Distributed Boosting Classifiers over Noisy Channels

Yongjune Kim*, Yuval Cassuto†, and Lav R. Varshney‡§
*DGIST, Daegu, South Korea, yjk@dgist.ac.kr
†Technion – Israel Institute of Technology, Haifa, Israel, ycassuto@ee.technion.ac.il
‡University of Illinois at Urbana-Champaign, Urbana, IL, USA, varshney@illinois.edu
§Salesforce Research, Palo Alto, CA, USA, lvarshney@salesforce.com

***Abstract*—We present a principled framework to address resource allocation for realizing boosting algorithms on substrates with communication noise. Boosting classifiers (e.g., AdaBoost) make a final decision via a weighted vote from local decisions of many base classifiers (weak classifiers). Suppose the base classifiers' outputs are communicated over noisy channels; these noisy outputs will degrade the final classification accuracy. We show this degradation can be effectively reduced by allocating more system resources for more important base classifiers. We formulate resource optimization problems in terms of importance metrics for boosting. Moreover, we show that the optimized noisy boosting classifiers can be more robust than bagging for noise during inference (test stage). We provide numerical evidence to demonstrate the benefits of our approach.**

I. Introduction

Boosting methods in machine learning construct a set of base (weak) classifiers and then classify a new data point by taking a *weighted* vote of their decisions [1]. Boosting can achieve good classification accuracy even if the base classifiers have performance that is only slightly better than random guessing [2], [3]. Adaptive boosting (AdaBoost) is the most widely used form of boosting [2], [4]; it works well for classification problems such as face detection [5] and can be extended to regression problems [6].

Consider the standard supervised classification problem. For the given training set $S = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, the objective of learning is to estimate the unknown classification function $f(\cdot)$. The input vector is $\mathbf{x}_n = (x_{n,1}, \ldots, x_{n,D})$ where D denotes the dimension of the input vectors. The output variables y_n are typically drawn from a discrete set of classes, i.e., $y \in \{1, \ldots, K\}$ where K denotes the number of classes. For a binary classification problem, we set $y \in \{+1, -1\}$, which we focus on.

The final output of AdaBoost is as follows:

$$f(\mathbf{x}) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t f_t(\mathbf{x})\right), \tag{1}$$

where $f_t(\mathbf{x})$ and α_t denote the local decision and the coefficient of the t-th base classifier, respectively. The final output (decision) $f(\mathbf{x})$ is the weighted voting from local decisions as shown in (1). AdaBoost assigns larger coefficients to more accurate (or important) base classifiers [3], [4]. Unlike AdaBoost, the coefficients of bagging classifiers are uniform (i.e., $\alpha_t = \frac{1}{T}$ for all $t \in \{1, \ldots, T\}$) and the output of bagging corresponds to majority voting [7].

Y. Kim was supported by the DGIST Start-up Fund Program of the Ministry of Science and ICT 2020090013. L. R. Varshney was supported in part by the National Science Foundation under Grant CCF-1717530. Y. Cassuto was supported in part by the US-Israel Binational Science Foundation and the Israel Science Foundation.

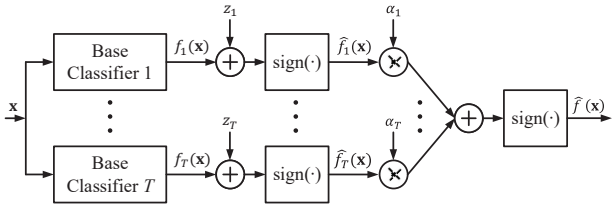


Fig. 1. Noisy boosting classifier. The final output of noisy boosting classifier $\widehat{f}(\mathbf{x})$ is given by (2).

Suppose that the outputs of base classifiers are corrupted by random noise as shown in Fig. 1. The noise $\mathbf{z} = (z_1, \ldots, z_T)$ captures *communication* errors over the channel between the base classifiers and the weighted voter. Alternatively, $\mathbf{z}$ can originate from noise in the *computation* hardware of base classifiers [8]. The corrupted output of the t-th base classifier is denoted by $\widehat{f}_t(\mathbf{x}) \in \{+1, -1\}$. We assume that the weighted vote is implemented in a noiseless manner.

