Robust Forward Algorithms via PAC-Bayes and Laplace Distributions

Asaf Noy
Department of Electrical Engineering
The Technion - Israel Institute of Technology
noyasaf@tx.technion.ac.il

Koby Crammer
Department of Electrical Engineering
The Technion - Israel Institute of Technology
koby@ee.technion.ac.il

Abstract

Laplace random variables are commonly used to model extreme noise in many fields, while systems trained to deal with such noises are often characterized by robustness properties. We introduce new learning algorithms that minimize objectives derived directly from PAC-Bayes bounds, incorporating Laplace distributions. The resulting algorithms are regulated by the Huber loss function and are robust to noise, as the Laplace distribution integrated large deviation of parameters. We analyze the convexity properties of the objective, and propose a few bounds which are fully convex, two of which jointly convex in the mean and standard-deviation under certain conditions. We derive new forward algorithms analogous to recent boosting algorithms, providing novel relations between boosting and PAC-Bayes analysis. Experiments show that our algorithms outperform AdaBoost, L1-LogBoost [10], and RobustBoost [11] in a wide range of input noise.

1 Introduction

Laplace random variables are commonly used as noise models in many fields such as signal processing, communication and control. Since the Laplace distribution decays exponentially from its mean, it is considered heavy-tailed compared to the Gaussian distribution, and used to model systems with anomalies such as extreme noise levels or outliers contamination. Systems trained to deal with these anomalies tend to be robust to such noise [30]. Robust statistics, pioneered, among others, by Peter Huber, is aimed for developing statistical methods that are not unduly affected by outliers. One of its key elements is the Huber loss function [18], extensively used in various applications [31] including robust filtering of Laplace-noise [2], as it allows the effect of outliers to be reduced while treating non-outliers in a relatively standard way.

PAC-Bayes bounds, introduced by McAllester [25], are a specific family of theoretical bounds that relates empirical performance of algorithms to their expected one. A few years later, Langford and Shawe-Taylor [23], and Herbirc [16] analyzed SVMs using PAC-Bayes bounds. Yet, there is still a gap between the statements emerging from the PAC-Bayes theory and algorithms that are actually analyzed by it. Specifically, to the best of our knowledge, no robust algorithms were analyzed nor derived via the PAC-Bayes framework.

In our work we use PAC-Bayes bounds based on Laplace-like distributions for developing new learning algorithms that possess appealing qualities, the foremost is outliers robustness. We investigate the key features of those in Sec. 3 and Sec. 4. We show a new connection between Laplace-noise and the Huber loss, paving the way for a better understanding of the relation between noise and robustness. In sec 5 we manage to analytically calculate the bound for the Laplace-like family of distributions, and prove that for separable training data, after a certain change of variables, the problem is in fact convex. In Sec. 6 we propose a function directly bounding the expected empirical error probability in the general case, and provide a condition and a modification, each is enough to ensure joint convexity. In Sec. 7 we first bound the zero-one loss with the LogLoss and then compute expectation over models. The result is a boosting-like algorithm similar to the LogitBoost, only regularized with the Huber function. The contribution of this result is twofold: it closes the gap between boosting algorithms and PAC-Bayes theory; and, develops a new boosting-like algo-

Appearing in Proceedings of the 17^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2014, Reykjavik, Iceland. JMLR: W&CP volume 33.

rithm which emerges from theory and naturally relates to Huber loss. This is as opposed to most boosting algorithms that are highly susceptible to outliers [24]. Experiments with synthetic and real-world datasets show that our algorithms outperform several boosting algorithms including AdaBoost [12], and other robust variants: L1-LogBoost [10] and RobustBoost [11].

2 Problem Setting and Background

We focus on binary classification, given a vector input $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d$, a classifier $h(\boldsymbol{x})$ outputs a single bit $y \in \mathcal{Y} = \{\pm 1\}$. We assume the existence of a joint distribution over the product space $\mathcal{X} \times \mathcal{Y}$, $(\boldsymbol{x}, y) \sim \mathcal{D}$, and restrict our discussion to linear functions, where $h(\boldsymbol{x}) = \text{sign}(\boldsymbol{\omega} \cdot \boldsymbol{x})$ for some d dimensional vector $\boldsymbol{\omega} \in \mathcal{H} \subseteq \mathbb{R}^d$. A classifier is evaluated using the expected loss, called risk, $\mathrm{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\left[\ell_{zo}(y(\boldsymbol{x}\cdot\boldsymbol{\omega}))\right]$, where the zero-one (or error) loss is defined to be $\ell_{zo}\left(y(\boldsymbol{\omega}\cdot\boldsymbol{x})\right) = 1$, if $y(\boldsymbol{\omega}\cdot\boldsymbol{x}) \leq 0$ and $\ell_{zo}\left(y(\boldsymbol{\omega}\cdot\boldsymbol{x})\right) = 0$ otherwise. The empirical risk is the average loss over the training set, i.e, $\frac{1}{m}\sum_{i=1}^{m}\ell_{zo}(y_i(\boldsymbol{x}_i\cdot\boldsymbol{\omega}))$. We focus in algorithms that receive a set $S = \{(\boldsymbol{x}_i, y_i)\}_1^m$ with m i.i.d. samples, $(\boldsymbol{x}_i, y_i) \sim \mathcal{D}$, and output a distribution over $\mathcal{H}$. We call this distribution *posterior* and denote it by Q. Also, such a distribution P over weights defined prior to observing the sample S is called *prior*.

Generalization theory relates or bounds the risk with the empirical risk. One such family of bounds is called PAC-Bayes bounds. McAllester [25, 26] introduced PAC-Bayesian analysis, which was later further refined [22, 29]. These bounds are aiming to analyze the performance of algorithms that output posterior distributions Q over functions $h \in \mathcal{H}$ and often are quite tight [16]. Our starting point is a theorem by Catoni [4] and Germain et al [14] which we now quote.

Theorem 1 (Cor. 2.2 [14], Thm. 1.2.1 [4]) *For any distribution $\mathcal{D}$, any set $\mathcal{H}$ of classifiers, any distributions P, Q of support $\mathcal{H}$, any $\delta \in [0,1]$, and any positive real scalar c, we have:*

$$\mathrm{E}_{\boldsymbol{\omega}\sim Q,(\boldsymbol{x},y)\sim\mathcal{D}}\left[\ell_{zo}(y(\boldsymbol{x}\cdot\boldsymbol{\omega}))\right] \leq \frac{1}{1-\exp(-c)}\times \quad (1)$$
$$\left[1-\exp\left\{-\frac{1}{m}\left(cE_{\boldsymbol{\omega}\sim Q}\left[\sum_{i=1}^{m}\ell_{zo}(y_i(\boldsymbol{x}_i\cdot\boldsymbol{\omega}))\right]\right.\right.\right.$$
$$\left.\left.\left.+\, D_{KL}(Q\|P)+\ln\frac{1}{\delta}\right)\right\}\right]$$

with probability of at least $1-\delta$.

