alpha} + 2\right) \\ &\qquad + C\sqrt{(T/2)\log(\log(2CT/V_T))} \\ &= \sum_{t\in[T]} f_t(\boldsymbol{w}_t^\star) + \widetilde{O}(\sqrt{TV_T}).\end{aligned}$$

□

Finally, we need verify the existence of Δ_j. Note that N is set as $\lceil\log(2CT/V_T)/2\rceil + 1$, thus we have

$$\Delta_N = 2^{N-1}\sqrt{1/(2C)} \geq 2^{\log\sqrt{2CT/V_T}}\sqrt{1/(2C)} = \Delta^\star$$

which means that there exists an index $j \in [N]$ such that $\Delta_j \leq \Delta^\star \leq \Delta_{j+1} = 2\Delta_j$, and thus $\Delta^\star/2 \leq \Delta_j \leq \Delta^\star$. This is why we simultaneously maintain multiple candidate learners and use the geometrical sequence with common ratio 2 as their epoch sizes.

It is worth noting that Besbes, Gur, and Zeevi (2015) have achieved the same result as Theorem 1, however, their algorithm requires knowing V_T ahead of time which is practically unavailable. By contrast, our method does not need the unknown function variation in advance, thus it can enjoy better applicability in practical scenarios. Besides, according to the minimax lower bound in (Besbes, Gur, and Zeevi 2015, Theorem 4), the result in Theorem 1 is also optimal up to logarithmic factors essentially.

Excess Risk

Although the function variation places some restrictions on the possible evolution of the underlying decision function, it still allows many different cases. In previous sections, we focus on the continuous variation case. Now we switch to a more specific situation when the underlying decision function stay the same most of time. Actually, this locally stationary environment appears quite naturally in practice. For example, think about the problem of product recommendation. It is often the case that the data from each month will stay stationary, and the changes only happen in the timing of month alternations. In such cases, comparing to a best fixed decision for each month is pretty reasonable.

Suppose $\sum_{t\in[T-1]} \mathbf{1}_{\boldsymbol{w}_t^\star \neq \boldsymbol{w}_{t+1}^\star} \leq K-1$ for some constant K, i.e., there are $K-1$ discrete changes in total. Divide the time horizon T into K epochs with equal size $\Delta = T/K$, and assume the k-th epoch is associated with a stationary distribution $\mathcal{D}_k$ for any $k \in [K]$. The following theorem demonstrates an excess risk bound for this situation.

Theorem 2. *Let $\bar{\boldsymbol{w}}_k$ and $\boldsymbol{w}_k^\star$ be the output and optimal classifiers of the k-th epoch, respectively. Define the risk by the sum of the ODM risk (regularized loss) of each epoch, i.e.,*

$$R(\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K) = \sum_{k\in[K]} \mathbb{E}_{\boldsymbol{x}}[f(\boldsymbol{w}_k; \boldsymbol{x})],$$

where $f(\boldsymbol{w}; \boldsymbol{x}) = \|\boldsymbol{w}\|^2/2 + \lambda\ell(\boldsymbol{w}; \boldsymbol{x})$. Then the following excess risk bound

$$R(\bar{\boldsymbol{w}}_1, \ldots, \bar{\boldsymbol{w}}_K) \leq R(\boldsymbol{w}_1^\star, \ldots, \boldsymbol{w}_K^\star) + \frac{G^2K^2}{\alpha T}\log(T/K)$$

holds in expectation, where $\bar{\boldsymbol{w}}_1, \ldots, \bar{\boldsymbol{w}}_K$ are returned by the algorithm and $\boldsymbol{w}_1^\star, \ldots, \boldsymbol{w}_K^\star$ are the optimal solution in each epoch under the batch learning setting.

Proof. Since the online ODM is performed in each epoch with a suitable step size, we have

$$\sum_{t\in\mathcal{E}_k} f(\boldsymbol{w}_t; \boldsymbol{x}_t) - \sum_{t\in\mathcal{E}_k} f(\boldsymbol{w}_k^\star; \boldsymbol{x}_t) \leq \frac{G^2}{\alpha}\log\Delta. \tag{5}$$

Table 1: Characteristics of experimental data sets collected in dynamic environments.

