Self-Bounding Majority Vote Learning Algorithms by the Direct Minimization of a Tight PAC-Bayesian C-Bound

Paul Viallard, Pascal Germain, Amaury Habrard, Emilie Morvant

Self-Bounding Majority Vote Learning Algorithms by the Direct Minimization of a Tight PAC-Bayesian C-Bound

Paul Viallard[1], Pascal Germain[2], Amaury Habrard[1], and Emilie Morvant[1]

[1] Univ Lyon, UJM-Saint-Etienne, CNRS, Institut d Optique Graduate School, Laboratoire Hubert Curien UMR 5516, F-42023, SAINT-ETIENNE, France
firstname.name@univ-st-etienne.fr
[2] Département d'informatique et de génie logiciel, Université Laval, Québec, Canada
pascal.germain@ift.ulaval.ca

Abstract. In the PAC-Bayesian literature, the C-Bound refers to an insightful relation between the risk of a majority vote classifier (under the zero-one loss) and the first two moments of its margin (*i.e.*, the expected margin and the voters' diversity). Until now, learning algorithms developed in this framework minimize the empirical version of the C-Bound, instead of explicit PAC-Bayesian generalization bounds. In this paper, by directly optimizing PAC-Bayesian guarantees on the C-Bound, we derive self-bounding majority vote learning algorithms. Moreover, our algorithms based on gradient descent are scalable and lead to accurate predictors paired with non-vacuous guarantees.

Keywords: Majority Vote · PAC-Bayesian · Self-Bounding Algorithm.

1 Introduction

In machine learning, ensemble methods [10] aim to combine hypotheses to make predictive models more robust and accurate. A weighted majority vote learning procedure is an ensemble method for classification where each voter/hypothesis is assigned a weight (*i.e.*, its influence in the final voting). Among the famous majority vote methods, we can cite Boosting [13], Bagging [5], or Random Forest [6]. Interestingly, most of the kernel-based classifiers, like Support Vector Machines [3, 7], can be seen as a majority vote of kernel functions. Understanding when and why weighted majority votes perform better than a single hypothesis is challenging. To study the generalization abilities of such majority votes, the PAC-Bayesian framework [34, 25] offers powerful tools to obtain Probably Approximately Correct (PAC) generalization bounds. Motivated by the fact that PAC-Bayesian analyses can lead to tight bounds (*e.g.*, [28]), developing algorithms to minimize such bounds is an important direction (*e.g.*, [14, 15, 11, 24]).

We focus on a class of PAC-Bayesian algorithms minimizing an upper bound on the majority vote's risk called the C-Bound[1] in the PAC-Bayesian literature [20]. This bound has the advantage of involving the majority vote's margin

[1] The C-Bound was introduced by Breiman in the context of Random Forest [6].

and its second statistical moment, *i.e.*, the diversity of the voters. Indeed, these elements are important when one learns a combination [10, 19]: A good majority vote is made up of voters that are "accurate enough" and "sufficiently diverse". Various algorithms have been proposed to minimize the C-Bound: MinCq [31], P-MinCq [2], CqBoost [32], or CB-Boost [1]. Despite being empirically efficient, and justified by theoretical analyses based on the C-Bound, all these methods minimize *only* the empirical C-Bound and not directly a PAC-Bayesian generalization bound on the C-Bound. This can lead to vacuous generalization bound values and thus to poor risk certificates.

In this paper, we cover three different PAC-Bayesian viewpoints on generalization bounds for the C-Bound [26, 33, 20]. Starting from these three views, we derive three algorithms to optimize generalization bounds on the C-Bound. By doing so, we achieve *self-bounding algorithms* [12]: the predictor returned by the learner comes with a statistically valid risk upper bound. Importantly, our algorithms rely on fast gradient descent procedures. As far as we know, this is the first work that proposes both efficient algorithms for C-Bound optimization and non-trivial risk bound values.

The paper is organized as follows. Section 2 introduces the setting. Section 3 recalls the PAC-Bayes bounds on which we build our results. Our self-bounding algorithms leading to non-vacuous PAC-Bayesian bounds are described in Section 4. We provide experiments in Section 5, and conclude in Section 6.

2 Majority Vote Learning

2.1 Notations and Setting

We stand in the context of learning a weighted majority vote for binary classification. Let $\mathcal{X} \subseteq \mathbb{R}^d$ be a d-dimensional input space, and $\mathcal{Y}=\{-1,+1\}$ be the label space. We assume an unknown data distribution $\mathcal{D}$ on $\mathcal{X}\times\mathcal{Y}$, we denote by $\mathcal{D}_\mathcal{X}$ the marginal distribution on $\mathcal{X}$. A learning algorithm is provided with a learning sample $\mathcal{S}=\{(\mathbf{x}_i, y_i)\}_{i=1}^m$ where each example $(\mathbf{x}_i, y_i)$ is drawn *i.i.d.* from $\mathcal{D}$, we denote by $\mathcal{S}\sim\mathcal{D}^m$ the random draw of such a sample. Given $\mathcal{H}$ a hypothesis set constituted by so-called *voters* $h : \mathcal{X}\to\mathcal{Y}$, and $\mathcal{S}$, the learner aims to find a weighted combination of the voters from $\mathcal{H}$; the weights are modeled by a distribution on $\mathcal{H}$. To learn such a combination in the PAC-Bayesian framework, we assume a *prior* distribution $\mathcal{P}$ on $\mathcal{H}$, and—after the observation of $\mathcal{S}$—we learn a *posterior* distribution $\mathcal{Q}$ on $\mathcal{H}$. More precisely, we aim to learn a well-performing classifier that is expressed as a *$\mathcal{Q}$-weighted majority vote* $\mathrm{MV}_\mathcal{Q}$ defined as

$$\forall \mathbf{x} \in \mathcal{X}, \quad \mathrm{MV}_\mathcal{Q}(\mathbf{x}) \triangleq \operatorname{sign}\left(\mathop{\mathbb{E}}_{h\sim\mathcal{Q}} h(\mathbf{x})\right) = \operatorname{sign}\left(\sum_{h\in\mathcal{H}} \mathcal{Q}(h) h(\mathbf{x})\right).$$

We thus want to learn $\mathrm{MV}_\mathcal{Q}$ that commits as few errors as possible on unseen data from $\mathcal{D}$, *i.e.*, that leads to a low true risk $r_\mathcal{D}^{\mathrm{MV}}(\mathcal{Q})$ under the 0-1-loss defined as

$$r_\mathcal{D}^{\mathrm{MV}}(\mathcal{Q}) \triangleq \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}} \mathbf{I}\Big[\mathrm{MV}_\mathcal{Q}(\mathbf{x}) \neq y\Big], \quad \text{where } \mathbf{I}[a] = \begin{cases} 1 \text{ if the assertion } a \text{ is true,} \\ 0 \text{ otherwise.} \end{cases}$$

2.2 Gibbs Risk, Joint Error and C-Bound

Since $\mathcal{D}$ is unknown, a common way to try to minimize $r_{\mathcal{D}}^{\mathrm{MV}}(\mathcal{Q})$ is the minimization of its empirical counterpart $r_{\mathcal{S}}^{\mathrm{MV}}(\mathcal{Q}) = \frac{1}{m}\sum_{i=1}^{m}\mathbf{I}\left[\mathrm{MV}_{\mathcal{Q}}(\mathbf{x}_i)\neq y_i\right]$ computed on the learning sample $\mathcal{S}$ through the Empirical Risk Minimization principle. However, learning the weights by the direct minimization of $r_{\mathcal{S}}^{\mathrm{MV}}(\mathcal{Q})$ does not necessarily lead to a low true risk. One solution consists then in looking for precise estimators or generalization bounds of the true risk $r_{\mathcal{D}}^{\mathrm{MV}}(\mathcal{Q})$ to minimize them. In the PAC-Bayesian theory, a well-known estimator of the true risk $r_{\mathcal{D}}^{\mathrm{MV}}(\mathcal{Q})$ is the **Gibbs risk** defined as the $\mathcal{Q}$-average risk of the voters as

$$r_{\mathcal{D}}(\mathcal{Q}) \;=\; \mathop{\mathbb{E}}_{h\sim\mathcal{Q}}\mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}}\mathbf{I}\left[h(\mathbf{x})\neq y\right].$$