The bound states that with high probability, the expected risk of the posterior is bounded by a monotonic function of the sum of the empirical risk and the Kullback-Leibler (KL) divergence between the posterior and prior distributions over the classifiers space. Equipped with this bound, we now describe and derive new algorithms that are aiming to minimize the risk by minimizing the PAC-Bayes bound, similarly to previous work such as that of Keshet et al [21].

3 A Laplace-like Family of Distributions

To employ PAC-Bayesian bounds we specify a family of distributions over elements $\boldsymbol{\omega} \in \mathcal{H}$. Most previous applications of this bound use the Gaussian distribution, that decays exponentially with the *squared* Euclidean norm. This causes the KL divergence in the bound to have a quadratic penalty in the difference between the prior mean and the posterior mean, forcing the later to be close to the former.

We thus propose a family of distribution over vectors $\boldsymbol{\omega}$ that depends on a generalized ℓ_1 distance of $\boldsymbol{\omega}$ from the mean $\boldsymbol{\mu}$. As we compute shortly in Remark 1, the KL divergence between two such distributions grows at most linearly in the difference between these mean vectors, allowing some elements of the posterior mean to be far from their respective prior elements.

$$Q(\boldsymbol{\omega};\boldsymbol{\mu},\boldsymbol{\sigma}) = \frac{1}{2^d\prod_{k=1}^{d}\sigma_k}e^{-\|\boldsymbol{\omega}-\boldsymbol{\mu}\|_{\boldsymbol{\sigma},1}} ,$$

where $\|\boldsymbol{\omega}\|_{\boldsymbol{\sigma},1} = \frac{1}{2d}\sum_{k=1}^{d}\frac{|\omega_k|}{\sigma_k}$. This is a uni-modal distribution with peak and mean at $\boldsymbol{\mu}$, and diagonal covariance $2\times\text{diag}(\sigma_1^2,..,\sigma_d^2)$. Since its entries are independent Laplace-distributed random variables (RV), we call this family *Laplace-like* (LL) distributions[1] and denote it by $\mathcal{L}^2$. An appealing quality of the $\mathcal{L}^2$ family is that for every given mean vector $\boldsymbol{\mu}$ and a bounded expected $\boldsymbol{\sigma}$-weighted ℓ_1-norm, $\mathrm{E}\left(\|\boldsymbol{\omega}-\boldsymbol{\mu}\|\right)_{\boldsymbol{\sigma},1} \leq 1$, the single continuous d-dimensional distribution $Q(\boldsymbol{\omega};\boldsymbol{\mu},\boldsymbol{\sigma})$ which maximizes the information-theoretic entropy maintains $Q \in LL$ [20], i.e LL is a maximum entropy family of distributions. Computing the KL divergence between $\mathcal{L}^2$ family members yields another unique quality.

Lemma 2 *Let $P\left(\boldsymbol{\mu}_P,\boldsymbol{\sigma}_P\right), Q\left(\boldsymbol{\mu}_Q,\boldsymbol{\sigma}_Q\right) \in \mathcal{L}^2$ be two LL distributions. The KL between these two distributions is well defined and given by,*

$$D_{KL}(Q\|P) = \sum_{k=1}^{d}\left[\frac{\sigma_{Q,k}}{\sigma_{P,k}}\left(\frac{|\mu_{Q,k}-\mu_{P,k}|}{\sigma_{Q,k}}+e^{-\frac{|\mu_{Q,k}-\mu_{P,k}|}{\sigma_{Q,k}}}\right)\right.$$
$$\left.+\log\left(\frac{\sigma_{P,k}}{\sigma_{Q,k}}\right)-1\right]. \quad (2)$$

[1] The family of distributions that are based on the ℓ_2 distance from the mean is called multivariate Laplace. Hence we use the different name: Laplace-like family.

The calculation appears in the App. A.

Remark 1 *We illustrate the properties of* (2) *in the* 1*-dimensional case. We denote by* $g_{\sigma_Q}(\mu_Q)$ *the KL divergence when setting* $\mu_P = 0$ *and* $\sigma_Q = \sigma_P$ *in* (2)*. We obtain,* $g_{\sigma_Q}(\mu_Q) = |\mu_Q|/\sigma_Q + \exp\{-|\mu_Q|/\sigma_Q\} - 1$ *. This function is roughly linear for large values of* $|\mu_Q|$ *since the exponent term vanishes then; while for small values we take a second order approximation of the exponent function, resulting in a quadratic function,* $g_{\sigma_Q}(\mu_Q) \approx |\mu_Q|/\sigma_Q + \left(1 - |\mu_Q|/\sigma_Q + \frac{1}{2}\left(|\mu_Q|/\sigma_Q\right)^2\right) - 1 = \mu_Q^2/\left(2{\sigma_Q}^2\right)$ *. These two properties are shared with the Huber loss function [18],* $h_a(x) = \frac{1}{2}x^2$ *if* $|x| \le a$*, and* $h_a(x) = a(|x| - a/2)$ *if* $|x| > a$*. Yet, while the Huber function can only be differentiated twice and is convex, our regularization function is smooth and strictly-convex. A plot of* $g_{\sigma_Q}(x)$ *for* $\sigma_Q = 1, 2$ *and* $h_a(x)$ *for* $a = 1, 0.5$ *is given in Fig. 1(a). Clearly, the parameter* σ*, similarly to the parameter* a *of the Huber function, controls the transition from quadratic to linear behavior in* μ_Q*. However,* σ_Q *has a probabilistic interpretation, strongly related to the bias-variance tradeoff, and in fact can be optimized as well.*

We note in passing that Huber [19] defined his *loss function* from a need to have robust estimator to outliers. It is mainly used as a *loss* (e.g. [3]). To the best of our knowledge, our derivation is the first that uses it (or a similar function) as a regularization, in general, and as a direct outcome of PAC-Bayes analysis, in particular. We expect that a learning algorithm equipped with such a regularization will perform well when there is a variance in the importance of weights associated with different features, as we shall see in the simulations below. This is because this regularization penalizes large weights linearly only, and not quadratically, as the squared norm does. We now proceed to analyze the expected loss with respect to LL distributions.

4 Expected Loss: Derivation and Analysis

In this section we restrict, for simplicity, our discussion to isotropic distributions[2], that is, $(\boldsymbol{\sigma}_Q)_k = \sigma_Q$ and $(\boldsymbol{\sigma}_P)_k = \sigma_P$ for $k = 1, \ldots, d$. We next compute the expected zero-one loss under a $\mathcal{L}^2$ distribution.