Data sets	#instance	#dimension	Data sets	#instance	#dimension
SEA	50,000	3	MG-2C-2D	200,000	2
SEA200G	90,000	10	GEARS-2C-2D	200,000	2
SEA500G	16,000	2	Chess	533	7
CIR500G	16,000	2	Usenet-1	1,500	100
SINE500G	16,000	2	Usenet-2	1,500	100
STA500G	16,000	2	Luxembourg	1,900	32
1CDT	55,283	2	Spam	9,324	500
1CHT	100,000	2	Weather	18,159	8
UG-2C-2D	200,000	3	Powersupply	29,928	2
UG-2C-3D	200,000	5	Electricity	45,312	8

By taking the expectation on the both sides, we obtain

$$\sum_{t\in\mathcal{E}_k} \mathbb{E}_{\boldsymbol{x}_t}[f(\boldsymbol{w}_t;\boldsymbol{x}_t)] - \sum_{t\in\mathcal{E}_k} \mathbb{E}_{\boldsymbol{x}_t}[f(\boldsymbol{w}_k^\star;\boldsymbol{x}_t)] \le \frac{G^2}{\alpha}\log\Delta.$$

Note that $\mathbb{E}_{\boldsymbol{x}_t}[f(\boldsymbol{w};\boldsymbol{x}_t)] = \mathbb{E}_{\boldsymbol{x}}[f(\boldsymbol{w};\boldsymbol{x})]$, thus

$$\mathbb{E}_{\boldsymbol{x}}\left[\sum_{t\in\mathcal{E}_k} f(\boldsymbol{w}_t;\boldsymbol{x})\right] \le \Delta\mathbb{E}_{\boldsymbol{x}}[f(\boldsymbol{w}^\star;\boldsymbol{x})] + \frac{G^2}{\alpha}\log\Delta.$$

According to the Jensen's inequality, we have $f(\bar{\boldsymbol{w}}_k;\boldsymbol{x}) \le \sum_{t\in\mathcal{E}_k} f(\boldsymbol{w}_t;\boldsymbol{x})/\Delta$. Therefore,

$$\mathbb{E}_{\boldsymbol{x}}[f(\bar{\boldsymbol{w}}_k;\boldsymbol{x})] \le \mathbb{E}_{\boldsymbol{x}}[f(\boldsymbol{w}^\star;\boldsymbol{x})] + \frac{G^2}{\alpha}\frac{\log\Delta}{\Delta}.$$

Summing over all the K epochs concludes the proof. □

Empirical Studies

In this section, we empirically evaluate the effectiveness of the proposed method. Specifically, we first present the data sets, and then introduce the experimental settings, followed by the compared baselines, and finally report the results.

Data sets

We adopt 12 synthetic data sets collected in dynamic environments, including sea, hyperplane, 1CDT, 2CDT, 1CHT, 2CHT, 1CSurr, UG-2C-2D, UG-2C-3D, UG-2C-5D, MG-2C-2D, and GEARS-2C-2D. Basic information is included in Table 1, and one may refer to (de Souza et al. 2015) for more details. Besides, to valid the efficacy of our proposed method in real applications, we further examine performance on 8 real data sets, including chess, usenet-1, usenet-2, Luxembourg, spam, whether, powersupply, and electricity. Note that the data set size ranges from 533 to more than 200,000, and the dimension ranges from 2 to 500, hence these data sets cover a broad range of properties.

Settings

Note that the standard cross-validation in the batch learning settings is not suitable here, due to inherent temporal relationships of the streaming data. Therefore, following the setup of previous works (Gama et al. 2014; Zhao et al. 2019b), for the data set with T instances, we select 10 different subsets with consecutive instances starting from $\{T/50, T/25, \dots, T/5\}$. All these subsets are with the same length $4T/5$. The experiments are conducted on these 10 subsets with various initializations, and the average and standard deviation of accuracy are reported as the final result.

Contenders

We compare the proposed dynamic ODM (D-ODM) with 5 contenders on both synthetic and real data sets, including (1) SVM-win: sliding window approach, the classifier is constantly updated by the nearest data samples in the window. Base classifiers are SVM (de Souza et al. 2015); (2) SVM-fix, batch implementation of SVM with a fixed window size (Syed, Liu, and Sung 1999); (3) SVM-ada, batch implementation of SVM with an adaptive window size (Klinkenberg 2004); (4) DWM, dynamic weighted majority algorithm, an adaptive ensemble based on the traditional weighted majority algorithm Winnow (Kolter and Maloof 2003; 2007); (5) AdaBoost.OL (Beygelzimer, Kale, and Luo 2015), an online adaptive boosting approach with nice theoretical guarantees.