Its empirical counterpart is defined as $r_{\mathcal{S}}(\mathcal{Q})=\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}_{h\sim\mathcal{Q}}\mathbf{I}\left[h(\mathbf{x}_i)\neq y_i\right]$. However, in ensemble methods where one wants to combine voters efficiently, the Gibbs risk appears to be an unfair estimator since it does not take into account the fact that a combination of voters has to compensate for the individual errors. This is highlighted by the decomposition of $r_{\mathcal{D}}(\mathcal{Q})$ in Equation (1) (due to Lacasse *et al.* [20]) into the expected **disagreement** and the expected **joint error**, respectively defined by

$$d_{\mathcal{D}}(\mathcal{Q}) = \mathop{\mathbb{E}}_{h_1\sim\mathcal{Q}}\mathop{\mathbb{E}}_{h_2\sim\mathcal{Q}}\mathop{\mathbb{E}}_{\mathbf{x}\sim\mathcal{D}_{\mathcal{X}}}\mathbf{I}\big[h_1(\mathbf{x})\neq h_2(\mathbf{x})\big],$$
$$\text{and}\quad e_{\mathcal{D}}(\mathcal{Q}) = \mathop{\mathbb{E}}_{h_1\sim\mathcal{Q}}\mathop{\mathbb{E}}_{h_2\sim\mathcal{Q}}\mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}}\mathbf{I}\big[h_1(\mathbf{x})\neq y\big]\mathbf{I}\big[h_2(\mathbf{x})\neq y\big].$$

Indeed, an increase of the voter's diversity, captured by the disagreement $d_{\mathcal{D}}(\mathcal{Q})$, have a negative impact on the Gibbs risk, as

$$r_{\mathcal{D}}(\mathcal{Q}) = e_{\mathcal{D}}(\mathcal{Q}) + \tfrac{1}{2}d_{\mathcal{D}}(\mathcal{Q}). \tag{1}$$

Despite this unfavorable behavior, many PAC-Bayesian results deal only with the Gibbs risks, thanks to a straightforward upper bound of the majority vote's risk which consists in upper-bounding it by twice the Gibbs risk [21], *i.e.*,

$$r_{\mathcal{D}}^{\mathrm{MV}}(\mathcal{Q}) \leq 2\,r_{\mathcal{D}}(\mathcal{Q}) \;=\; 2e_{\mathcal{D}}(\mathcal{Q}) + d_{\mathcal{D}}(\mathcal{Q}). \tag{2}$$

This bound is tight only when the Gibbs risk is low (*e.g.*, when voters with large weights perform well individually [14, 21]). Recently, Masegosa *et al.* [24] propose to deal directly with the joint error as

$$r_{\mathcal{D}}^{\mathrm{MV}}(\mathcal{Q}) \leq 4e_{\mathcal{D}}(\mathcal{Q}) \;=\; 2r_{\mathcal{D}}(\mathcal{Q}) + 2e_{\mathcal{D}}(\mathcal{Q}) - d_{\mathcal{D}}(\mathcal{Q}). \tag{3}$$

Equation (3) is tighter than Equation (2) if $e_{\mathcal{D}}(\mathcal{Q})\leq\frac{1}{2}d_{\mathcal{D}}(\mathcal{Q}) \Leftrightarrow r_{\mathcal{D}}(\mathcal{Q})\leq d_{\mathcal{D}}(\mathcal{Q})$; This captures the fact that the voters need to be sufficiently diverse and commit errors on different points. However, when the joint error $e_{\mathcal{D}}(\mathcal{Q})$ exceeds $\frac{1}{4}$, the bound exceeds 1 and is uninformative. Another bound—known as the C-Bound in the PAC-Bayes literature [20]—has been introduced to capture this trade-off between the Gibbs risk $r_{\mathcal{D}}(\mathcal{Q})$ and the disagreement $d_{\mathcal{D}}(\mathcal{Q})$, and is recalled in the following theorem.

Theorem 1 (C-Bound). *For any distribution $\mathcal{D}$ on $\mathcal{X}\times\mathcal{Y}$, for any voters set $\mathcal{H}$, for any distribution $\mathcal{Q}$ on $\mathcal{H}$, if $r_{\mathcal{D}}(\mathcal{Q})<\frac{1}{2} \iff 2e_{\mathcal{D}}(\mathcal{Q})+d_{\mathcal{D}}(\mathcal{Q})<1$, we have*

$$r_{\mathcal{D}}^{MV}(\mathcal{Q}) \;\leq 1-\frac{\left(1-2r_{\mathcal{D}}(\mathcal{Q})\right)^2}{1-2d_{\mathcal{D}}(\mathcal{Q})} \quad \triangleq \quad C_{\mathcal{D}}(\mathcal{Q}) = 1-\frac{\left(1-[2e_{\mathcal{D}}(\mathcal{Q})+d_{\mathcal{D}}(\mathcal{Q})]\right)^2}{1-2d_{\mathcal{D}}(\mathcal{Q})}.$$

*The **empirical C-Bound** is denoted by $C_{\mathcal{S}}(\mathcal{Q})$ where the empirical disagreement is defined by $d_{\mathcal{S}}(\mathcal{Q})=\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}_{h_1\sim\mathcal{Q}}\mathbb{E}_{h_2\sim\mathcal{Q}}\mathbf{I}[h_1(\mathbf{x}_i)\neq h_2(\mathbf{x}_i)]$, and the empirical joint error is defined by $e_{\mathcal{S}}(\mathcal{Q})=\frac{1}{m}\sum_{i=1}^{m}\mathbb{E}_{h_1\sim\mathcal{Q}}\mathbb{E}_{h_2\sim\mathcal{Q}}\mathbf{I}[h_1(\mathbf{x}_i)\neq y_i]\mathbf{I}[h_2(\mathbf{x}_i)\neq y_i]$.*

As Equation (3), the C-Bound is tighter than Equation (2) when $r_{\mathcal{D}}(\mathcal{Q}) \leq d_{\mathcal{D}}(\mathcal{Q})$ and looks for a good trade-off between individual risks and disagreement. The main interest of the C-bound compared to Equation (3) is that when $e_{\mathcal{D}}(\mathcal{Q})$ is close to $\frac{1}{4}$, the C-Bound can be close to 0 depending on the value of the disagreement $d_{\mathcal{D}}(\mathcal{Q})$: the C-bound is then more precise. Moreover, it is important to notice that the C-Bound is always tighter than Equation (3) and tighter than Equation (2) when $r_{\mathcal{D}}(\mathcal{Q}) \leq d_{\mathcal{D}}(\mathcal{Q})$. We summarize the relationships between Equations (2), (3) and $C_{\mathcal{D}}(\mathcal{Q})$ in the next theorem.

Theorem 2 (From Germain et al. [32] and Masegosa et al. [24]). *For any distribution $\mathcal{D}$ on $\mathcal{X}\times\mathcal{Y}$, for any voters set $\mathcal{H}$, for any distribution $\mathcal{Q}$ on $\mathcal{H}$, if $r_{\mathcal{D}}(\mathcal{Q})<\frac{1}{2}$, we have*

$$\begin{array}{ll} \textit{(i)}\ C_{\mathcal{D}}(\mathcal{Q}) \leq 4e_{\mathcal{D}}(\mathcal{Q}) \leq 2r_{\mathcal{D}}(\mathcal{Q}), & \textit{if } r_{\mathcal{D}}(\mathcal{Q}) \leq d_{\mathcal{D}}(\mathcal{Q}), \\ \textit{(ii)}\ 2r_{\mathcal{D}}(\mathcal{Q}) \leq C_{\mathcal{D}}(\mathcal{Q}) \leq 4e_{\mathcal{D}}(\mathcal{Q}), & \textit{otherwise.} \end{array}$$

In this paper, we focus on the minimization of PAC-Bayesian generalization bounds on the C-Bound to get a low-risk majority vote. In Section 3, we recall such PAC-Bayesian bounds that have been introduced in the literature.

2.3 Related Works

Previous algorithms have been developed to minimize the *empirical* C-Bound $C_{\mathcal{S}}(\mathcal{Q})$. Roy *et al.* [31] first proposed MinCq where this minimization is expressed as a quadratic problem. MinCq considers a specific voters' set to regularize the minimization process. One drawback of MinCq is that the optimization problem is not scalable to large datasets. Lately, Bauvin *et al.* [1] proposed CB-Boost that minimizes $C_{\mathcal{S}}(\mathcal{Q})$ in a greedy procedure with the advantage to be more scalable while obtaining sparser majority vote. However, since both MinCq and CB-Boost minimize the empirical $C_{\mathcal{S}}(\mathcal{Q})$, the PAC-Bayesian generalization bound associated with their learned majority vote predictors can be vacuous. Note that CB-Boost has been proposed to improve another algorithm called CqBoost [32]. When it comes to deriving a learning algorithm that directly minimizes a PAC-Bayesian bound, it is mentioned in the literature that optimizing a PAC-Bayesian

bound on the C-bound is not trivial [24, 22]. This underlines the need of other majority vote learning algorithms based on the C-Bound, which motivates our contributions of Section 4.

3 PAC-Bayesian C-Bounds

We recall now three PAC-Bayesian generalization bounds on the C-Bound referred hereafter as the **PAC-Bayesian C-Bounds**. Considering these three approaches has the interest to offer a large coverage of the PAC-Bayesian C-bound literature. Our contribution, described in Section 4, consists in deriving a self-bounding algorithm for each of these PAC-Bayesian C-Bounds. This shows that the PAC-Bayesian C-Bound offers various ways to learn majority votes that might have been overlooked until now.