Lemma 3 *Assume the posterior is an isotropic Laplace-like distribution* $Q \sim \mathcal{L}^2\left(\boldsymbol{\mu}_Q, \sigma_Q\right)$*. The expected zero-one loss (i.e probability for an incorrect classification) of an example* $(\boldsymbol{x}, y)$*, with all elements of* $\boldsymbol{x}$ *differ from each other (that is* $x_k \neq x_j$ *for all* $k \neq j$*), is given by,*

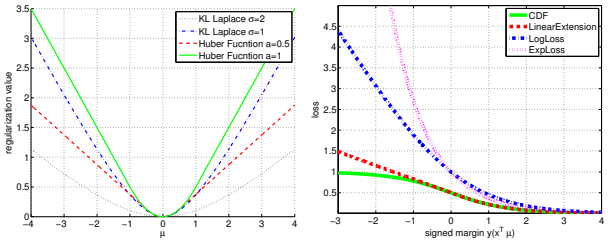
$$\mathrm{E}_{\boldsymbol{\omega}\sim Q}\left[\ell_{zo}\left(y\boldsymbol{\omega}\cdot\boldsymbol{x}\right)\right] = \ell_{cdf}\left(y\boldsymbol{\mu}\cdot\boldsymbol{x}\right)$$
$$= \begin{cases} \mathcal{E}(\boldsymbol{x}, \ y, \boldsymbol{\mu}_Q, \sigma_Q) & y\left(\boldsymbol{\mu}_Q\cdot\boldsymbol{x}\right) \ge 0 \\ 1 - \mathcal{E}(\boldsymbol{x}, -y, \boldsymbol{\mu}_Q, \sigma_Q) & y\left(\boldsymbol{\mu}_Q\cdot\boldsymbol{x}\right) < 0 \end{cases}, \quad (3)$$

where the first equality defines ℓ_{cdf} *and,*

$$\mathcal{E}(\boldsymbol{x}, y, \boldsymbol{\mu}_Q, \sigma_Q) = \mathcal{E}\left(\boldsymbol{x}, \frac{y\boldsymbol{x}\cdot\boldsymbol{\mu}_Q}{\sigma_Q}\right)$$
$$= \sum_{k=1}^{d} \alpha_k(\boldsymbol{x}) \exp\left\{-\frac{y\left(\boldsymbol{x}\cdot\boldsymbol{\mu}_Q\right)}{\sigma_Q|x_k|}\right\}, \quad (4)$$

and, define $\xi(\boldsymbol{x}) = sort(|\boldsymbol{x}|)$*, we have,* (3)

$$\alpha_k(\boldsymbol{x}) = \xi_k \left(\prod_{j=1}^{d}\xi_j\right)^{-2} \prod_{j=1, j\neq k}^{d}\left|\xi_j^{-1} - \xi_k^{-1}\right|^{-1}$$
$$\times \sum_{m=1}^{d}(-1)^{m+k}\left(\xi_k^{-1} + \xi_m^{-1}\right)^{-1} \prod_{j=1, j\neq m}^{d}\left|\xi_j^{-1} - \xi_m^{-1}\right|^{-1}.$$



(a) Regularization (b) Loss

Figure 1: (a) Illustrations of regularization compared with the Huber function. (b) The expected zero-one loss (CDF, green solid) and 3 upper bounds we develop and analyze.

The proof of the lemma appears in App. B. Notice that if the distribution over inputs $\boldsymbol{x} \sim \mathcal{D}$ is continuous, then the set $\{\alpha_k(\boldsymbol{x})\}$ is well-defined almost surely. We will deal with this case here for simplicity[3] . Notice also that about half of the coefficients $\alpha_k(\boldsymbol{x})$ are negative. Thus, it is not clear whether $\mathcal{E}(\boldsymbol{x}, y, \boldsymbol{\mu}_Q, \sigma_Q)$ for $y\left(\boldsymbol{\mu}_Q\cdot\boldsymbol{x}\right) \ge 0$ is even non-negative as a sum and *difference* of exponential functions. It will be shown that it is non-negative, and even strictly convex after we change its arguments below.

Substituting Lemma 2 and Lemma 3 in (1) we conclude that the PAC-Bayes bound for isotropic $P, Q \in$

[2]Generalization of the following for an arbitrary vector $\boldsymbol{\sigma}_Q$ is straightforward by replacing each example $\boldsymbol{x}$ with $\boldsymbol{x}'=(\sigma_{Q,1}x_1, \ldots, \sigma_{Q,d}x_d)$.

[3]The CDF is also well-defined and can be calculated for cases where $|x_{i,j}| = |x_{i,k}|$, by taking a limit and getting a distribution which is a mix of the one above with the Bilateral-Gamma distribution family.

$\mathcal{L}^2$ is monotonic in the following quantity,

$$\sum_{k=1}^{d} \frac{\sigma_Q}{\sigma_P}\left(\frac{|\mu_{Q,k}-\mu_{P,k}|}{\sigma_Q}+e^{\left\{-\frac{|\mu_{Q,k}-\mu_{P,k}|}{\sigma_Q}\right\}}\right)+d\log\left(\frac{\sigma_P}{\sigma_Q}\right)$$
$$-d+c\sum_{i=1}^{m}\begin{cases}\mathcal{E}(\boldsymbol{x}_i,\ \ y_i,\boldsymbol{\mu}_Q,\sigma_Q) & y_i\left(\boldsymbol{\mu}_Q\cdot\boldsymbol{x}_i\right)\geq 0\\ 1-\mathcal{E}(\boldsymbol{x}_i,-y_i,\boldsymbol{\mu}_Q,\sigma_Q) & y_i\left(\boldsymbol{\mu}_Q\cdot\boldsymbol{x}_i\right)<0\end{cases} \tag{5}$$

In other words, one strategy to obtain low risk is to minimize the bound, or equivalently (5). A common practice in machine learning is to derive convex objective functions, for which a local minima point is also a global one, often allowing us to better find a minimizer efficiently. Unfortunately, the objective of (5), as well as any PAC-Bayes objective in general, is not convex, since *any* CDF loss is concave for negative margin values (i.e. a majority-mistake).

Our goal is therefore to derive convex optimization problems and respective algorithms which would yield efficient algorithms to minimize (a surrogate of) the PAC-Bayes bound. Our first step towards this goal is a change of variables,

$$\boldsymbol{\mu}=\boldsymbol{\mu}_Q/\sigma_Q \quad , \quad \sigma=\sigma_Q/\sigma_P\ . \tag{6}$$

Conceptually, $\boldsymbol{\mu}$ is the mean normalized in standard-deviation units, and σ is the normalized standard-deviation in the units of the prior's standard-deviation. Additionally, we set the prior mean to zero, $\boldsymbol{\mu}_P=0$.

5 Separable Training Data

We first focus on the realizable case where the training data is linearly separable. We plug the new variables of (6) into the objective (5), assuming the margin is non-negative for all examples, we get the new objective for which the PAC-Bayes bound is monotonic in,

$$\mathcal{F}(\boldsymbol{\mu},\sigma;c)=-d\log\sigma e+\sigma\sum_{k=1}^{d}\left[|\mu_k|+\exp\left(-|\mu_k|\right)\right]$$
$$+c\sum_{i=1}^{m}\ell\left(y_i\boldsymbol{x}_i\cdot\boldsymbol{\mu}\right)\ , \tag{7}$$

for $\ell\left(y_i\boldsymbol{x}_i\cdot\boldsymbol{\mu}\right)=\sum_{k=1}^{d}\alpha_{i,k}\exp\left\{-\frac{y_i(\boldsymbol{x}_i\cdot\boldsymbol{\mu})}{|x_{i,k}|}\right\}$. The next theorem states that under the separability assumption, the objective of (7) is in fact convex.