Results

Table 2 summarizes detailed results obtained on twenty data sets. As can be seen, the overall performance of our method is superior or highly competitive to the other compared methods. Specifically, D-ODM performs significantly better than SVM-win / SVM-fix / SVM-ada / DWM / AdaBoost.OL on 18 / 19 / 20 / 14 / 19 over 20 data sets, and achieves the best accuracy on 17 data sets. In addition, comparing with other methods that do not consider margin distribution, D-ODM is always better or comparable, almost never worse than them, as shown in the win / tie / loss counts.

Conclusions

The maturing of margin theory, which disclosed the importance of margin distribution for generalization performance, gives rise to a promising research direction, i.e., the optimal margin distribution learning. Although the superiority of this new learning paradigm has been verified under batch learning settings, it remains open for online learning settings. Aware of this problem, we propose the dynamic optimal margin distribution machine and theoretically

Table 2: Performance comparisons in terms of mean and standard deviation of accuracy (%). The best accuracy on each data set is bolded. ●/○ indicates the performance of D-ODM is significantly better/worse than compared methods (paired t-tests at 95% significance level). The win/tie/loss counts for D-ODM are summarized in the last row.

Data sets	SVM-win	SVM-fix	SVM-ada	DWM	AdaBoost.OL	D-ODM
SEA	73.94±0.12●	86.19±0.06●	83.47±0.09●	87.04±0.03●	78.95±0.18●	**87.49±0.08**
hyperplane	83.74±0.03●	87.98±0.03●	81.94±0.07●	88.36±0.25●	72.73±0.14●	**89.29±0.12**
1CDT	98.71±0.05●	99.77±0.06●	99.77±0.08●	99.90±0.09●	99.52±0.08●	**99.98±0.10**
2CDT	94.86±0.06●	95.19±0.13●	95.18±0.15●	90.21±0.67●	94.11±0.18●	**95.99±0.09**
1CHT	98.75±0.18●	99.63±0.17●	99.63±0.18●	99.69±0.26●	99.27±0.50●	**99.90±0.05**
2CHT	87.70±0.04●	89.48±0.12●	88.89±0.13●	85.92±0.72●	81.23±0.14●	**90.05±0.08**
1CSurr	97.99±0.04●	94.24±1.08●	93.56±1.08●	96.31±0.50●	93.11±1.39●	**98.05±0.18**
UG-2C-2D	94.47±0.13●	95.41±0.10●	94.92±0.12●	95.59±0.11	94.69±0.13●	**95.99±0.11**
UG-2C-3D	93.60±0.73●	95.05±0.64	94.48±0.71●	95.14±0.62	94.31±0.69●	**95.58±0.15**
UG-2C-5D	74.82±0.45●	91.74±0.26●	90.37±0.35●	**92.82±0.23**○	89.84±0.38●	92.11±0.07
MG-2C-2D	**90.20±0.07**	84.98±0.06●	84.22±0.06●	90.15±0.06	85.03±0.02●	90.16±0.07
GEARS-2C-2D	95.54±0.01●	95.41±0.01●	95.26±0.02●	95.82±0.02	94.11±0.05●	**95.98±0.13**
Chess	69.67±1.51●	77.73±1.56●	69.18±3.65●	73.77±0.66●	78.39±2.34	**78.44±0.10**
Usenet-1	68.92±1.12	64.18±2.24●	67.68±1.86●	64.43±4.53●	65.03±1.31●	**68.98±0.06**
Usenet-2	74.44±0.71●	73.99±0.69●	72.64±0.84●	73.37±0.93●	70.56±0.93●	**75.77±0.05**
Luxembourg	88.57±0.28●	98.25±0.19●	97.43±0.42●	92.61±0.40●	89.12±0.97●	**98.98±0.16**
Spam	83.91±2.20●	92.44±0.80●	91.01±0.94●	91.49±1.09●	88.23±1.31●	**93.06±0.26**
Weather	68.54±0.55●	67.79±0.65●	77.26±0.33●	70.86±0.42●	71.20±0.41●	**79.11±0.29**
Powersupply	73.33±0.25●	71.17±0.15●	69.39±0.17●	72.18±0.29●	72.34±0.36●	**75.12±0.35**
Electricity	74.20±0.08●	62.01±0.59●	58.69±0.58●	**78.60±0.41**○	62.22±0.81●	76.80±0.42
D-ODM: w/t/l	18/2/0	19/1/0	20/0/0	14/4/2	19/1/0	