We observe that noise in the individual base classifiers affects the overall classification accuracy in a way that strongly depends on the coefficient vector $\boldsymbol{\alpha}$. That is, an erroneous $\widehat{f}_t$ with a large coefficient α_t is more likely to corrupt the final classification output than a base classifier with a smaller coefficient. Following this observation, we develop a principled framework to optimize the classification accuracy by allocating system resources to base classifiers according to their importance prescribed in the coefficient vector $\boldsymbol{\alpha}$.

In this paper, we assume that the impact of $\mathbf{z}$ can be controlled by allocating system resources. Ideally, the system resources should be allocated to minimize the classification error probability. However, the classification error probability of boosting depends on the data sets and base classifiers (and their training algorithms); the classification error probability is not simply related to the system resources and does not yield tractable optimization procedures.

To circumvent this problem, we minimize *proxies* instead

of the classification error probability. First, we define three proxies: 1) Markov proxy, 2) Chernoff proxy, and 3) Gaussian proxy. Next, we formulate optimization problems to minimize these proxies for a given resource budget. This kind of indirect approach is effective in many problems, e.g., [9]–[11].

In particular, we investigate an example where the outputs of base classifiers are corrupted by additive noise over the channels between base classifiers and the aggregator (weighted voter). Here, the noise level over these channels can be controlled by allocating transmit power. We show the proposed framework can effectively reduce the classification error probability for a given transmit-power budget. Our approach provides a general framework to allocate a limited resource for boosting classifiers and can also be applied to settings of noisy computations. For example, the quality of computations on noisy hardware can be changed by controlling supply voltage [8], replicating computations [12], [13], and implementing granular bit precisions [10]. Based on the proposed framework, we can optimize these system resources in a principled manner.

Our problem of noisy AdaBoost is distinct from AdaBoost in the presence of noisy labels. A well-known model of random classification noise (RCN) assumes that each label y in the training set is flipped independently [14], [15]. Several studies have investigated the behavior of AdaBoost under label noise and proposed more robust training algorithms [16]–[18]. Note that the *data* noise affects all base classifiers during training; hence, it affects the AdaBoost model (i.e., base classifiers and their coefficients) permanently. Our model assumes that the *system* noise during inference (test stage) affects the decisions of base classifiers independently. We optimize system resources to mitigate the noise impact without altering the trained AdaBoost models.

It is well known that classification accuracy of AdaBoost tends to degrade more than that of bagging for the RCN model [16], [19]. The reason is that AdaBoost more aggressively fits noisy instances in the training set [15], [16]. Contrarily, we show that AdaBoost can be more robust than bagging in our problem setting where noise flips the base classifiers' outputs during inference (test). This is because the accuracy improvement by the proposed optimization is more effective as the coefficient variability increases, or formally, as the geometric mean of the coefficients decreases.

The rest of this paper is organized as follows. Section II explains the noisy AdaBoost model. Section III develops three metrics for the importance of base classifiers originating from three optimization problems. Section IV formulates and solves resource-allocation problems based on these importance metrics. Section V provides numerical results and Section VI concludes.

II. Noisy AdaBoost Model

A. AdaBoost

AdaBoost trains the base classifiers in sequence to minimize an exponential error function [3], [4]. Each base classifier is trained using a weighted form of the training set in which the data weights $\mathbf{w} = (w_1, \ldots, w_N)$ depend on the performance of previous base classifiers. In particular, data points that are misclassified by one of the base classifiers are given greater weight when used to train the next base classifier. Once all base classifiers have been trained, their outputs are combined through weighted voting [3].

Note that the data weights $\mathbf{w} = (w_1, \ldots, w_N)$ are distinct from the classifier coefficients $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_T)$. AdaBoost determines both values during training. Once training is done, only the coefficients $\boldsymbol{\alpha}$ are used to classify new data points. The training of AdaBoost is described in Algorithm 1. The indicator function $\mathcal{I}(f_t(\mathbf{x}_n) \neq y_n)$ equals 1 if $f_t(\mathbf{x}_n) \neq y_n$ and 0 otherwise.