Theorem 4 *Assume that the set $\{(\boldsymbol{x}_i,y_i)\}_{i=1}^m$ is linearly separable. Then, $\mathcal{F}(\boldsymbol{\mu},\sigma;c)$ is strictly-convex (separately) in $\boldsymbol{\mu},\sigma$, for $\ell\left(y_i\boldsymbol{x}_i\cdot\boldsymbol{\mu}\right)=\sum_{k=1}^{d}\alpha_{i,k}\exp\left\{-\frac{y_i(\boldsymbol{x}_i\cdot\boldsymbol{\mu})}{|x_{i,k}|}\right\}$.*

The full proof of Theorem 4 appears in App. C. We now briefly discuss a general scheme for solving the following optimization problems, which are separately convex in $\boldsymbol{\mu}$ and σ, using alternate optimization. Fixing σ, one can use any algorithm to minimize the convex problem in $\boldsymbol{\mu}$. Fixing the minimizing value for $\boldsymbol{\mu}$, the only term of the objective (7) (and also (10) below) depending on σ is the regularization, for which the solution is obtained by setting the derivative to zero,

$$\sigma^*=\frac{d}{\sum_{k=1}^{d}|\mu_k|+\exp\left(-|\mu_k|\right)}\ . \tag{8}$$

Note that the solution satisfies $\sigma^*\in(0,1)$ because $|\mu_k|+\exp\left(-|\mu_k|\right)\geq 1$ holds for all $k=1,\ldots,d$. Furthermore, σ is related to the standard deviation of the posterior, while $\boldsymbol{\mu}-\mathbf{0}$ is the posterior's-mean bias from the prior's mean. Thus, equation (8) can be thought of a bias-variance tradeoff. When the optimal solution of the mean $\boldsymbol{\mu}$ is far from zero (the prior's mean), the optimal posterior is also concentrated about its mean, and vice versa, if the posterior's mean is close to the prior's mean (low bias), the variance is large, compared with the prior's variance.

We conclude this section by stating a bound on the optimal parameters norm, providing intuition concerning the solutions as well as useful for minimization algorithms, as we shall see in the experiments section.

Lemma 5 *Let $\ell(z)$ be a non-negative loss of the margin z, for which $\ell(z=0)=\eta$ for some $\eta>0$. The minimizer $(\boldsymbol{\mu}^*,\sigma_Q{}^*)$ of (7) satisfies, $d\,e^{cm\eta/d}\geq\|\boldsymbol{\mu}^*\|_1$ and $\sigma^*\geq e^{-cm\eta/d}$.*

The proof appears in App. E. According to Lemma 4, one can observe a natural way of determining the hyper-parameter c for our algorithm to be of the order $c=\Theta\left(d/(m\eta)\right)$, in which case $\|\boldsymbol{\mu}^*\|_1=O(d)$ and $\sigma^*=\Omega(1)$. Simulation results in Sec. 8 support this choice.

6 Non-Separable Training Data

As an upper bounded monotone increasing function over the reals, cumulative distribution functions are not convex in general, and thus direct minimizations of PAC-Bayes bounds in general, and our objective of (5) in particular, are not convex as well.

There are two possible approaches. In this section, we upper bound the expected zero-one loss, i.e. the CDF defined in (3), with a convex function, that is, we bound $\mathrm{E}_{\boldsymbol{\omega}\sim Q}\left[\ell_{zo}\left(y\boldsymbol{\omega}\cdot\boldsymbol{x}\right)\right]=\ell_{cdf}\left(y\boldsymbol{\omega}\cdot\boldsymbol{x}\right)\leq\mathrm{Bound}_1\left(y\boldsymbol{\omega}\cdot\boldsymbol{x}\right)$. In the next section, we bound the zero-one loss with convex functions, and then take the expectation of the bounding loss, that is, $\mathrm{E}_{\boldsymbol{\omega}\sim Q}\left[\ell_{zo}\left(y\boldsymbol{\omega}\cdot\boldsymbol{x}\right)\right]\leq\mathrm{E}_{\boldsymbol{\omega}\sim Q}\left[\mathrm{Bound}_2\left(y\boldsymbol{x}\cdot\boldsymbol{\omega}\right)\right]$. The

next theorem holds for the bound introduced in this section, but is ,in fact, much more general.

Theorem 6 *Let $\ell(y\boldsymbol{x}\cdot\boldsymbol{\mu})$ be an arbitrary convex loss. Denote by $\mathcal{F}(\boldsymbol{\mu},\sigma;c)$ the objective of (7). Then, (1) $\mathcal{F}(\boldsymbol{\mu},\sigma;c)+\frac{1}{2}\|\boldsymbol{\mu}\|^2$ is jointly strictly-convex in $(\boldsymbol{\mu},\sigma)$. (2) $\mathcal{F}(\boldsymbol{\mu},\sigma;c)$ is jointly strictly-convex in $(\boldsymbol{\mu},\sigma)$ for $\|\boldsymbol{\mu}\|_\infty\leq 1$.*

The proof appears in App. F. We now derive and analyze a convex bound on the expected loss, which together with our last analysis ensures convexity of the objective for the entire range of margins, and not only for the separable case. This theorem can be applied in other contexts as there is no restriction about the loss function, other than being convex.

As mentioned above, the CDF is convex for positive margin values, and concave otherwise. Therefore we bound the concave part with the tightest convex upper bound: a linear function,

$$\ell_{lin}(y\boldsymbol{x}\cdot\boldsymbol{\mu})=\begin{cases}\sum_{k=1}^{d}\alpha_k(\boldsymbol{x})e^{\left\{-\frac{y\boldsymbol{x}\cdot\boldsymbol{\mu}}{|x_k|}\right\}} & y\boldsymbol{x}\cdot\boldsymbol{\mu}\geq 0\\ \frac{1}{2}-\beta(\boldsymbol{x})y\boldsymbol{x}\cdot\boldsymbol{\mu} & y\boldsymbol{x}\cdot\boldsymbol{\mu}<0\end{cases}, \quad (9)$$

for $\beta(\boldsymbol{x})=\sum_{k=1}^{d}\frac{\alpha_k(\boldsymbol{x})}{|x_k|}>0$. We also denote $\beta_i=\beta(\boldsymbol{x}_i)$. The bound is illustrated with a dashed-red line in Fig. 1(b). The function ℓ_{lin} is convex, as its second derivative is always non-negative, and by construction satisfy $\ell_{cdf}(z)\leq\ell_{lin}(z)$. The objective of (5), after changing variables using (6), can be bounded with,

$$\mathcal{F}_{lin}(\boldsymbol{\mu},\sigma;c)=-d\log\sigma+\sigma\sum_{k=1}^{d}\left[|\mu_k|+\exp\left\{-|\mu_k|\right\}\right]+c\sum_{i=1}^{m}\ell_{lin}(y\boldsymbol{x}_i\cdot\boldsymbol{\mu})\ . \quad (10)$$

The function $\mathcal{F}_{lin}(\boldsymbol{\mu},\sigma;c)$ is convex in $\boldsymbol{\mu}$ and σ (individually): The loss depends only in $\boldsymbol{\mu}$ and its second derivative with respect to $\boldsymbol{\mu}$ is non-negative. The regularization is a function of both and we already showed in Theorem 4 that it is indeed convex in each individually. From the same reason the function is *jointly convex* under the conditions of Theorem 6.