3.1 An Intuitive Bound—McAllester's View

We recall the most intuitive and interpretable PAC-Bayesian C-Bound [32]. It consists in upper-bounding separately the Gibbs risk $r_{\mathcal{D}}(\mathcal{Q})$ and the disagreement $d_{\mathcal{D}}(\mathcal{Q})$ with the usual PAC-Bayesian bound of McAllester [26] that bounds the deviation between true and empirical values with the Euclidean distance.

Theorem 3 (PAC-Bayesian C-Bound of Roy et al. [32]). *For any distribution $\mathcal{D}$ on $\mathcal{X}\times\mathcal{Y}$, for any prior distribution $\mathcal{P}$ on $\mathcal{H}$, for any $\delta>0$, we have*

$$\Pr_{\mathcal{S}\sim\mathcal{D}^m}\left(\forall\mathcal{Q}\ \text{on}\ \mathcal{H},\ C_{\mathcal{D}}(\mathcal{Q}) \leq \underbrace{1-\frac{\left(1-2\min\left[\frac{1}{2}, r_{\mathcal{S}}(\mathcal{Q})+\sqrt{\frac{1}{2}\psi_r(\mathcal{Q})}\right]\right)^2}{1-2\max\left[0, d_{\mathcal{S}}(\mathcal{Q})-\sqrt{\frac{1}{2}\psi_d(\mathcal{Q})}\right]}}_{C^{\mathtt{M}}_{\mathcal{S}}(\mathcal{Q})}\right) \geq 1-2\delta, \tag{4}$$

with $\psi_r(\mathcal{Q}) = \frac{1}{m}\left[\mathrm{KL}(\mathcal{Q}\|\mathcal{P})+\ln\frac{2\sqrt{m}}{\delta}\right]$, *and* $\psi_d(\mathcal{Q}) = \frac{1}{m}\left[2\,\mathrm{KL}(\mathcal{Q}\|\mathcal{P})+\ln\frac{2\sqrt{m}}{\delta}\right]$, *and* $\mathrm{KL}(\mathcal{Q}\|\mathcal{P}) = \mathbb{E}_{h\sim\mathcal{Q}}\ln\frac{\mathcal{Q}(h)}{\mathcal{P}(h)}$ *is the KL-divergence between $\mathcal{Q}$ and $\mathcal{P}$.*

While there is no algorithm that directly minimizes Equation (4), this kind of interpretable bound can be seen as a justification of the optimization of $r_{\mathcal{S}}(\mathcal{Q})$ and $d_{\mathcal{S}}(\mathcal{Q})$ in the empirical C-Bound such as for MinCq [31] or CB-Boost [1]. In Section 4.1, we derive a first algorithm to directly minimize it.

However, this PAC-Bayesian C-Bound can have a severe disadvantage with a small m and a Gibbs risk close to $\frac{1}{2}$: even for a $\mathrm{KL}(\mathcal{Q}\|\mathcal{P})$ close to 0 and a low empirical C-Bound, the value of the PAC-Bayesian C-Bound will be close to 1. To overcome this drawback, one solution is to follow another PAC-Bayesian point of view, the one proposed by Seeger [33] that compares the true and empirical values through $\mathrm{kl}(a\|b)=a\log\left[\frac{a}{b}\right]+(1-a)\log\left[\frac{1-a}{1-b}\right]$, knowing that $|a-b| \leq \sqrt{\frac{1}{2}\mathrm{kl}(a\|b)}$ (Pinsker's inequality).

In the next two subsections, we recall such bounds. The first one in Theorem 4 involves the risk and the disagreement, while the second one in Theorem 5 simultaneously bounds the joint error and the disagreement.

3.2 A Tighter Bound—Seeger's view

The PAC-Bayesian generalization bounds based on the Seeger's approach [33] are known to produce tighter bounds [15]. As for Theorem 3, the result below bounds independently the Gibbs risk $r_{\mathcal{D}}(\mathcal{Q})$ and the disagreement $d_{\mathcal{D}}(\mathcal{Q})$.

Theorem 4 (**PAC-Bayesian C-Bound (PAC-Bound 1) of Germain et al. [15]**). *Under the same assumptions and notations as Theorem 3, we have*

$$\Pr_{\mathcal{S}\sim\mathcal{D}^m}\left(\forall\mathcal{Q}\ \textit{on}\ \mathcal{H},\ C_{\mathcal{D}}(\mathcal{Q}) \le \underbrace{1-\frac{\left(1-2\min\left[\frac{1}{2},\overline{\mathrm{kl}}\left(r_{\mathcal{S}}(\mathcal{Q})\mid\psi_r(\mathcal{Q})\right)\right]\right)^2}{1-2\max\left[0,\underline{\mathrm{kl}}\left(d_{\mathcal{S}}(\mathcal{Q})\mid\psi_d(\mathcal{Q})\right)\right]}}_{C^{\mathrm{S}}_{\mathcal{S}}(\mathcal{Q})}\right)\ge 1-2\delta, \tag{5}$$

with $\overline{\mathrm{kl}}(q|\psi)=\max\{p\in(0,1)|\mathrm{kl}(q\|p)\le\psi\}$, *and* $\underline{\mathrm{kl}}(q|\psi)=\min\{p\in(0,1)|\mathrm{kl}(q\|p)\le\psi\}$.

The form of this bound makes the optimization a challenging task: the functions $\overline{\mathrm{kl}}$ and $\underline{\mathrm{kl}}$ do not benefit from closed-form solutions. However, we see in Section 3.2 that the optimization of $\overline{\mathrm{kl}}$ and $\underline{\mathrm{kl}}$ can be done by the bisection method [30], leading to an easy-to-solve algorithm to optimize this PAC-Bayesian C-Bound.

3.3 Another Tighter Bound–Lacasse's View

The last theorem on which we build our contributions is described below. Proposed initially by Lacasse *et al.* [20], its interest is that it simultaneously bounds the joint error and the disagreement (as explained by Germain *et al.* [15]). Here, to compute the bound, we need to find the worst C-Bound value that can be obtained with a couple of joint error and disagreement denoted by (e,d) belonging to the set $A_{\mathcal{S}}(\mathcal{Q})$ that is defined by

$$A_{\mathcal{S}}(\mathcal{Q}) = \left\{(e,d)\ \middle|\ \mathrm{kl}\left(e_{\mathcal{S}}(\mathcal{Q}),d_{\mathcal{S}}(\mathcal{Q})\|e,d\right)\le\kappa(\mathcal{Q})\right\},$$

where $\kappa(\mathcal{Q}) = \frac{1}{m}\left[2\mathrm{KL}(\mathcal{Q}\|\mathcal{P})+\ln\frac{2\sqrt{m}+m}{\delta}\right]$,
and $\mathrm{kl}(q_1,q_2\|p_1,p_2) = q_1\ln\frac{q_1}{p_1}+q_2\ln\frac{q_2}{p_2}+(1-q_1-q_2)\ln\frac{1-q_1-q_2}{1-p_1-p_2}$.
The set $A_{\mathcal{S}}(\mathcal{Q})$ can actually contain some pairs not achievable by any $\mathcal{D}$, it can then be restricted to the valid subset $\widetilde{A}_{\mathcal{S}}(\mathcal{Q})$ defined in the theorem below.

Theorem 5 (**PAC-Bayesian C-Bound (PAC-Bound 2) of Germain et al. [15]**). *Under the same assumptions as Theorem 3, we have*

$$\Pr_{\mathcal{S}\sim\mathcal{D}^m}\left(\forall\mathcal{Q}\ \textit{on}\ \mathcal{H},\ C_{\mathcal{D}}(\mathcal{Q})\le\sup_{(e,d)\in\widetilde{A}_{\mathcal{S}}(\mathcal{Q})}\left[1-\frac{\left(1-(2e+d)\right)^2}{1-2d}\right]\right)\ge 1-\delta,$$
$$\textit{where}\ \widetilde{A}_{\mathcal{S}}(\mathcal{Q}) = \left\{(e,d)\in A_{\mathcal{S}}(\mathcal{Q})\ \middle|\ d\le 2\sqrt{e}-2e\,,\ 2e+d<1\right\}.$$

Optimizing this bound *w.r.t.* $\mathcal{Q}$ can be challenging, since it boils down to optimize indirectly the set $\widetilde{A}_{\mathcal{S}}(\mathcal{Q})$. Hence, a direct optimization by gradient descent is not possible. In Section 4.3 we derive an approximation easier to optimize.

Algorithm 1 Minimization of Equation (4) by GD

Given: learning sample $\mathcal{S}$, prior distribution $\mathcal{P}$ on $\mathcal{H}$, the objective function $G^{\mathtt{M}}_{\mathcal{S}}(\mathcal{Q})$
Update function[2] UPDATE-$\mathcal{Q}$
Hyperparameters: number of iterations T
function MINIMIZE-$\mathcal{Q}$
$\mathcal{Q} \leftarrow \mathcal{P}$
for $t \leftarrow 1$ to T **do** $\mathcal{Q} \leftarrow$UPDATE-$\mathcal{Q}(G^{\mathtt{M}}_{\mathcal{S}}(\mathcal{Q}))$
return $\mathcal{Q}$

4 Self-Bounding Algorithms for PAC-Bayesian C-Bounds

In this section, we present our contribution that consists in proposing three self-bounding algorithms to directly minimize the PAC-Bayesian C-Bounds.