Algorithm 1 Training of AdaBoost for binary classification [3]

1: Initialize the data weights $\mathbf{w}$ by setting $w_n^{(1)} = \frac{1}{N}$.
2: **for** $t = 1 : T$ **do**
3: Fit a base classifier $f_t(\mathbf{x})$ to the training set by minimizing $J_t = \sum_{n=1}^{N} w_n^{(t)} \mathcal{I}(f_t(\mathbf{x}_n) \neq y_n)$.
4: Evaluate $\varepsilon_t = \frac{\sum_{n=1}^{N} w_n^{(t)} \mathcal{I}(f_t(\mathbf{x}_n) \neq y_n)}{\sum_{n=1}^{N} w_n^{(t)}}$.
5: Compute $\alpha_t = \log \frac{1-\varepsilon_t}{\varepsilon_t}$.
6: Update $w_n^{(t+1)} = w_n^{(t)} \exp\{\alpha_t \mathcal{I}(f_t(\mathbf{x}_n) \neq y_n)\}$.
7: **end for**
8: **return** the trained base classifiers $\{f_t(\cdot)\}$ and the corresponding coefficients $\boldsymbol{\alpha}$ for $t \in \{1, \ldots, T\}$.

The classification error probability of the trained model $f(\cdot)$ is given by $P_{e,f} = \Pr(f(\mathbf{x}) \neq y)$ where y is the true label corresponding to $\mathbf{x}$.

Remark 1 (Positive Coefficients): If a base classifier is better than random guessing, then $\alpha_t > 0$ for any $t \in \{1, \ldots, T\}$ [2].

Remark 2 (Normalized Coefficients): We normalize the coefficients such that $\sum_{t=1}^{T} \alpha_t = 1$. Note that normalization does not affect the classification output in (1).

B. Noisy AdaBoost

Suppose that the base classifiers' outputs may be flipped due to the noise z_t, i.e., $f_t(\mathbf{x}) \neq \widehat{f}_t(\mathbf{x})$ where $\widehat{f}_t(\mathbf{x}) = \operatorname{sign}(f_t(\mathbf{x}) + z_t)$ as shown in Fig. 1. The mismatch event of the t-th base classifier is denoted by $\delta_t = \mathcal{I}(f_t(\mathbf{x}) \neq \widehat{f}_t(\mathbf{x}))$. Then, we can define the base classifiers' mismatch probabilities as $\mathbf{p} = (p_1, \ldots, p_T)$ where $p_t \triangleq \Pr(f_t(\mathbf{x}) \neq \widehat{f}_t(\mathbf{x})) = \mathbb{E}[\delta_t]$. In the sequel, the expectation over the distribution of $\mathbf{x}$ will be replaced by the empirical mean over the given data set.

The final output of noisy AdaBoost is given by

$$\widehat{f}(\mathbf{x}) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t \widehat{f}_t(\mathbf{x})\right). \tag{2}$$

Then, the final mismatch probability (i.e., mismatch probability of the final output) is given by

$$P_m \triangleq \Pr(f(\mathbf{x}) \neq \widehat{f}(\mathbf{x})), \tag{3}$$

which captures the negative impact of $\mathbf{z}$ on the final classification accuracy. We can expect that the final mismatch

probability P_m depends on the base classifiers' mismatch probabilities $\mathbf{p}$.

The classification error probability of the noisy AdaBoost is upper bounded by

$$P_e = P_{e,\widehat{f}} \leq P_{e,f} + P_m, \tag{4}$$

where $P_{e,f}$ denotes the classification error probability of noise-free AdaBoost. Note that $P_{e,f}$ solely depends on the AdaBoost algorithm and the dataset, i.e., $P_{e,f}$ is independent of $\mathbf{z}$. Hence, we focus on P_m to reduce the deleterious impact of $\mathbf{z}$.

III. Importance Metrics of Base Classifiers

We define three proxies to the mismatch probability: 1) Markov proxy, 2) Chernoff proxy, and 3) Gaussian proxy. These proxies induce different importance metrics of base classifiers. We provide theoretical justification for the proxies and the corresponding metrics.

A. Markov Proxy

Let us define the *Markov proxy*, which comes from Markov's inequality.

Definition 3 (Markov Proxy): The Markov proxy $\widehat{p}_M$ of the mismatch probability is given by

$$\widehat{p}_M = \sum_{t=1}^{T} \alpha_t p_t, \tag{5}$$

which is the nonnegative weighted sum of p_t.