This function has an additional appealing property: the regularization is now equivalent to the loss term over additional artificial examples, as stated in the next lemma. A similar property exists in some boosting algorithms [9].

Lemma 7 *Define $2d$ artificial example set, $\mathcal{A}=\{(\boldsymbol{e}_k,y):k=1,\ldots,d,y\in\mathcal{Y}\}$, where $\boldsymbol{e}_k$ is the kth standard basis vector ($(\boldsymbol{e}_k)_j=1$ for $k=j$ and $(\boldsymbol{e}_k)_j=0$ otherwise). The regularization term $\sum_{k=1}^{d}\left[|\mu_k|+\exp\left\{-|\mu_k|\right\}\right]$ of $\mathcal{F}_{lin}(\boldsymbol{\mu},\sigma;c)$ defined above, equals (up to an additive constant d) to twice the linear-loss over the $2d$ additional-examples, $2\sum_{(\boldsymbol{x},y)\in\mathcal{A}}\ell_{lin}(y\boldsymbol{x}_i\cdot\boldsymbol{\mu})$.*

The proof appears in App. G. Alternating minimization of the linear bound, using (8) for setting σ and coordinate descent for optimizing over $\boldsymbol{\mu}$, leads to a robust and efficient learning algorithm which we call **RobuCoP**, for a Robust coordinate PAC-Bayes algorithm (pseudo-code appears in App. H. However, since the Hessian of (10) is vanishing for weights $\boldsymbol{\mu}$ with large values, second order optimization methods may not work well in practice. This motivates us to employ the approach presented next.

7 Bounding the Zero-One Loss

Our main quantity of interest is the empirical expected zero-one loss computed for a single example in (3). In general, as discussed above, it is not convex, and thus there is a need to bound it. In the last section we derived a linear bound of the expected loss. Here, we take a complementary approach, and instead of bounding the expected zero-one loss, we compute an exact exception of losses, each bounding the zero-one loss. We define the LogLoss and ExpLoss, $\ell_{log}(y(\boldsymbol{\omega}\cdot\boldsymbol{x}))=\log_2(1+\exp(-y(\boldsymbol{\omega}\cdot\boldsymbol{x})))$ and $\ell_{exp}(y(\boldsymbol{\omega}\cdot\boldsymbol{x}))=\exp(-y(\boldsymbol{\omega}\cdot\boldsymbol{x}))$. It is well known that, $\ell_{zo}(y(\boldsymbol{\omega}\cdot\boldsymbol{x}))\leq\ell_{log}(y(\boldsymbol{\omega}\cdot\boldsymbol{x})),\ell_{exp}(y(\boldsymbol{\omega}\cdot\boldsymbol{x}))$, therefore, $\mathrm{E}_{\omega\sim Q}\left[\ell_{zo}(y(\boldsymbol{\omega}\cdot\boldsymbol{x}))\right]\leq\mathrm{E}_{\omega\sim Q}\left[\ell_{log}(y(\boldsymbol{\omega}\cdot\boldsymbol{x}))\right]\leq\log_2(1+\mathrm{E}_{\omega\sim Q}\left[\ell_{exp}(y(\boldsymbol{\omega}\cdot\boldsymbol{x}))\right])$, where the last inequality follows from Jensen inequality.

Let us compute the expected ExpLoss. In this section we do assume nothing about the data. Additionally, we return to general distributions with vector parameter $\boldsymbol{\sigma}_Q$, and as we shall see shortly, there is no need to employ the change of variables of (6). Given an input series, we scale it in order to bound it in the unit ℓ_∞-ball, that is $\max_{i=1}^{m}\|\boldsymbol{x}_i\|_\infty<1$. Let, $Q\sim\mathcal{L}^2(\boldsymbol{\mu}_Q,\boldsymbol{\sigma}_Q)$, be a distribution with bounded variances $0<\sigma_{Q,k}\leq 1$ for $k=1,\ldots,d$ (otherwise, the expected ExpLoss is not bounded from above). Then,

$$\mathrm{E}_Q\left[e^{-y\boldsymbol{x}\cdot\boldsymbol{\omega}}\right]=\prod_{k=1}^{d}\frac{1}{2\sigma_{Q,k}}\left[\frac{e^{-y\boldsymbol{x}\cdot\boldsymbol{\mu}}}{\sigma_{Q,k}^{-1}-yx_k}+\frac{e^{-y\boldsymbol{x}\cdot\boldsymbol{\mu}}}{yx_k+\sigma_{Q,k}^{-1}}\right]=e^{-y\boldsymbol{x}\cdot\boldsymbol{\mu}}\prod_{k=1}^{d}\frac{1}{1-(x_k\sigma_{Q,k})^2}\ . \quad (11)$$

We denote for simplicity, $D_i=D_i(\boldsymbol{\sigma}_Q)=\prod_{k=1}^{d}\left[1/\left(1-(x_{i,k}\sigma_{Q,k})^2\right)\right]$, for each example $(\boldsymbol{x}_i,y_i)$. It is well defined since both $\|\boldsymbol{\sigma}_Q\|_\infty\leq 1$ and $\|\boldsymbol{x}_i\|_\infty<1$. We use the last equation to provide a bound of (the loss term of) (5) using the LogLoss. As

Input: Training set $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$, $\boldsymbol{\mu}_P \in \mathbb{R}^d$, $\boldsymbol{\sigma}_Q \in (0,1)^d$, $c > 0$, No. of iterations T.
Initialization: $\boldsymbol{\mu}_Q^{(1)} = \boldsymbol{\mu}_P$; $D_i = \prod_{k=1}^d \left[1 - (\boldsymbol{x}_{i,k}\boldsymbol{\sigma}_{Q,k})^2\right]^{-1}$ for $i = 1, \ldots, m$
Loop For $t = 1, \ldots, T$ do:

- Choose coordinate $k \in \{1, .., d\}$
- Set: $\gamma_k^+ = \sum_{i=1, y_i \boldsymbol{x}_{i,k} \geq 0}^m \frac{D_i |x_{i,k}|}{D_i + e^{y_i \boldsymbol{x}_i \cdot \boldsymbol{\mu}_Q^{(t)}}}$; $\gamma_k^- = \sum_{i=1, y_i \boldsymbol{x}_{i,k} < 0}^m \frac{D_i |x_{i,k}|}{D_i + e^{y_i \boldsymbol{x}_i \cdot \boldsymbol{\mu}_Q^{(t)}}}$
- Update: If $\left(\gamma_k^+ \exp\left\{\frac{2\mu_{Q,k}^{(t)}}{\sigma_{Q,k}}\right\} \geq \gamma_k^-\right)$

  then $\mu_{Q,k}^{(t+1)} \leftarrow \mu_{Q,k}^{(t)} + \sigma_{Q,k} \log\left(\frac{1+\sqrt{1+4c\gamma_k^-\left[\exp\left\{-\frac{\mu_{Q,k}^{(t)}}{\sigma_{Q,k}}\right\}+c\gamma_k^+\right]}}{2c\gamma_k^-}\right)$

  else $\mu_{Q,k}^{(t+1)} \leftarrow \mu_{Q,k}^{(t)} + \sigma_{Q,k} \log\left(\frac{1+\sqrt{1+4c\gamma_k^+\left[\exp\left\{\frac{\mu_{Q,k}^{(t)}}{\sigma_{Q,k}}\right\}+c\gamma_k^-\right]}}{2\left[\exp\left\{\frac{\mu_{Q,k}^{(t)}}{\sigma_{Q,k}}\right\}+c\gamma_k^-\right]}\right)$