4.1 Algorithm Based on McAllester's View

We derive in Algorithm 1 a method to directly minimize the PAC-Bayesian C-Bound of Theorem 3 by Gradient Descent (GD). An important aspect of the optimization is that if $r_{\mathcal{S}}(\mathcal{Q})+\sqrt{\frac{1}{2}\psi_r(\mathcal{Q})} \geq \frac{1}{2}$, the gradient of the numerator in $C^{\mathtt{M}}_{\mathcal{S}}(\mathcal{Q})$ with respect to $\mathcal{Q}$ is 0 which makes the optimization impossible. Hence, we aim at minimizing the following constraint optimization problem:

$$\min_{\mathcal{Q}} \underbrace{\left[1 - \frac{\left(1 - 2\min\left[\frac{1}{2}, r_{\mathcal{S}}(\mathcal{Q})+\sqrt{\frac{1}{2}\psi_r(\mathcal{Q})}\right]\right)^2}{1 - 2\max\left[0, d_{\mathcal{S}}(\mathcal{Q})-\sqrt{\frac{1}{2}\psi_d(\mathcal{Q})}\right]}\right]}_{C^{\mathtt{M}}_{\mathcal{S}}(\mathcal{Q})} \quad \text{s.t} \quad r_{\mathcal{S}}(\mathcal{Q})+\sqrt{\tfrac{1}{2}\psi_r(\mathcal{Q})} \leq \tfrac{1}{2}.$$

From this formulation, we deduce a non-constrained optimization problem: $\min_{\mathcal{Q}} \left[C^{\mathtt{M}}_{\mathcal{S}}(\mathcal{Q}) + \mathbf{B}(r_{\mathcal{S}}(\mathcal{Q})+\sqrt{\frac{1}{2}\psi_r(\mathcal{Q})}-\frac{1}{2})\right]$, where $\mathbf{B}$ is the barrier function defined as $\mathbf{B}(a)=0$ if $a \leq 0$ and $\mathbf{B}(a)=+\infty$ otherwise. Due to the nature of $\mathbf{B}$, this problem is not suitable for optimization: the objective function will be infinite when $a>0$. To tackle this drawback, we replace $\mathbf{B}$ by the approximation introduced by Kervadec *et al.* [17] called the log-barrier extension and defined as

$$\mathbf{B}_{\lambda}(a) = \begin{cases} -\frac{1}{\lambda}\ln(-a), & \text{if } a \leq -\frac{1}{\lambda^2}, \\ \lambda a - \frac{1}{\lambda}\ln(\frac{1}{\lambda^2}) + \frac{1}{\lambda}, & \text{otherwise.} \end{cases}$$

In fact, $\mathbf{B}_{\lambda}$ tends to $\mathbf{B}$ when λ tends to $+\infty$. Compared to the standard log-barrier[3], the function $\mathbf{B}_{\lambda}$ is differentiable even when the constraint is not satisfied, *i.e.*, when $a > 0$. By taking into account the constraint $r_{\mathcal{S}}(\mathcal{Q})+\sqrt{\frac{1}{2}\psi_r(\mathcal{Q})} \leq \frac{1}{2}$,

[2] UPDATE-$\mathcal{Q}$ is a generic update function, *i.e.*, it can be for example a standard update of GD or the update of another algorithm like Adam [18] or COCOB [27].
[3] The reader can refer to [4] for an introduction of interior-point methods.

we solve by GD with Algorithm 1 the following problem:

$$\min_{\mathcal{Q}} \; G^{\mathtt{M}}_{\mathcal{S}}(\mathcal{Q}) \;=\; \min_{\mathcal{Q}} \; C^{\mathtt{M}}_{\mathcal{S}}(\mathcal{Q}) + \mathbf{B}_\lambda\left(r_{\mathcal{S}}(\mathcal{Q}) + \sqrt{\tfrac{1}{2}\psi_r(\mathcal{Q})} - \tfrac{1}{2}\right).$$

For a given λ, the optimizer will thus find a solution with a good trade-off between minimizing $C^{\mathtt{M}}_{\mathcal{S}}(\mathcal{Q})$ and the log-barrier extension function $\mathbf{B}_\lambda$. As we show in the experiments, minimizing the McAllester-based bound does not lead to the tightest bound. Indeed, as mentioned in Section 3, such bound is looser than Seeger-based bounds, and leads to a looser PAC-Bayesian C-Bound.

4.2 Algorithm Based on Seeger's View

In order to obtain better generalization guarantees, we should optimize the Seeger-based C-bound of Theorem 4. In the same way as in the previous section, we seek at minimizing the following optimization problem:

$$\min_{\mathcal{Q}} \underbrace{\left[1 - \frac{\left(1 - 2\min\left[\tfrac{1}{2}, \overline{\mathrm{kl}}\left(r_{\mathcal{S}}(\mathcal{Q}) \mid \psi_r(\mathcal{Q})\right)\right]\right)^2}{1 - 2\max\left[0, \underline{\mathrm{kl}}\left(d_{\mathcal{S}}(\mathcal{Q}) \mid \psi_d(\mathcal{Q})\right)\right]}\right]}_{C^{\mathtt{S}}_{\mathcal{S}}(\mathcal{Q})} \quad \text{s.t} \quad \overline{\mathrm{kl}}\left(r_{\mathcal{S}}(\mathcal{Q}) \mid \psi_r(\mathcal{Q})\right) \leq \tfrac{1}{2},$$

with $\overline{\mathrm{kl}}(q|\psi) = \max\{p \in (0,1) | \mathrm{kl}(q\|p) \leq \psi\}$, and $\underline{\mathrm{kl}}(q|\psi) = \min\{p \in (0,1) | \mathrm{kl}(q\|p) \leq \psi\}$. For the same reasons as for deriving Algorithm 1, we propose to solve by GD:

$$\min_{\mathcal{Q}} \; G^{\mathtt{S}}_{\mathcal{S}}(\mathcal{Q}) \;=\; \min_{\mathcal{Q}} \; C^{\mathtt{S}}_{\mathcal{S}}(\mathcal{Q}) + \mathbf{B}_\lambda\big(\overline{\mathrm{kl}}\left(r_{\mathcal{S}}(\mathcal{Q}) \mid \psi_r(\mathcal{Q})\right) - \tfrac{1}{2}\big).$$

The main challenge to optimize it is to evaluate $\overline{\mathrm{kl}}$ or $\underline{\mathrm{kl}}$ and to compute their derivatives. To do so, we follow the bisection method to calculate $\overline{\mathrm{kl}}$ and $\underline{\mathrm{kl}}$ proposed by Reeb *et al.* [30]. This method is summarized in the functions COMPUTE-$\overline{\mathrm{kl}}(q|\psi)$ and COMPUTE-$\underline{\mathrm{kl}}(q|\psi)$ of Algorithm 2, and consists in refining iteratively an interval $[p_{\min}, p_{\max}]$ with $p \in [p_{\min}, p_{\max}]$ such that $\mathrm{kl}(q\|p) = \psi$. For the sake of completeness, we provide the derivatives of $\underline{\mathrm{kl}}$ and $\overline{\mathrm{kl}}$ with respect to q and ψ, that are:

$$\frac{\partial \mathrm{k}(q|\psi)}{\partial q} = \frac{\ln \frac{1-q}{1-\mathrm{k}(q|\psi)} - \ln \frac{q}{\mathrm{k}(q|\psi)}}{\frac{1-q}{1-\mathrm{k}(q|\psi)} - \frac{q}{\mathrm{k}(q|\psi)}}, \text{ and } \frac{\partial \mathrm{k}(q|\psi)}{\partial \psi} = \frac{1}{\frac{1-q}{1-\mathrm{k}(q|\psi)} - \frac{q}{\mathrm{k}(q|\psi)}}, \tag{6}$$

with k is either $\underline{\mathrm{kl}}$ or $\overline{\mathrm{kl}}$. To compute the derivatives with respect to the posterior $\mathcal{Q}$, we use the chain rule for differentiation with a deep learning framework (such as PyTorch [29]). The global algorithm is summarized in Algorithm 2.

4.3 Algorithm Based on Lacasse's View

Theorem 5 jointly upper-bounds the joint error $e_{\mathcal{D}}(\mathcal{Q})$ and the disagreement $d_{\mathcal{D}}(\mathcal{Q})$; But as pointed out in Section 3.3 its optimization can be hard. To ease its manipulation, we derive below a C-Bound resulting of a reformulation of the constraints involved in the set $\widetilde{A}_{\mathcal{S}}(\mathcal{Q}) = \{(e,d) \in A_{\mathcal{S}}(\mathcal{Q}) \mid d \leq 2\sqrt{e} - 2e, 2e + d < 1\}$.