We derive an upper bound on the mismatch probability P_m based on Markov's inequality and show that this upper bound can be lowered by minimizing the Markov proxy $\widehat{p}_M$.

Theorem 4 (Upper Bound by Markov's Inequality): The mismatch probability of $\mathbf{x}_n$ is upper bounded as follows:

$$P_m(\mathbf{x}_n) \leq \frac{2\widehat{p}_M}{\gamma_n}, \tag{6}$$

where

$$\gamma_n = \left| \sum_{t=1}^{T} \alpha_t f_t(\mathbf{x}_n) \right|, \tag{7}$$

which represents the decision margin of $\mathbf{x}_n$. Then, an upper bound on the mismatch probability P_m is given by $P_m \leq \left(\frac{2}{N} \sum_{n=1}^{N} \frac{1}{\gamma_n} \right) \cdot \widehat{p}_M$.

Proof: The proof is given in [20]. ∎

Higher decision margin γ_n and/or lower Markov proxy $\widehat{p}_M$ reduce the upper bound on the mismatch probability. The margin γ_n depends only on the input vector $\mathbf{x}_n$ and the trained AdaBoost classifier model. In contrast, $\widehat{p}_M$ depends on $\mathbf{z}$, whose distribution we can control by resource allocation; hence minimizing $\widehat{p}_M$ is pursued in Section IV.

Remark 5: For a given dataset and trained AdaBoost classifier model, the upper bound (6) depends only on the Markov proxy $\widehat{p}_M$. Hence, our objective is to minimize $\widehat{p}_M$ by controlling $\mathbf{p} = (p_1, \ldots, p_T)$.

B. Chernoff Proxy

We define the *Chernoff proxy* via the Chernoff bound.

Definition 6 (Chernoff Proxy): The Chernoff proxy $\widehat{p}_C$ of the mismatch probability is given by

$$\widehat{p}_C(s) = \sum_{t=1}^{T} \left(e^{s\alpha_t} - 1 \right) p_t, \tag{8}$$

where $s > 0$ is a parameter.

Remark 7: Since $\alpha_t > 0$ and $s > 0$, $e^{s\alpha_t} - 1 > 0$.

We derive an upper bound on the mismatch probability P_m from the Chernoff bound and show that this upper bound can be reduced by minimizing the Chernoff proxy $\widehat{p}_C$.

Theorem 8 (Upper Bound by Chernoff Bound): The mismatch probability is upper bounded as follows:

$$P_m \leq \mathbb{E}\left[\exp\left(-s \cdot \frac{\gamma_n}{2} \right) \right] \cdot \exp\left(\widehat{p}_C(s) \right), \tag{9}$$

for any $s > 0$. Note that $\mathbb{E}\left[\exp\left(-s \cdot \frac{\gamma_n}{2} \right) \right]$ can be calculated by $\mathbb{E}\left[\exp\left(-s \cdot \frac{\gamma_n}{2} \right) \right] = \frac{1}{N} \sum_{n=1}^{N} \exp\left(-s \cdot \frac{\gamma_n}{2} \right)$.

Proof: The proof is given in [20]. ∎

This upper bound can be tightened by minimizing the Chernoff proxy $\widehat{p}_C$. Also, similarly to the Markov proxy, a higher decision margin γ_n decreases the upper bound. In addition, s should be carefully chosen because of a trade-off relation between $\mathbb{E}\left[\exp\left(-s \cdot \frac{\gamma_n}{2} \right) \right]$ and $\widehat{p}_C$. A smaller s decreases $\widehat{p}_C$ while increasing $\mathbb{E}\left[\exp\left(-s \cdot \frac{\gamma_n}{2} \right) \right]$.

C. Gaussian Proxy

As in Definition 3 and Definition 6, we define a third proxy metric via Gaussian approximation.