Output: $\boldsymbol{\mu}_Q^{(T+1)}$

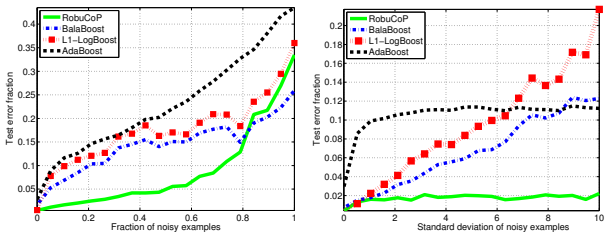
Figure 2: The BaLaBoost Algorithm.

before, $(\boldsymbol{\mu}_P, \boldsymbol{\sigma}_P) = (\mathbf{0}, \mathbf{1})$. The LogLoss bound is,

$$\mathcal{F}_{\log}\left(\boldsymbol{\mu}_Q, \boldsymbol{\sigma}_Q\right) = \sum_{k=1}^d \left(\sigma_{Q,k} e^{-\frac{|\mu_{Q,k}|}{\sigma_{Q,k}}} - \log\left(\sigma_{Q,k}\right)\right) + \|\boldsymbol{\mu}_Q\|_1 + c\sum_{i=1}^m \log_2\left(1 + D_i e^{-y\boldsymbol{x}_i \cdot \boldsymbol{\mu}_Q}\right) . \quad (12)$$

By computing the second derivatives of $\mathcal{F}_{\log}\left(\boldsymbol{\mu}_Q, \boldsymbol{\sigma}_Q\right)$ defined in (12), one can verify that this function is convex separately in $\boldsymbol{\mu}$ and $\sigma_{Q,k}$ for each k. We now derive a coordinate-descent, boosting-like algorithm for (12). A similar algorithm can be also derived when replacing the LogLoss with the ExpLoss (also convex, omitted due to lack of space). We focus on the LogLoss as it is more involved, and was found to be more robust to outliers [5]. We fix $\boldsymbol{\sigma}_Q$ for now, and optimize over $\boldsymbol{\mu}_Q$. The remaining part of this section is devoted to the derivation of an efficient boosting-like algorithm, minimizing (12) iteratively over coordinates. On each iteration t, a single coordinate $\mu_{Q,k}^{(t)}$ is chosen (others [5, 10] fixed a subset of coordinates) and modified by $\delta_k^{(t)} = \mu_{Q,k}^{(t+1)} - \mu_{Q,k}^{(t)}$.

The next lemma builds on Sec. 2 of Duchi and Singer [10], proposing the L1-LogBoost algorithm,



(a) Test error vs. p (b) Test error vs. σ

Figure 3: (a) Test error of four algorithms vs the fraction p of noisy examples for $\sigma = 2$ (b) Test error of four algorithms vs the standard deviation σ of noise for $p = 0.1$.

aiming to minimize a $l1$-regularized version of the LogitBoost algorithm [27]. In addition to the smooth Huber-like regularization on Sec. 3, a key difference lies in the incremental change, naturally determined by $\sigma_{Q,k}$, instead of using a vector from an a-priori fixed set of templates, supplied as input for the algorithm.

Lemma 8 *The difference between* (12) *evaluated at time* t *and time* $t+1$ *is lower bounded,*

$$\mathcal{F}_{\log}(\boldsymbol{\mu}_Q^{(t)}, \boldsymbol{\sigma}_Q) - \mathcal{F}_{\log}(\boldsymbol{\mu}_Q^{(t+1)}, \boldsymbol{\sigma}_Q) \geq \left|\mu_{Q,k}^{(t)}\right| + \sigma_{Q,k} e^{-\frac{\left|\mu_{Q,k}^{(t)}\right|}{\sigma_{Q,k}}} - \left|\mu_{Q,k}^{(t)} + \delta_k^{(t)}\right| - \sigma_{Q,k} e^{-\frac{\left|\mu_{Q,k}^{(t)} + \delta_k^{(t)}\right|}{\sigma_{Q,k}}} + c\sigma_{Q,k}\left(\gamma_k^+\left[1 - e^{-\frac{\delta_k^{(t)}}{\sigma_{Q,k}}}\right] + \gamma_k^-\left[1 - e^{\frac{\delta_k^{(t)}}{\sigma_{Q,k}}}\right]\right) .$$

where we denote, $\gamma_k^+ = \sum_{i=1, y_i x_{i,k} \geq 0}^m q_t(i)\,|x_{i,k}|$, $\gamma_k^- = \sum_{i=1, y_i x_{i,k} < 0}^m q_t(i)\,|x_{i,k}|$ *and* , $q_t(i) = D_i / \left(D_i + e^{y_i \boldsymbol{x}_i \cdot \boldsymbol{\mu}_Q^{(t)}}\right)$.

The proof appears in App. H. In order to *boost* the rate of convergence to the minimizer of $\mathcal{F}_{\log}$, we wish to maximize the last incremental lower bound at each step. Equivalently, omitting terms independent of $\delta_k^{(t)}$, we can minimize the following objective,

$$\arg\min_{\delta_k^{(t)}} \left[\left|\mu_{Q,k}^{(t)} + \delta_k^{(t)}\right| + \sigma_{Q,k} e^{-\frac{\left|\mu_{Q,k}^{(t)} + \delta_k^{(t)}\right|}{\sigma_{Q,k}}} + c\sigma_{Q,k}\left(\gamma_k^+ e^{-\frac{\delta_k^{(t)}}{\sigma_{Q,k}}} + \gamma_k^- e^{\frac{\delta_k^{(t)}}{\sigma_{Q,k}}}\right)\right] . \quad (13)$$

The next lemma allows us to eliminate the absolute values in the above terms.