analyze its dynamic regret. We also derive an excess risk bound of the proposed method. In the future, we will further consider the *adaptive regret* (Hazan and Seshadhri 2009; Daniely, Gonen, and Shalev-Shwartz 2015) for the optimal margin distribution learning methods.

References

Besbes, O.; Gur, Y.; and Zeevi, A. J. 2015. Non-stationary stochastic optimization. *Operations Research* 63(5):1227–1244.

Beygelzimer, A.; Kale, S.; and Luo, H. 2015. Optimal and adaptive algorithms for online boosting. In *Proceedings of the 32nd International Conference on Machine Learning*, 2323–2331.

Cesa-Bianchi, N., and Lugosi, G. 2006. *Prediction, learning, and games*. Cambridge, UK: Cambridge University Press.

Cheng, F.; Zhang, J.; and Wen, C. 2016. Cost-sensitive large margin distribution machine for classification of imbalanced data. *Pattern Recognition Letters* 80(C):107–112.

Cortes, C., and Vapnik, V. N. 1995. Support-vector networks. *Machine Learning* 20(3):273–297.

Daniely, A.; Gonen, A.; and Shalev-Shwartz, S. 2015. Strongly adaptive online learning. In *Proceedings of the 32nd International Conference on Machine Learning*, 1405–1411.

de Souza, V. M. A.; Silva, D. F.; Gama, J.; and Batista, G. E. A. P. A. 2015. Data stream classification guided by clustering on nonstationary environments and extreme verification latency. In *Proceedings of the 2015 SIAM International Conference on Data Mining*, 873–881.

Freund, Y., and Schapire, R. E. 1995. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the 2nd European Conference on Computational Learning Theory*, 23–37.

Freund, Y., and Schapire, R. E. 1999. Large margin classification using the perceptron algorithm. *Machine Learning* 37(3):277–296.

Gama, J.; Zliobaite, I.; Bifet, A.; Pechenizkiy, M.; and Bouchachia, A. 2014. A survey on concept drift adaptation. *ACM Computing Surveys* 46(4):44:1–44:37.

Gao, W., and Zhou, Z.-H. 2013. On the doubt about margin explanation of boosting. *Artificial Intelligence* 203:1–18.

Hall, E. C., and Willett, R. 2013. Dynamical models and tracking regret in online convex programming. In *Proceedings of the 30th International Conference on Machine Learning*, 579–587.

Hazan, E.; Agarwal, A.; and Kale, S. 2007. Logarithmic regret algorithms for online convex optimization. *Machine Learning* 69(2-3):169–192.

Hazan, E., and Seshadhri, C. 2009. Efficient learning algorithms for changing environments. In *Proceedings of the 26th International Conference on Machine Learning*, 393–400.

Hazan, E. 2016. Introduction to online convex optimization. *Foundations and Trends in Optimization* 2(3-4):157–325.

Kivinen, J.; Smola, A. J.; and Williamson, R. C. 2001. Online learning with kernels. In *Advances in Neural Information Processing Systems*, 785–792.

Klinkenberg, R. 2004. Learning drifting concepts: Example selection vs. example weighting. *Intelligent Data Analysis* 8(3):281–300.

Kolter, J. Z., and Maloof, M. A. 2003. Dynamic weighted majority: A new ensemble method for tracking concept drift. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, 123–130.

Kolter, J. Z., and Maloof, M. A. 2007. Dynamic weighted majority: An ensemble method for drifting concepts. *Journal of Machine Learning Research* 8:2755–2790.

Li, Y., and Long, P. M. 1999. The relaxed online maximum margin algorithm. In *Advances in Neural Information Processing Systems*, 498–504.

Mokhtari, A.; Shahrampour, S.; Jadbabaie, A.; and Ribeiro, A. 2016. Online optimization in dynamic environments: Improved regret rates for strongly convex problems. In *Proceedings of the 55th IEEE Conference on Decision and Control*, 7195–7201.