Definition 9 (Gaussian Proxy): The Gaussian proxy $\widehat{p}_G$ of the mismatch probability is given by

$$\widehat{p}_G = \sum_{t=1}^{T} \alpha_t^2 p_t. \tag{10}$$

Suppose $\widehat{f}(\mathbf{x}_n) = \text{sign}(\widehat{g}(\mathbf{x}_n))$ where $\widehat{g}(\mathbf{x}_n)$ is given by

$$\widehat{g}(\mathbf{x}_n) = \sum_{t=1}^{T} \alpha_t \widehat{f}_t(\mathbf{x}_n) = \pm\gamma_n + v_n, \tag{11}$$

where we set $\pm\gamma_n$ and v_n as *signal* term and the *noise* term, respectively. The signal term γ_n follows from (7). The noise term v_n is $v_n = -2\left(\sum_{t \in \mathcal{T}_n^+} \alpha_t \delta_{t,n} - \sum_{t \in \mathcal{T}_n^-} \alpha_t \delta_{t,n} \right)$ where $\mathcal{T}_n^+ = \{t \mid f_t(\mathbf{x}_n) = 1\}$ and $\mathcal{T}_n^- = \{t \mid f_t(\mathbf{x}_n) = -1\}$, respectively. Also, $\delta_{t,n} = \mathcal{I}(f_t(\mathbf{x}_n) \neq \widehat{f}_t(\mathbf{x}_n))$.

Theorem 10: The noise term v_n for $n \in \{1, \ldots, N\}$ can be modeled as a Gaussian distribution, i.e., $v_n \sim \mathcal{N}(\mu_v, \sigma_v^2)$ by the central limit theorem. Then,

$$\mu_v = -2\left(\sum_{t \in \mathcal{T}_n^+} \alpha_t p_t - \sum_{t \in \mathcal{T}_n^-} \alpha_t p_t \right), \tag{12}$$

$$\sigma_v^2 = 4 \sum_{t=1}^{T} \alpha_t^2 p_t (1 - p_t). \tag{13}$$

TABLE I
COMPARISON OF IMPORTANCE METRICS OF BASE CLASSIFIERS

Proxy	Importance metric β	Remarks
Markov	α	Definition 3
Chernoff	$e^{s\alpha}-1$	Definition 6
Gaussian	α^2	Definition 9

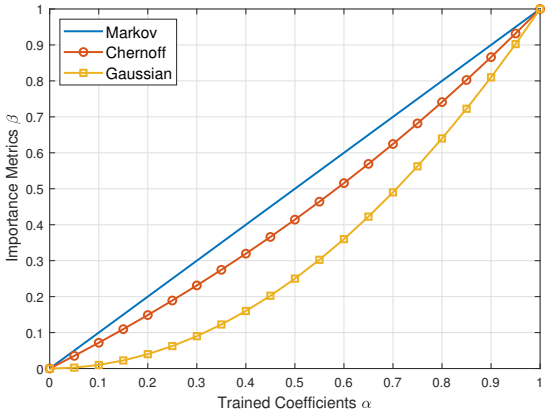


Fig. 2. Comparison of importance metrics β in Table I (with $s = \log 2$ for the Chernoff proxy).

Proof: The proof is given in [20]. ∎

Observe that the variance in (13) is *data independent*, and thus its minimization by resource allocation is an effective way to reduce the classification error probability. Based on the Gaussian approximation, we can derive an estimate of the mismatch probability P_m.

Corollary 11: An estimate of the mismatch probability is

$$P_m(\mathbf{x}_n) = Q\left(\frac{\gamma_n - \mu_v}{\sigma_v}\right) \approx Q\left(\frac{\gamma_n - \mu_v}{2\sqrt{\widehat{p}_G}}\right), \tag{14}$$

where $\sigma_v^2 \approx 4\widehat{p}_G$ for $p_t \ll 1$ (i.e., $p_t^2 \ll p_t$). Note that $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty \exp\left(-\frac{u^2}{2}\right) du$.

The minimum μ_v and the minimum $\widehat{p}_G$ are desired to reduce the estimate of mismatch probability. However, μ_v depends on the input vector $\mathbf{x}_n$, hence, we cannot easily minimize μ_v. In contrast, the Gaussian proxy depends only on the trained $\boldsymbol{\alpha}$ and the base classifiers' mismatch probability $\mathbf{p}$. Thus, we minimize the Gaussian proxy to reduce the mismatch probability.

Each of the three proxies can be described by $\sum_{t=1}^T \beta_t p_t$ where β_t denotes the *importance metric* of the t-th base classifier. Table I lists the importance metrics for the three proxies. Fig. 2 plots the dependence of each importance metric on α. It illustrates how resource allocation based on β would give preference to a larger α.