Lemma 9 *The optimizer of* (13), $\delta_k^{(t)}$, *satisfies,*

$$\text{sign}\left(\mu_{Q,k}^{(t)} + \delta_k^{(t)}\right) = \text{sign}\left(\gamma_k^+ \exp\left(\frac{2\mu_{Q,k}^{(t)}}{\sigma_{Q,k}}\right) - \gamma_k^-\right) .$$

The proof appears in App. J. We use this lemma to solve (13) by considering two cases according to the sign of $\gamma_k^+ \exp\left(2\mu_{Q,k}^{(t)}/\sigma_{Q,k}\right) - \gamma_k^-$. We derive here the

Algorithm	TIM1	TIM2	Mgc	Wine	VJ	VJ2
RobuCop	**94.3**	**92.4**	**82.2**	99.4	**92.0**	**91.6**
BaLaBoost	92.6	91.9	79.6	98.9	91.9	89.0
AdaBoost	81.3	83.9	66.6	95.3	90.5	89.2
RobustBoost	92.5	92.1	80.6	97.3	90.5	89.2
GentleBoost	92.5	91.5	77.8	95.3	90.9	87.5
SVM	91.9	91.6	77.6	**99.5**	91.7	87.3
l1-LogBoost	91.1	90.5	69.2	98.6	91.1	87.0
LogitBoost	91.0	89.9	69.2	98.5	91.1	86.7

Table 1: Algorithms' comparison

case where the sign is positive, and omit the case where it's negative. We set to zero the derivative of (13) , and solve for the incremental change $\delta_k^{(t)} = \mu_{Q,k}^{(t+1)} - \mu_{Q,k}^{(t)}$,

$$\delta_k^{(t)} = \sigma_{Q,k} \log \left(\frac{1+\sqrt{1+4c\gamma_k^- \left[e^{\mu_{Q,j}^{(t)}/\sigma_{Q,k}} + c\gamma_k^+ \right]}}{2c\gamma_k^-} \right) \quad (14)$$

After updating $\boldsymbol{\mu}$, the optimal value of $\boldsymbol{\sigma}_Q$ can be found by minimizing (12).

We call our algorithm **BaLaBoost** For PAC-Bayesian Laplace Boosting algorithm. Pseudocode is given in Fig. 2. We note that, as some boosting algorithms [10, 27], BaLaBoost may not enjoy the weak-to-strong learnability properties of AdaBoost. One can view this algorithm as a coordinate-optimization algorithm over the weights. We compare (14) to Eq. (4) of Duchi and Singer [10]: $\sigma_{Q,k}$ is analogous to the template parameter a_k, yet it has a natural interpretation and withal, can be *optimized*. The two updates are similar, as both employ a lower bound technique on the change of the losses values, yet differ, since not derived from the same regularization. One can view (14) as having *adaptive* learning rate, via the optimization of $\sigma_{Q,k}$, while their algorithm has *fixed* learning rate. This may yield better solutions for data with some features having larger (additive) noise than others.

8 Experiments

We first demonstrate the properties of the algorithms with synthetic data (see App. K for further details). Four algorithms are evaluated: AdaBoost [12], L1-LogBoost [10], RobuCoP of Sec. 6, BalaBoost of Sec. 7. Parameters for the later three algorithms were set once after optimizing over additional training and test sets. Note, the value of c set for RobuCoP is aligned with the estimated value suggested by Lemma 5, that is $\frac{2d}{m}$. We point out that L1-LogBoost is a regularized version of LogitBoost, that is considered robust to various kinds of noises [27].

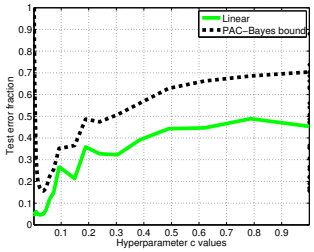


Figure 5: Test error of RobuCoP (solid green) and the PAC-Bayes bound (dotted black).

Fig. 5 shows the generalization error as measured on a test set compared with the PAC-Bayes bound. Clearly the trend of the bound follows the trend of the test error, and additionally, its values are below 1 and the gap between the bound and measured test-error is low compared to other similar bounds. Thus the bound provides relatively accurate estimate of the test error, which supports the approach of PAC-Bayes bounds optimization. Additionally the value of c minimizing the bound is also the minimizer of the test-error, i.e. it can be used to set the hyperparameter.

Fig. 3(a) shows the test error for a value of $\sigma = 2$ with fraction of noisy examples ranging between $p = 0 \ldots 1$. Here AdaBoost performs the worst (it is sensitive to noise), then L1-LogBoost. Our two algorithms perform the best: RobuCoP for low values of noise, while BalaBoost for high values. Fig. 3(b) shows the test error for a fixed fraction on noisy training set $p = 0.1$ with standard deviation ranges from $\sigma = 1 \ldots 10$. As before AdaBoost and l1-LogBoost perform the worst, where AdaBoost performs better for higher values of noise. As before, BalaBoost performs better than both algorithms, and RobuCoP outperforms it (except for very low noisy values). We emphasize the fact that RobuCop's performance is not degrading even for very high noise levels.

We also evaluated the described algorithms, together with RobustBoost [11] (five algorithms altogether), on the task of vowel recognition from a set of 8 possibilities used in the VJ (vocal joystick) project, broken into 28 binary classification problems (one-vs-one). Details appear in App. K. The results are shown in Fig. 4. As with the synthetic data, BaLaBoost and RobuCop achieve better results than the 3 other classifiers when dealing with lower noise levels, while RobuCop performs significantly better than the other 4 classifiers for high noise levels over all the data sets. We hypothesize that as other algorithms that maintain a distribution, RobuCop and BaLaBoost optimize their hyperparameters according to the noise levels already in the tuning phase, absorbing the noise level via the prior distributions optimization. An empirical evidence supports this hypothesis is the fact that the optimizers of the standard deviation of the LL distribution are typically monotone-increasing with the level noise for all data sets. We also hypothesize that RobuCop per-

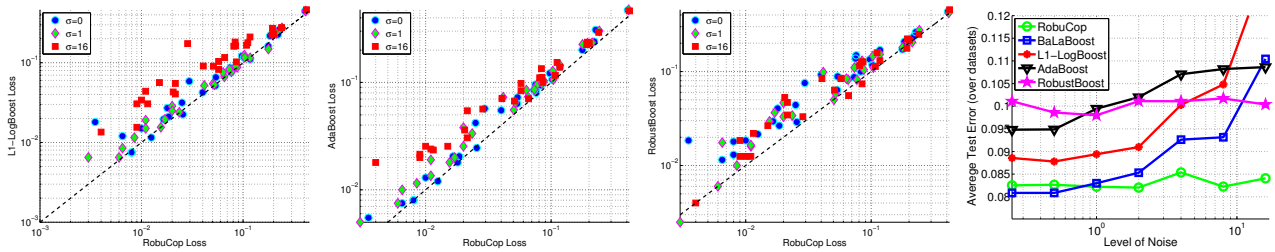


Figure 4: Three left panels: test error of RobuCoP vs L1-LogBoost, Adaboost and RobustBoost respectively, each calculated over 28 different data sets and 3 levels of noise. Right panel: classifiers' average error over 28 data sets for 8 values of noise levels.

forms the best under most noise conditions as it minimizes a tighter bound over the test error. However, it suffers from a relatively slow training time compared with BaLaBoost, which achieved the fastest running time among all the described algorithms. Thus, BaLaBoost balances nicely performance (noisy data) and training time. RobustBoost also performed relatively well when high noise levels were presented, but suffered from a major deviation in performance, often providing poor results compared to the rest. We conjecture that this happens because RobustBoost's potential function is non-convex , while the rest minimize a convex loss, hence more stable over various scenarios.