Algorithm 2 Minimization of Equation (4) by GD

Given: learning sample $\mathcal{S}$, prior distribution $\mathcal{P}$ on $\mathcal{H}$, the objective function $G^{\mathtt{M}}_{\mathcal{S}}(\mathcal{Q})$
Update function UPDATE-$\mathcal{Q}$
Hyperparameters: number of iterations T
function MINIMIZE-$\mathcal{Q}$
  $\mathcal{Q} \leftarrow \mathcal{P}$
  for $t \leftarrow 1$ to T **do**
    Compute $G^{\mathtt{S}}_{\mathcal{S}}(\mathcal{Q})$ using COMPUTE-$\overline{\mathrm{kl}}(q|\psi)$ and COMPUTE-$\underline{\mathrm{kl}}(q|\psi)$
    $\mathcal{Q} \leftarrow$UPDATE-$\mathcal{Q}(G^{\mathtt{S}}_{\mathcal{S}}(\mathcal{Q}))$ (thanks to the derivatives in Equation (6))
  return $\mathcal{Q}$

Hyperparameters: tolerance ϵ, maximal number of iterations $T_{\max}$
function COMPUTE-$\overline{\mathrm{kl}}(q|\psi)$ (RESP. COMPUTE-$\underline{\mathrm{kl}}(q|\psi)$)
  $p_{\max} \leftarrow 1$ and $p_{\min} \leftarrow q$ (resp. $p_{\max} \leftarrow q$ and $p_{\min} \leftarrow 0$)
  for $t \leftarrow 1$ to $T_{\max}$ **do**
    $p = \frac{1}{2}\left[p_{\min} + p_{\max}\right]$
    if $\mathrm{kl}(q\|p) = \psi$ or $(p_{\min} - p_{\max}) < \epsilon$ **then return** p
    if $\mathrm{kl}(q\|p) > \psi$ **then** $p_{\max} = p$ (resp. $p_{\min} = p$)
    if $\mathrm{kl}(q\|p) < \psi$ **then** $p_{\min} = p$ (resp. $p_{\max} = p$)
  return p

Theorem 6. *Under the same assumptions as Theorem 3, we have*

$$\Pr_{\mathcal{S}\sim\mathcal{D}^m}\left(C_{\mathcal{D}}(\mathcal{Q}) \leq \sup_{(e,d)\in\widehat{A}_{\mathcal{S}}(\mathcal{Q})} \underbrace{\left[1 - \frac{\left[1-(2e+d)\right]^2}{1-2d}\right]}_{C^{\mathrm{L}}(e,d)}\right) \geq 1-\delta, \qquad (7)$$

$$\text{where } \widehat{A}_{\mathcal{S}}(\mathcal{Q}) = \left\{(e,d)\in A_{\mathcal{S}}(\mathcal{Q}) \,\middle|\, d \leq 2\sqrt{\min\left(e,\tfrac{1}{4}\right)} - 2e,\ d < \tfrac{1}{2}\right\},$$

and $A_{\mathcal{S}}(\mathcal{Q}) = \left\{(e,d) \middle| \mathrm{kl}(e_{\mathcal{S}}(\mathcal{Q}), d_{\mathcal{S}}(\mathcal{Q}) \| e, d) \leq \kappa(\mathcal{Q})\right\}$, *with* $\kappa(\mathcal{Q}) = \frac{2\mathrm{KL}(\mathcal{Q}\|\mathcal{P}) + \ln\frac{2\sqrt{m}+m}{\delta}}{m}$.

Proof. Beforehand, we explain how we fixed the constraints involved in $\widehat{A}_{\mathcal{S}}(\mathcal{Q})$. We add to $A_{\mathcal{S}}(\mathcal{Q})$ three constraints: $d \leq 2\sqrt{e} - 2e$ (from Prop. 9 of [15]), $d \leq 1-2e$, and $d < \frac{1}{2}$. We remark that when $e \leq \frac{1}{4}$, we have $2\sqrt{e} - 2e \leq 1 - 2e$. Then, we merge $d \leq 2\sqrt{e} - 2e$ and $d \leq 1-2e$ into $d \leq 2\sqrt{\min\left(e, \frac{1}{4}\right)} - 2e$. Indeed, we have

$$d \leq 2\sqrt{\min(e, \tfrac{1}{4})} - 2e \iff \begin{cases} d \leq 2\sqrt{e} - 2e & \text{if } e \leq \frac{1}{4}, \\ d < 1 - 2e & \text{if } e \geq \frac{1}{4}. \end{cases}$$

We prove now that under the constraints involved in $\widehat{A}_{\mathcal{S}}(\mathcal{Q})$, we still have a valid bound on $C_{\mathcal{D}}(\mathcal{Q})$. To do so, we consider two cases.

Case 1: If for all $(e,d) \in \widehat{A}_{\mathcal{S}}(\mathcal{Q})$ we have $2e+d<1$.
In this case $(e_{\mathcal{D}}(\mathcal{Q}), d_{\mathcal{D}}(\mathcal{Q})) \in \widehat{A}_{\mathcal{S}}(\mathcal{Q})$, then we have $2e_{\mathcal{D}}(\mathcal{Q})+d_{\mathcal{D}}(\mathcal{Q})<1$ and Theorem 1 holds. We have $C_{\mathcal{D}}(\mathcal{Q}) = 1-\frac{[1-(2e_{\mathcal{D}}(\mathcal{Q})+d_{\mathcal{D}}(\mathcal{Q}))]^2}{1-2d_{\mathcal{D}}(\mathcal{Q})} \leq \sup_{(e,d)\in\widehat{A}_{\mathcal{S}}(\mathcal{Q})} C^{\mathtt{L}}(e,d)$.

Case 2: If there exists $(e,d) \in \widehat{A}_{\mathcal{S}}(\mathcal{Q})$ such that $2e+d=1$.
We have $\sup_{(e,d)\in\widehat{A}_{\mathcal{S}}(\mathcal{Q})} C^{\mathtt{L}}(e,d) = 1$ that is a valid bound on $C_{\mathcal{D}}(\mathcal{Q})$. □

Theorem 6 suggests then the following constrained optimization problem:

$$\min_{\mathcal{Q}} \left\{ \sup_{(e,d)\in\left[0,\frac{1}{2}\right]^2} \left(1-\frac{\left[1-(2e+d)\right]^2}{1-2d}\right) \text{ s.t. } (e,d)\in\widehat{A}_{\mathcal{S}}(\mathcal{Q}) \right\} \text{ s.t. } 2e_{\mathcal{S}}(\mathcal{Q})+d_{\mathcal{S}}(\mathcal{Q})\leq 1,$$

with $\widehat{A}_{\mathcal{S}}(\mathcal{Q})=\left\{(e,d)\middle| d \leq 2\sqrt{\min\left(e,\frac{1}{4}\right)}-2e,\ d<\frac{1}{2},\ \mathrm{kl}(e_{\mathcal{S}}(\mathcal{Q}),d_{\mathcal{S}}(\mathcal{Q})\|e,d)\leq\kappa(\mathcal{Q})\right\}$.
Actually, we can rewrite this constrained optimization problem into an unconstrained one using the barrier function. We obtain

$$\min_{\mathcal{Q}} \left\{ \max_{(e,d)\in\left[0,\frac{1}{2}\right]^2} \left(C^{\mathtt{L}}(e,d) - \mathbf{B}\left[d-2\sqrt{\min\left(e,\frac{1}{4}\right)}-2e\right] - \mathbf{B}\left[d-\frac{1}{2}\right] \right.\right.$$
$$\left.\left. - \mathbf{B}\left[\mathrm{kl}\left(e_{\mathcal{S}}(\mathcal{Q}),d_{\mathcal{S}}(\mathcal{Q})\|e,d\right)-\kappa(\mathcal{Q})\right]\right) + \mathbf{B}\left[2e_{\mathcal{S}}(\mathcal{Q})+d_{\mathcal{S}}(\mathcal{Q})-1\right] \right\}, \quad (8)$$

where $C^{\mathtt{L}}(e,d) = 1-\frac{(1-(2e+d))^2}{1-2d}$ if $d<\frac{1}{2}$, and $C^{\mathtt{L}}(e,d)=1$ otherwise. However, this problem cannot be optimized directly by GD. In this case, we have a min-max optimization problem, *i.e.*, for each descent step we need to find the couple (e,d) that maximizes the $C^{\mathtt{L}}(e,d)$ given the three constraints that define $\widehat{A}_{\mathcal{S}}(\mathcal{Q})$ before updating the posterior distribution $\mathcal{Q}$.