Ou, G.; Wang, Y.; Pang, W.; and Coghill, G. M. 2017. Large margin distribution machine recursive feature elimination. In *The 4th International Conference on Systems and Informatics*, 1518–1523.

Reyzin, L., and Schapire, R. E. 2006. How boosting the margin can also boost classifier complexity. In *Proceedings of 23rd International Conference on Machine Learning*, 753–760.

Rosenblatt, F. 1958. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review* 65(6):386–407.

Schapire, R. E.; Freund, Y.; Barlett, P.; and Lee, W. S. 1998. Boosting the margin: a new explanation for the effectives of voting methods. *Annuals of Statistics* 26(5):1651–1686.

Schölkopf, B., and Smola, A. J. 2001. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. Cambridge, MA: MIT Press.

Shalev-Shwartz, S.; Singer, Y.; and Srebro, N. 2007. Pegasos: Primal estimated sub-gradient solver for svm. In *Proceedings of the 24th International Conference on Machine Learning*, 807–814.

Shalev-Shwartz, S. 2012. Online learning and online convex optimization. *Foundations and Trends in Machine Learning* 4(2):107–194.

Syed, N. A.; Liu, H.; and Sung, K. K. 1999. Handling concept drifts in incremental learning with support vector machines. In *Proceedings of the 5th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 317–321.

Vapnik, V. N. 1995. *The Nature of Statistical Learning Theory*. New York, NY: Springer.

Wang, L.; Sugiyama, M.; Jing, Z.; Yang, C.; Zhou, Z.-H.; and Feng, J. 2011. A refined margin analysis for boosting algorithms via equilibrium margin. *Journal of Machine Learning Research* 12:1835–1863.

Yang, T.; Zhang, L.; Jin, R.; and Yi, J. 2016. Tracking slowly moving clairvoyant: Optimal dynamic regret of online learning with true and noisy gradient. In *Proceedings of the 33rd International Conference on Machine Learning*, 449–457.

Zhang, T., and Zhou, Z.-H. 2014. Large margin distribution machine. In *Proceedings of the 20th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 313–322.

Zhang, T., and Zhou, Z.-H. 2017. Multi-class optimal distribution machine. In *Proceedings of the 34th International Conference on Machine Learning*, 4063–4071.

Zhang, T., and Zhou, Z.-H. 2018a. Optimal margin distribution clustering. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, 4474–4481.

Zhang, T., and Zhou, Z.-H. 2018b. Semi-supervised optimal margin distribution machines. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 3104–3110.

Zhang, T., and Zhou, Z.-H. 2019. Optimal margin distribution machine. *IEEE Transactions on Knowledge and Data Engineering*.

Zhang, L.; Yang, T.; Yi, J.; Jin, R.; and Zhou, Z.-H. 2017. Improved dynamic regret for non-degeneracy functions. In *Advances in Neural Information Processing Systems*, 732–741.

Zhao, P.; Wang, G.; Zhang, L.; and Zhou, Z.-H. 2019a. Bandit convex optimization in non-stationary environments. *arXiv preprint* abs/1907.12340.

Zhao, P.; Wang, X.; Xie, S.; Guo, L.; and Zhou, Z.-H. 2019b. Distribution-free one-pass learning. *IEEE Transaction on Knowledge and Data Engineering*.

Zhao, P.; Cai, L.-W.; and Zhou, Z.-H. 2019. Handling concept drift via model reuse. *Machine Learning*.

Zhao, P.; Hoi, S. C. H.; and Jin, R. 2011. Double updating online learning. *Journal of Machine Learning Research* 12:1587–1615.

Zhou, Y.-H., and Zhou, Z.-H. 2016. Large margin distribution learning with cost interval and unlabeled data. *IEEE Transactions on Knowledge and Data Engineering* 28(7):1749–1763.

Zhou, Z.-H. 2012. *Ensemble Methods: Foundations and Algorithms*. Boca Raton, FL: CRC Press.

Zhou, Z.-H. 2014. Large margin distribution learning. In *Proceedings of the 6th IAPR International Workshop on Artificial Neural Networks in Pattern Recognition*, 1–11.

Zinkevich, M. 2003. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning*, 928–936.