9 Related Work

Support vector machines [6] were analyzed using PAC-Bayes bounds in a few contexts [23, 16]. They used Gaussian distributions with fixed isotropic covariance matrices, inducing the Euclidean norm-regularization. Another approach [1] is to split the training set into two parts, using the first to learn data-dependent prior, and the second to learn the posterior. To the best of our knowledge, our work is among the first to tie Boosting and PAC-Bayes bounds, and in particular the ExpLoss was induced naturally from the distribution we employed (see (4)).

There are few previous approaches that are close in spirit to our approach of minimizing PAC-Bayes bounds. Germain et al [15] derived a specialized loss [13] plugged in a PAC-Bayes bound, and used quasi-uniform distributions over a finite set of classifiers. They also proposed to bound the KL between prior and posterior explicitly using ℓ_p norms. In our work, we induce a regularization term by setting a specific distribution (LL) over weights, which *naturally* induces an Huber-like regularization term. Later, Roy et al [28] used similar quasi-uniform distributions to induce quadratic programs that minimize the variance of the margin. Crammer et al [7] described the Gaussian margin machine algorithm, induced from a PAC-Bayes bound, combined with a Gaussian distribution, where the learning algorithm seeks not only for the mean parameters, but also the covariance matrix. They proposed a hinge-like loss, that was not convex, and a stochastic optimization procedure. Another approach proposed recently [8] is to derive an algorithm from a PAC-Bayes bound, based on uniform distribution over compact non-finite set of weights. They proposed to obtain a convex objective by bounding the expectation of the (uniform) posterior with a maximal value over all weights with non-zero (and equal) probability. The resulting problem was solved using stochastic gradient and proximal algorithms.

10 Summary

We derived novel learning algorithms by plugging a Laplace-like distribution over weights into a PAC-Bayes bound, and analyzed the conditions for which they can be solved efficiently. There is no clear winner in terms between the algorithms. **RobuCop** minimizes the tightest convex function bounding the CDF, and performs the best in our experiments, yet it does not have an analytic update per coordinate, and it is convex but not strictly-convex (for each $(\boldsymbol{\mu}, \sigma)$). **BaLaBoost** is more general in terms of inputs and choice of $\boldsymbol{\sigma}$. Experiments demonstrate the merits of our methods compared to other well established boosting algorithms, and specifically they are more robust to various input noise: fraction of noisy input or standard deviation of the noise. Finally, we also derived a quadratic-bound of (7) (similar to the derivations of Sec. 6) which is looser but easier to implement, and an algorithm based on the ExpLoss (rather than the LogLoss). Both derivations and algorithms are omitted due to lack of space. We plan to further explore the characteristics of noisy classification and its relationship to robustness, and investigate the conditions for which one algorithm outperforms the other.

Acknowledgments: The authors thank Assaf Hallak for his enlightening insights. This research was funded in part by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI).

References

[1] Amiran Ambroladze, Emilio Parrado-Hernández, and John Shawe-Taylor. Tighter pac-bayes bounds. In *NIPS*, pages 9–16, 2006.

[2] Aleksandr Y Aravkin, Bradley M Bell, James V Burke, and Gianluigi Pillonetto. An l1-laplace robust kalman smoother. *IEEE Transactions on Automatic Control*, 56(12):2898, 2011.

[3] Peter Buhlmann and Torsten Hothorn. Boosting algorithms: Regularization, prediction and model fitting. *Statistical Science*, 22(4), April 2007.

[4] O. Catoni. *PAC-Bayesian surpevised classication: the thermodynamics of statistical learning*. Institute of Mathematical Statistics, 2007.

[5] M. Collins, R.E. Schapire, and Y. Singer. Logistic regression, AdaBoost and Bregman distances. *Machine Learning*, 47(2/3):253–285, 2002.

[6] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20, 1995.

[7] Koby Crammer, Mehryar Mohri, and Fernando Pereira. Gaussian margin machines. *JMLR - Proceedings Track*, 5, 2009.

[8] Koby Crammer and Tal Wagner. Volume regularization for binary classification. In *NIPS2012*.

[9] Ofer Dekel, Shai Shalev-Shwartz, and Yoram Singer. Smooth e-intensive regression by loss symmetrization. In *COLT*, pages 433–447, 2003.

[10] John Duchi and Yoram Singer. Boosting with structural sparsity. *ICML 26*, 2009.

[11] Yoav Freund. A more robust boosting algorithm. *arXiv preprint arXiv:0905.2138*, 2009.

[12] Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Euro-COLT 2*, 1995.

[13] Pascal Germain, Alexandre Lacasse, François Laviolette, and Mario Marchand. A pac-bayes risk bound for general loss functions. In *NIPS*, pages 449–456, 2006.

[14] Pascal Germain, Alexandre Lacasse, François Laviolette, and Mario Marchand. Pac-bayesian learning of linear classifiers. In *ICML*, 2009.

[15] Pascal Germain, Alexandre Lacasse, François Laviolette, Mario Marchand, and Sara Shanian. From pac-bayes bounds to kl regularization. In *NIPS*, 2009.

[16] R. Herbrich and T. Graepel. A pac-bayesian margin bound for linear classifiers. *IEEE Trans. Inf. Theor.*, 48(12):3140–3150, September 2006.

[17] Roger A. Horn and Charles R. Johnson. *Topics in matrix analysis*. Cambridge University Press, Cambridge, 1991.

[18] Peter Huber. *Robust Statistics*. Wiley, 1974.

[19] Peter J. Huber. Robust estimation of a location parameter. *Annals of Mathematical Statistics*, 35(1):73–101, March 1964.

[20] E. T. Jaynes. Information theory and statistical mechanics. *Phys. Rev.*, 106, 1957.

[21] Joseph Keshet, David A. McAllester, and Tamir Hazan. Pac-bayesian approach for minimization of phoneme error rate. In *ICASSP*, 2011.

[22] J. Langford and M. Seeger. Bounds for averaging classifiers, 2002.

[23] J. Langford and J. Shawe-Taylor. PAC-bayes and margins. In *NIPS*, 2002.

[24] Philip M Long and Rocco A Servedio. Random classification noise defeats all convex potential boosters. *Machine Learning*, 78(3):287–304, 2010.

[25] D. McAllester. PAC-Bayesian model averaging. In *Proceedings of COLT*, 1999.

[26] D. McAllester. PAC-Bayesian stochastics model selection. *MLJ*, 5, 2003.

[27] I. Eckley R. McDonald, D. Hand. An empirical comparison of three boosting algorithms on real data sets with artificial class noise. 2003.

[28] Jean-Francis Roy, François Laviolette, and Mario Marchand. From pac-bayes bounds to quadratic programs for majority votes. In *ICML*, 2011.

[29] M. Seeger. PAC-Bayesian generalization bounds for gaussian processes. *JMLR*, 3:233–269, 2002.

[30] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *The 25th international conference on Machine learning*, 2008.

[31] You-Gan Wang, Xu Lin, and Zhidong Bai. Robust estimation using the huber function with a data-dependent tuning constant. *Journal of Computational and Graphical Statistics*, 16(2), 2007.