First, to derive our optimization procedure, we focus on the inner maximization problem when $e_{\mathcal{S}}(\mathcal{Q})$ and $d_{\mathcal{S}}(\mathcal{Q})$ are fixed in order to find the optimal (e,d). However, the function $C^{\mathtt{L}}(e,d)$ we aim at maximizing is not concave for all $(e,d)\in\mathbb{R}^2$, implying that the implementation of its maximization can be hard[4]. Fortunately, $C^{\mathtt{L}}(e,d)$ is quasi-concave [15] for $(e,d)\in[0,1]\times[0,\frac{1}{2}]$. Then by definition of quasi-concavity, we have:

$$\begin{aligned} & \forall\alpha\in[0,1], \quad \left\{(e,d)\middle| 1-\frac{\left[1-(2e+d)\right]^2}{1-2d}\geq 1-\alpha\right\} \\ \iff & \forall\alpha\in[0,1], \quad \left\{(e,d)\ \middle|\ \alpha(1-2d)-\left[1-(2e+d)\right]^2\geq 0\right\}. \end{aligned}$$

Hence, for any fixed $\alpha\in[0,1]$ we can look for (e,d) that maximizes $C^{\mathtt{L}}(e,d)$ and respects the constraints involved in $\widehat{A}_{\mathcal{S}}(\mathcal{Q})$. This is equivalent to solve the

[4] For example, when using CVXPY [9], that uses Disciplined Convex Programming (DCP [16]), the maximization of a non-concave function is not possible.

Algorithm 3 Minimization of Equation (7) by GD

Given: learning sample $\mathcal{S}$, prior $\mathcal{P}$ on $\mathcal{H}$, the objective function $G_{\mathcal{S}}^{e^*,d^*}(\mathcal{Q})$
Update function UPDATE-$\mathcal{Q}$
Hyperparameters: number of iterations T
function MINIMIZE-$\mathcal{Q}$
  $\mathcal{Q} \leftarrow \mathcal{P}$
  for $t \leftarrow 1$ to T **do**
    $(e^*, d^*) \leftarrow$MAXIMIZE-e-$d(e_{\mathcal{S}}(\mathcal{Q}), d_{\mathcal{S}}(\mathcal{Q}))$
    $\mathcal{Q} \leftarrow$ UPDATE-$\mathcal{Q}(G_{\mathcal{S}}^{e^*,d^*}(\mathcal{Q}))$
  return $\mathcal{Q}$

Given: learning sample $\mathcal{S}$, joint error $e_{\mathcal{S}}(\mathcal{Q})$, disagreement $d_{\mathcal{S}}(\mathcal{Q})$
Hyperparameters: tolerance ϵ
function MAXIMIZE-e-$d(e_{\mathcal{S}}(\mathcal{Q}), d_{\mathcal{S}}(\mathcal{Q}))$
  $\alpha_{\min} = 0$ and $\alpha_{\max} = 1$
  while $\alpha_{\max} - \alpha_{\min} > \epsilon$ **do**
    $\alpha = \frac{1}{2}(\alpha_{\min} + \alpha_{\max})$
    $(e, d) \leftarrow$ Solve Equation (9)
    if $C^{\mathrm{L}}(e, d) \geq 1-\alpha$ **then** $\alpha_{\max} \leftarrow \alpha$ **else** $\alpha_{\min} \leftarrow \alpha$
  return (e, d)

following problem for a given $\alpha \in [0, 1]$:

$$\max_{(e,d)\in[0,\frac{1}{2}]^2} \quad \alpha(1-2d) - \Big[1-(2e+d)\Big]^2 \tag{9}$$
$$\text{s.t.} \quad d \leq 2\sqrt{\min\left(e, \tfrac{1}{4}\right)} - 2e \quad \text{and} \quad \mathrm{kl}\left(e_{\mathcal{S}}(\mathcal{Q}), d_{\mathcal{S}}(\mathcal{Q}) \| e, d\right) \leq \kappa(\mathcal{Q}).$$

In fact, we aim at finding $\alpha \in [0, 1]$ such that the maximization of Equation (9) leads to $1-\alpha$ equal to the largest value of $C^{\mathrm{L}}(e, d)$ under the constraints. To do so, we make use of the "Bisection method for quasi-convex optimization" [4] that is summarized in MAXIMIZE-e-d in Algorithm 3. We denote by (e^*, d^*) the solution of Equation (9). It remains then to solve the outer minimization problem that becomes:

$$\min_{\mathcal{Q}} \Big\{ \mathbf{B}\left[2e_{\mathcal{S}}(\mathcal{Q}) + d_{\mathcal{S}}(\mathcal{Q}) - 1\right] - \mathbf{B}\left[\mathrm{kl}\left(e_{\mathcal{S}}(\mathcal{Q}), d_{\mathcal{S}}(\mathcal{Q}) \| e^*, d^*\right) - \kappa(\mathcal{Q})\right] \Big\}.$$

Since the barrier function $\mathbf{B}$ is not suitable for optimization, we approximate this problem by replacing $\mathbf{B}$ by the log-barrier extension $\mathbf{B}_\lambda$, *i.e.*, we have

$$\min_{\mathcal{Q}} G_{\mathcal{S}}^{e^*,d^*}(\mathcal{Q}) = \min_{\mathcal{Q}} \Big\{ \mathbf{B}_\lambda\left[2e_{\mathcal{S}}(\mathcal{Q}) + d_{\mathcal{S}}(\mathcal{Q}) - 1\right] - \mathbf{B}_\lambda\left[\mathrm{kl}\left(e_{\mathcal{S}}(\mathcal{Q}), d_{\mathcal{S}}(\mathcal{Q}) \| e^*, d^*\right) - \kappa(\mathcal{Q})\right] \Big\}.$$

The global method is summarized in Algorithm 3. As a side note, we mention that the classic Danskin Theorem [8] used in min-max optimization theory is

Table 1. Comparison of the true risks "$r_{\mathcal{T}}^{\mathrm{MV}}$" and bound values "Bnd" obtained for each algorithm. "Bnd" is the value of the bound that is optimized, excepted for MinCq and CB-Boost for which we report the bound obtained with Theorem 6 instantiated with the majority vote learned. Results in **bold** are the couple ($r_{\mathcal{T}}^{\mathrm{MV}}$,Bnd) associated to **the lowest risk** value. *Italic* and underlined results are the couple ($r_{\mathcal{T}}^{\mathrm{MV}}$,Bnd) associated respectively to *the lowest bound* value and the second lowest bound values.

	Alg.1		Alg.2		Alg.3		CB-Boost		MinCq		Masegosa		2R	
	$r_{\mathcal{T}}^{\mathrm{MV}}$	Bnd	$r_{\mathcal{T}}^{\mathrm{MV}}$	Bnd	$r_{\mathcal{T}}^{\mathrm{MV}}$	Bnd	$r_{\mathcal{T}}^{\mathrm{MV}}$	Bnd	$r_{\mathcal{T}}^{\mathrm{MV}}$	Bnd	$r_{\mathcal{T}}^{\mathrm{MV}}$	Bnd	$r_{\mathcal{T}}^{\mathrm{MV}}$	Bnd
letter:AvsB	.009	.323	.018	.114	**.000**	**.085**	.000	.104	.009	.451	.004	.070	*.018*	*.056*
letter:DvsO	**.013**	**.469**	.018	.298	.018	.205	.022	.224	.022	.999	.018	.185	*.044*	*.174*
letter:OvsQ	.017	.489	.017	.332	**.009**	**.229**	.017	.249	.039	1	.013	.210	*.030*	*.201*
credit	.141	.912	.141	.874	.129	.816	.144	.855	**.126**	**.929**	.132	.869	*.150*	*.651*
glass	.047	.904	.047	.832	.056	.798	**.037**	**.911**	.056	.999	.056	.903	*.047*	*.566*
heart	.250	.976	.264	.962	.250	.955	.270	.981	.270	1	**.243**	**1.19**	*.250*	*.787*
tictactoe	.063	.815	.084	.750	**.056**	**.610**	.063	.649	.071	.782	.058	.580	*.152*	*.511*
usvotes	.041	.741	.046	.584	.037	.508	.037	.590	.046	.985	**.032**	**.490**	*.060*	*.342*
wdbc	.060	.725	.053	.603	.032	.523	**.025**	**.591**	.039	.992	.035	.513	*.063*	*.362*
mnist:1vs7	.006	.161	.005	.061	**.005**	**.038**	.005	.040	.015	.994	*.006*	*.034*	.006	.043
mnist:4vs9	.017	.238	.016	.167	.016	.110	.016	.113	.046	.960	***.016***	***.106***	.063	.148
mnist:5vs6	.011	.210	.011	.124	*.011*	*.078*	.011	.081	.035	.999	**.011**	**.073**	.036	.109
fash:COvsSH	**.108**	**.462**	.109	.433	.110	.366	.110	.371	.185	.894	*.111*	*.358*	.146	.409
fash:SAvsBO	.018	.217	.018	.134	.019	.094	.019	.097	.034	1	***.018***	***.087***	.020	.114
fash:TOvsPU	.029	.245	.029	.165	**.029**	**.133**	.030	.136	.045	.809	.030	.125	*.051*	*.123*
adult	.163	.532	.163	.514	**.163**	**.492**	.163	.495	.204	1	**.163**	**.492**	*.200*	*.413*
Mean	.062	.526	.065	.434	.059	.378	.061	.405	.078	.925	.059	.393	.083	.313

not applicable in our case since our objective function is not differentiable for all $(e, d) \in [0, \frac{1}{2}]^2$. We discuss this point in Supplemental.