Remark 12: The importance metrics are positive (i.e., $\beta_t > 0$) because of $\alpha_t > 0$ (Remark 1) and $e^{s\alpha_t} - 1 > 0$ (Remark 7).

IV. RESOURCE ALLOCATION FOR NOISY ADABOOST

A. Formulation of Optimization Problems

We investigate optimization approaches to determine the optimal $\mathbf{p} = (p_1, \ldots, p_T)$ for a given resource constraint. By optimizing the proposed proxies, we attempt to reduce the mismatch probability, i.e., reduce the noise impact on classification accuracy.

An important assumption is that the mismatch probabilities of base classifiers can be controlled by allocating the system resources. Suppose that the mismatch probability of the base classifier p_t can be described by resource r_t, i.e., $p_t = p(r_t)$. Then, we can formulate the following optimization problem for a given resource budget $\mathcal{C}$:

$$\begin{aligned} \underset{\mathbf{r}}{\text{minimize}} \quad & \sum_{t=1}^T \beta_t p(r_t) \\ \text{subject to} \quad & \sum_{t=1}^T c(r_t) \le \mathcal{C} \end{aligned} \tag{15}$$

where the objective function depends on the importance metric $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_T)$. Also, $c(r_t)$ denotes the cost of the allocated resource to the t-th base classifier.

If $p(r_t)$ and $c(r_t)$ are *convex*, then the optimization problem (15) is also *convex* because β_t is positive for all t in any of the three proxies (Remark 12). In such cases, for the Markov proxy and the Gaussian proxy, we can obtain the optimal resource allocation by solving (15) directly using convex programming. For the Chernoff proxy (8), due to the free parameter s, we propose an iterative algorithm to jointly find the optimal s and $\mathbf{p}$ (see [20, Algorithm 2]).

B. Example: Communication Power Allocation

Suppose that $f_t(\mathbf{x}) \in \{+1, -1\}$ is transmitted using a symbol from $\{r_t, -r_t\}$, which is corrupted by the noise z_t as in Fig. 1. We assume that the additive noise can be modeled as Gaussian distribution, i.e., $z_t \sim \mathcal{N}(0, \sigma_t^2)$. Then $p(r_t) = Q\left(\sqrt{\mathsf{SNR}_t}\right) = Q\left(\frac{r_t}{\sigma_t}\right)$ where the signal-to-noise ratio (SNR) corresponding to the t-th base classifier is $\mathsf{SNR}_t = \frac{r_t^2}{\sigma_t^2}$. Hence, p_t can be controlled by allocating transmit power $c(r_t) = r_t^2$. Then, the optimization problem (15) will be

$$\begin{aligned} \underset{\mathbf{r}}{\text{minimize}} \quad & \sum_{t=1}^T \beta_t Q\left(\frac{r_t}{\sigma_t}\right) \\ \text{subject to} \quad & \sum_{t=1}^T r_t^2 \le \mathcal{C}, \quad r_t \ge 0, \quad t = 1, \ldots, T \end{aligned} \tag{16}$$

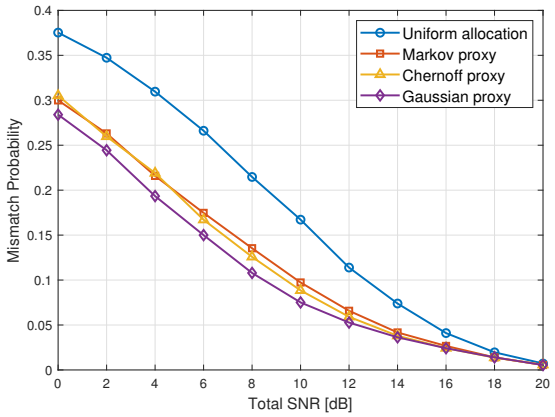
where $\mathcal{C}$ represents the total power budget.

Remark 13: The power allocation problem (16) is a convex optimization problem since $p(r_t) = Q(r_t)$ is convex for $r_t \ge 0$. Note that $\frac{d^2Q(x)}{dx^2} = \frac{x}{\sqrt{2\pi}}\exp\left(-\frac{x^2}{2}\right) \ge 0$. The optimal solution of (16) is derived in [20].