5 Experimental Evaluation

5.1 Empirical Setting

Our experiments[5] have a two-fold objective: *(i)* assessing the guarantees given by the associated PAC-Bayesian bounds, and *(ii)* comparing the performance of the different C-bound based algorithms in terms of risk optimization. To achieve this objective, we compare the three algorithms proposed in this paper to the following state-of-the-art PAC-Bayesian methods for majority vote learning:

- MinCq [31] and CB-Boost [1] that are based on the minimization of the empirical C-Bound. For comparison purposes and since MinCq and CB-Boost do not explicitly minimize a PAC-Bayesian bound, we report the bound values of Theorem 6 instantiated with the models learned;
- The algorithm proposed by Masegosa *et al.* [24] that optimizes a PAC-Bayesian bound on $r_{\mathcal{D}}^{\mathrm{MV}}(\mathcal{Q}) \leq 4e_{\mathcal{D}}(\mathcal{Q})$ (see Theorem 9 of [24]);

[5] Experiments are done with PyTorch [29] and CVXPY [9]. The source code is available at `https://github.com/paulviallard/ECML21-PB-CBound`.

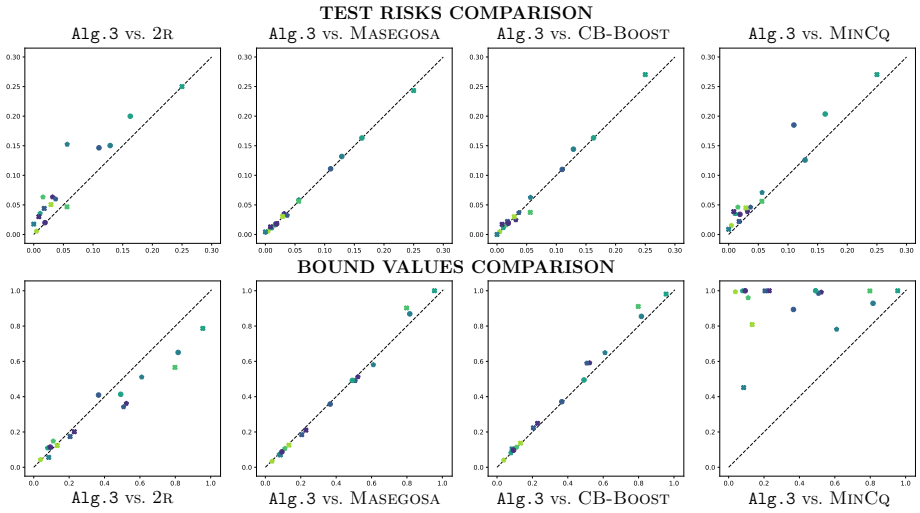

Fig. 1. Pairwise comparisons of the test risks (first line) and the bounds (second line) between Algorithm 3 and the baseline algorithms. Algorithm 3 is represented on the x-axis, while the y-axis is used for the other approaches. Each dataset corresponds to a point in the plot and a point above the diagonal indicates that Algorithm 3 is better.

- An algorithm[6], denoted by 2R, to optimize a PAC-Bayesian bound based only on the Gibbs risk [21]: $r_{\mathcal{D}}^{\mathrm{MV}}(\mathcal{Q}) \leq 2r_{\mathcal{D}}(\mathcal{Q}) \leq 2\overline{\mathrm{kl}}(r_{\mathcal{S}}(\mathcal{Q})|\psi_r(\mathcal{Q}))$.

We follow a general setting similar to the one of Masegosa *et al.* [24]. The prior distribution $\mathcal{P}$ on $\mathcal{H}$ is set as the uniform distribution, and the voters in $\mathcal{H}$ are decision trees: 100 trees are learned with 50% of the training data (the remaining part serves to learn the posterior $\mathcal{Q}$). More precisely, for each tree $\sqrt{d}$ features of the d-dimensional input space are selected, and the trees are learned by using the Gini criterion until the leaves are pure.

In this experiment, we consider 16 classic datasets[7] that we split into a train set $\mathcal{S}$ and a test set $\mathcal{T}$. We report for each algorithm in Table 1, the test risks (on $\mathcal{T}$) and the bound values (on $\mathcal{S}$, such that the bounds hold with prob. at least 95%). The parameters of the algorithms are selected as follows. **1) For** Masegosa's algorithm we kept the default parameters [24]. **2) For** all the other bounds minimization algorithms, we set T=2,000 iterations for all the datasets except for adult, fash and mnist where T=200. We fix the objective functions with λ=100, and we use COCOB-Backprop optimizer [27] as UPDATE-$\mathcal{Q}$ (its parameter remains the default one). For Algorithm 3, we fix the tolerance ϵ=.01, *resp.* ϵ=10^{-9}, to compute $\underline{\mathrm{kl}}$, *resp.* $\overline{\mathrm{kl}}$. Furthermore, the maximal number of iterations $T_{\max}$ in MAXIMIZE-e-d is set to 1,000. **3) For** MINCQ, we select the

[6] The algorithm 2R is similar to Algorithm 2, but without the numerator of the C-Bound (*i.e.*, the disagreement). More details are given in the Supplemental.

[7] An overview of the datasets is presented in the Supplemental.

margin parameter among 20 values uniformly distributed in $[0, \frac{1}{2}]$ by 3-fold cross validation. Since this algorithm is not scalable due to its high time complexity, we reduce the training set size to m=400 when learning with MinCq on the large datasets: adult, fash and mnist (MinCq is still competitive with less data on this datasets). For CB-Bound which is based on a Boosting approach, we fix the maximal number of boosting iterations to 200.

5.2 Analysis of the Results

Beforehand, we compare only our three self-bounding algorithms. From Table 1, as expected we observe that Algorithm 1 based on the McAllester's bound (that is more interpretable but less tight) provides the worst bound. Algorithm 3 always provides tighter bounds than Algorithms 1 and 2, and except for letter:DvsO, fash:COvsSH, and fash:SAvsBO Algorithm 3 leads to the lowest test risks. We believe that Algorithm 3 based on the Lacasse's bound provides lower bounds than Algorithm 2 based on the Seeger's bound because the Lacasse's approach bounds simultaneously the joint error and the disagreement. Algorithm 3 appears then to be the best algorithm among our three self-bounding algorithms that minimize a PAC-Bayesian C-Bound.

In the following we focus then on comparing our best contribution represented by Algorithm 3 to the baselines; Figure 1 summarizes this comparison.

First, 2R gives the lowest bounds among all the algorithms, but at the price of the largest risks. This clearly illustrates the limitation of considering *only* the Gibbs risk as an estimator of the majority vote risk: As discussed in Section 2.2, the Gibbs risk is an unfair estimator since an increase of the diversity between the voters can have a negative impact on the Gibbs risk.

Second, compared to Masegosa's approach, the results are comparable: Algorithm 3 tends to provide tighter bounds, and similar performances that lie in the same order of magnitude, as illustrated in Table 1. This behavior was expected since minimizing the bound of Masegosa [24] or the PAC-Bayesian C-Bound boils down to minimize a trade-off between the risk and the disagreement.

Third, compared to empirical C-bound minimization algorithms, we see that Algorithm 3 outputs better results than CB-Boost and MinCq for which the difference is significative and the bounds are close to 1 (*i.e.*, non-informative). Optimizing the risk bounds tend then to provide better guarantees that justify that optimizing the empirical C-bound is often too optimistic.

Overall, from these experiments, our Algorithm 3 is the one that provides the best trade-off between having good performances in terms of risk optimization and ensuring good theoretical guarantees with informative bounds.

6 Conclusion and Future Work

In this paper, we present new learning algorithms driven by the minimization of PAC-Bayesian generalization bounds based on the C-Bound. More precisely, we

propose to solve three optimization problems, each one derived from an existing PAC-Bayesian bound. Our methods belong to the class of *self-bounding* learning algorithms: The learned predictor comes with a tight and statistically valid risk upper bound. Our experimental evaluation has confirmed the quality of the learned predictor and the tightness of the bounds with respect to state-of-the-art methods minimizing the C-Bound.

As future work, we would like to study extensions of this work to provide meaningful bounds for learning (deep) neural networks. In particular, an interesting perspective would be to adapt the C-Bound to control the diversity and the weights in a neural network.

Acknowledgements

This work was supported by the French Project APRIORI ANR-18-CE23-0015. Moreover, Pascal Germain is supported by the NSERC Discovery grant RGPIN-2020-07223 and the Canada CIFAR AI Chair Program. The authors thank Rémi Emonet for insightful discussions.