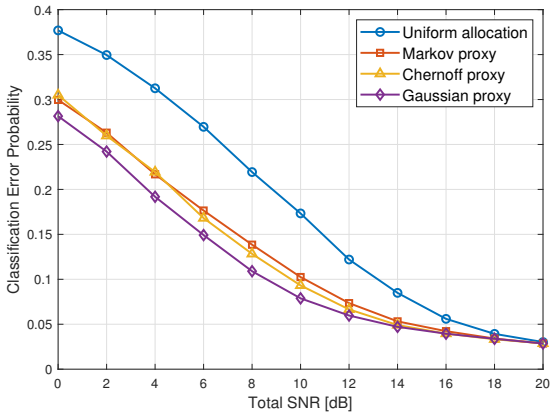
Corollary 14: If $\sigma_t = \sigma$ for all $t \in \{1, \ldots, T\}$, then the *optimized* proxy of (16) can be approximated as:

$$\sum_{t=1}^T \beta_t Q\left(\frac{r_t^*}{\sigma_t}\right) \approx \frac{T}{2}\exp\left(-\frac{\mathcal{C}}{2T\sigma^2}\right)\left(\prod_{t=1}^T \beta_t\right)^{\frac{1}{T}} \tag{17}$$

where $\left(\prod_{t=1}^T \beta_t\right)^{\frac{1}{T}}$ is the geometric mean of $\boldsymbol{\beta}$.



(a)



(b)

Fig. 3. Evaluation of optimized SNRs ($T = 20$): (a) Mismatch probability and (b) classification error probability.

Proof: The proof is given in [20]. ■

We observe that a smaller geometric mean of $\boldsymbol{\beta}$ implies a lower proxy value.

Remark 15: The geometric mean of $\boldsymbol{\beta}$ is maximized for the uniform $\boldsymbol{\alpha} = \left(\frac{1}{T}, \ldots, \frac{1}{T}\right)$. Thus the non-uniform coefficients of AdaBoost's classifiers contribute to lower classification error probability. This suggests an advantage of AdaBoost over bagging that assigns the same coefficients to all classifiers (i.e., $\boldsymbol{\alpha} = \left(\frac{1}{T}, \ldots, \frac{1}{T}\right)$). This is a noteworthy fact because AdaBoost is known to be less robust than bagging in the problem of noisy data labels [15], [16].

V. Numerical Results

We validate the tools and analytic results with the UCI breast cancer dataset [21]. We compare mismatch probabilities and classification error probabilities of uniform resource allocation and optimized resource allocations for Markov, Chernoff, and Gaussian proxies. The noise-free AdaBoost was trained by Algorithm 1 with decision stumps as base classifiers. Based on the training output $\boldsymbol{\alpha}$, we compute $\boldsymbol{\beta}$ as shown in Table I and solve the corresponding optimization problems by (16).

Fig. 3 evaluates the mismatch probabilities and the classification error probabilities of the test set for $T = 20$, respectively. We observe that nonuniform communication power allocations can lower the mismatch probability as well as the classification error probability. Among the three nonuniform power allocations (Markov proxy, Chernoff proxy, and Gaussian proxy), the power allocation based on Gaussian proxy achieves the best performance in this dataset. We emphasize that Fig. 3 plots the actual mismatch and classification error probabilities over the test set optimized with different proxies, and *not* the values of the proxies themselves. Note that the horizontal axis corresponds to the total SNR budget $\frac{\mathcal{C}}{\sigma^2}$. The SNR gain is $4.2\,\mathrm{dB}$ at $P_e = 0.1$. For higher SNR, the mismatch probabilities converge to zero.

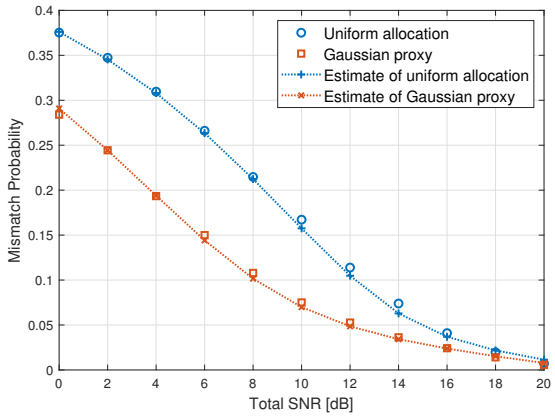


Fig. 4. Mismatch probabilities and their estimates by (14) for $T = 20$.

Fig. 4 shows the mismatch probabilities and their estimates for the uniform power allocation and the optimized power allocation using the Gaussian proxy, respectively. The estimates of mismatch probability are calculated by (14). We observe that the estimates match the mismatch probabilities well, which justifies using the Gaussian approximation in Section III-C. On the other hand, the other proxies (Markov and Chernoff) have the advantage that their values are proven upper bounds on the mismatch probability (while the Gaussian proxy in general is not a bound).

VI. Conclusion

In this paper, we propose a principled approach to optimize resource allocation for boosting classifiers. We defined three proxies and the corresponding importance metrics for base classifiers based on Markov inequality, Chernoff bound, and Gaussian approximation. By exploiting the positivity of the importance metrics, we formulated convex resource-allocation problems to minimize the impact of noise. We showed that the proposed approach can effectively improve the classification accuracy for the additive Gaussian noise model. Also, we found that the non-uniform coefficients in boosting offer an advantage over uniform ones (e.g., in bagging) for this noise model.

References

[1] T. G. Dietterich, "Ensemble methods in machine learning," in *Proc. Int. Workshop Multiple Classifier Syst.*, Dec. 2000, pp. 1–15.

[2] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *J. Comput. Syst. Sci.*, vol. 55, no. 1, pp. 119–139, Dec. 1997.

[3] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[4] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Jul. 1996, pp. 148–156.

[5] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognition (CVPR)*, Dec. 2001, pp. I–511–I–518.

[6] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Stat.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.

[7] L. Breiman, "Bagging predictors," *Mach. Learn.*, vol. 24, no. 2, pp. 123–140, Aug. 1996.

[8] N. R. Shanbhag, N. Verma, Y. Kim, A. D. Patil, and L. R. Varshney, "Shannon-inspired statistical computing for the nanoscale era," *Proc. IEEE*, vol. 107, no. 1, pp. 90–107, Jan. 2019.

[9] H. V. Poor and J. B. Thomas, "Applications of Ali-Silvey distance measures in the design of generalized quantizers for binary decision systems," *IEEE Trans. Commun.*, vol. 25, no. 9, pp. 893–900, Sep. 1977.

[10] C. Sakr, Y. Kim, and N. Shanbhag, "Analytical guarantees on numerical precision of deep neural networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Aug. 2017, pp. 3007–3016.

[11] C. Sakr, A. Patil, S. Zhang, Y. Kim, and N. Shanbhag, "Minimum precision requirements for the SVM-SGD learning algorithm," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Mar. 2017, pp. 1138–1142.

[12] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," *Automata Studies*, vol. 34, pp. 43–98, 1956.

[13] M. A. Donmez, M. Raginsky, A. C. Singer, and L. R. Varshney, "Cost-reliability tradeoffs in fusing unreliable computational units," *IEEE Open J. Signal Process.*, vol. 1, pp. 77–89, May 2020.

[14] D. Angluin and P. Laird, "Learning from noisy examples," *Mach. Learn.*, vol. 2, no. 4, pp. 343–370, Apr. 1988.

[15] B. Frenay and M. Verleysen, "Classification in the presence of label noise: A survey," *IEEE Trans. Neural Netw.*, vol. 25, no. 5, pp. 845–869, May 2014.

[16] T. G. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Mach. Learn.*, vol. 40, no. 2, pp. 139–157, Aug. 2000.

[17] C. Domingo and O. Watanabe, "MadaBoost: A modification of adaBoost," in *Proc. Annu. Conf. Comput. Learn. Theory (COLT)*, Jun.-Jul. 2000, pp. 180–189.

[18] P. M. Long and R. A. Servedio, "Random classification noise defeats all convex potential boosters," *Mach. Learn.*, vol. 78, no. 3, pp. 287–304, Mar. 2010.

[19] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of Machine Learning*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[20] Y. Kim, Y. Cassuto, and L. R. Varshney, "Boosting classifiers with noisy inference," *arXiv preprint arXiv:1909.04766*, 2019.

[21] A. Asuncion and D. Newman, "UCI machine learning repository," 2007. [Online]. Available: http://archive.ics.uci.edu/ml