References

1. Bauvin, B., Capponi, C., Roy, J., Laviolette, F.: Fast greedy C-bound minimization with guarantees. Mach. Learn. (2020)
2. Bellet, A., Habrard, A., Morvant, E., Sebban, M.: Learning A Priori Constrained Weighted Majority Votes. Mach. Learn. (2014)
3. Boser, B., Guyon, I., Vapnik, V.: A training algorithm for optimal margin classifiers. In: COLT (1992)
4. Boyd, S., Vandenberghe, L.: Convex Optimization. Cambridge Univ. Press (2004)
5. Breiman, L.: Bagging Predictors. Mach. Learn. (1996)
6. Breiman, L.: Random Forests. Mach. Learn. (2001)
7. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. (1995)
8. Danskin, J.: The Theory of Max-Min, with Applications. SIAM J. Appl. Math. (1966)
9. Diamond, S., Boyd, S.: CVXPY: A Python-embedded modeling language for convex optimization. J. Mach. Learn. Res. (2016)
10. Dietterich, T.: Ensemble methods in machine learning. In: International workshop on multiple classifier systems (2000)
11. Dziugaite, G.K., Roy, D.: Computing Nonvacuous Generalization Bounds for Deep (Stochastic) Neural Networks with Many More Parameters than Training Data. In: UAI (2017)
12. Freund, Y.: Self Bounding Learning Algorithms. In: COLT (1998)
13. Freund, Y., Schapire, R.: Experiments with a New Boosting Algorithm. In: ICML (1996)
14. Germain, P., Lacasse, A., Laviolette, F., Marchand, M.: PAC-Bayesian Learning of Linear Classifiers. In: ICML (2009)
15. Germain, P., Lacasse, A., Laviolette, F., Marchand, M., Roy, J.: Risk Bounds for the Majority Vote: From a PAC-Bayesian Analysis to a Learning Algorithm. J. Mach. Learn. Res. (2015)

16. Grant, M., Boyd, S., Ye, Y.: Disciplined Convex Programming. In: Global optimization (2006)
17. Kervadec, H., Dolz, J., Yuan, J., Desrosiers, C., Granger, E., Ayed, I.B.: Constrained Deep Networks: Lagrangian Optimization via Log-Barrier Extensions. CoRR **abs/1904.04205** (2019)
18. Kingma, D., Ba, J.: Adam: A Method for Stochastic Optimization. In: ICLR (2015)
19. Kuncheva, L.: Combining pattern classifiers: methods and algorithms. John Wiley & Sons (2014)
20. Lacasse, A., Laviolette, F., Marchand, M., Germain, P., Usunier, N.: PAC-Bayes Bounds for the Risk of the Majority Vote and the Variance of the Gibbs Classifier. In: NIPS (2006)
21. Langford, J., Shawe-Taylor, J.: PAC-Bayes & Margins. In: NIPS (2002)
22. Lorenzen, S.S., Igel, C., Seldin, Y.: On PAC-Bayesian bounds for random forests. Mach. Learn. (2019)
23. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards Deep Learning Models Resistant to Adversarial Attacks. In: ICLR (2018)
24. Masegosa, A., Lorenzen, S.S., Igel, C., Seldin, Y.: Second Order PAC-Bayesian Bounds for the Weighted Majority Vote. In: NeurIPS (2020)
25. McAllester, D.: Some PAC-Bayesian Theorems. Mach. Learn. (1999)
26. McAllester, D.: PAC-Bayesian Stochastic Model Selection. Mach. Learn. (2003)
27. Orabona, F., Tommasi, T.: Training Deep Networks without Learning Rates Through Coin Betting. In: NIPS (2017)
28. Parrado-Hernández, E., Ambroladze, A., Shawe-Taylor, J., Sun, S.: PAC-Bayes Bounds with Data Dependent Priors. J. Mach. Learn. Res. (2012)
29. Paszke, A. *et al.*: PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: NeurIPS (2019)
30. Reeb, D., Doerr, A., Gerwinn, S., Rakitsch, B.: Learning Gaussian Processes by Minimizing PAC-Bayesian Generalization Bounds. In: NeurIPS (2018)
31. Roy, J., Laviolette, F., Marchand, M.: From PAC-Bayes Bounds to Quadratic Programs for Majority Votes. In: ICML (2011)
32. Roy, J., Marchand, M., Laviolette, F.: A Column Generation Bound Minimization Approach with PAC-Bayesian Generalization Guarantees. In: AISTATS (2016)
33. Seeger, M.: PAC-Bayesian Generalisation Error Bounds for Gaussian Process Classification. J. Mach. Learn. Res. (2002)
34. Shawe-Taylor, J., Williamson, R.: A PAC Analysis of a Bayesian Estimator. In: COLT (1997)

SUPPLEMENTAL OF

Self-Bounding Majority Vote Learning Algorithms by the Direct Minimization of a Tight PAC-Bayesian C-Bound

A Section 5—Details on the Datasets

Table 2 presents an overview of the datasets we use in our experiments (the split train/test, the dimensionality and the url to the dataset).

Table 2. Datasets overview.

	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{T}\rvert$	Dim.	Link
letter:OvsQ	1303	233	16	https://archive.ics.uci.edu/ml/datasets/letter+recognition
letter:DvsO	1331	227	16	https://archive.ics.uci.edu/ml/datasets/letter+recognition
letter:AvsB	1327	228	16	https://archive.ics.uci.edu/ml/datasets/letter+recognition
credit	327	326	46	https://archive.ics.uci.edu/ml/datasets/Credit+Approval
heart	149	148	13	https://archive.ics.uci.edu/ml/datasets/heart+disease
glass	107	107	9	https://archive.ics.uci.edu/ml/datasets/glass+identification
tictactoe	479	479	9	https://archive.ics.uci.edu/ml/datasets/Tic-Tac-Toe+Endgame
usvotes	218	217	48	https://archive.ics.uci.edu/ml/datasets/congressional+voting+records
wdbc	285	284	30	https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)
adult	30162	15060	104	https://archive.ics.uci.edu/ml/datasets/adult
mnist:1vs7	13007	2163	784	http://yann.lecun.com/exdb/mnist
mnist:4vs9	11791	1991	784	http://yann.lecun.com/exdb/mnist
mnist:5vs6	11339	1850	784	http://yann.lecun.com/exdb/mnist
fash:TOvsPU	12000	2000	784	https://github.com/zalandoresearch/fashion-mnist
fash:SAvsBO	12000	2000	784	https://github.com/zalandoresearch/fashion-mnist
fash:COvsSH	12000	2000	784	https://github.com/zalandoresearch/fashion-mnist

B Section 4.3—About Danskin's Theorem

As mentioned in the main paper, in the context of the justification of the function MAXIMIZE-*e*-*d* in Algorithm 3, we now discuss the possible application of Danskin's Theorem [8, Section I]. The statement of the theorem is as follows.

Theorem 7 (Danskin's Theorem). *Let $\mathcal{A} \subset \mathbb{R}^m$ be a compact set and $\phi : \mathbb{R}^n \times \mathcal{A} \to \mathbb{R}$ s.t. for all $\mathbf{a} \in \mathcal{A}$, we have that ϕ is continuously differentiable, then $\Phi(\mathbf{x}) = \max_{\mathbf{a}\in\mathcal{A}} \phi(\mathbf{x}, \mathbf{a})$ is directionally differentiable with directional derivatives*

$$\Phi'(\mathbf{x}, \mathbf{d}) = \max_{\mathbf{a}\in\mathcal{A}^*} \langle \mathbf{d}, \nabla_{\mathbf{x}} \phi(\mathbf{x}, \mathbf{a}) \rangle ,$$

where $\mathcal{A}^* = \{\mathbf{a}^* \mid \phi(\mathbf{x}, \mathbf{a}^*) = \max_{\mathbf{a}\in\mathcal{A}} \phi(\mathbf{x}, \mathbf{a})\}$ and $\langle \cdot, \cdot \rangle$ is the dot product.

To optimize a problem $\min_{\mathbf{x}\in\mathbb{R}^n} \Phi(\mathbf{x})$ with $\Phi(\mathbf{x}) = \max_{\mathbf{a}\in\mathcal{A}} \phi(\mathbf{x}, \mathbf{a})$, this theorem tells us that under several assumptions, if we know a maximizer $\mathbf{a} \in \mathcal{A}$, then, we have an analytical expression of the directional derivatives of $\Phi(\mathbf{x})$. Thus, from this theorem, we also know a gradient to minimize the problem $\min_{\mathbf{x}\in\mathbb{R}^n} \Phi(\mathbf{x})$.

Corollary 1 (Madry et al. [23]). *Assuming that the conditions of Theorem 7 are fulfilled and let* $\mathbf{a}^* \in \mathcal{A}^*$ *be a maximizer of* ϕ*. If* $\mathbf{d} = \nabla_{\mathbf{x}}\phi(\mathbf{x}, \mathbf{a}^*)$ *with* $\|\mathbf{d}\|_2^2 > 0$ *then* $-\mathbf{d}$ *is a descent direction for* $\Phi(\mathbf{x})$, i.e., $\Phi'(\mathbf{x}, \mathbf{d}) > 0$.

Proof. By definition of the directional derivative $\Phi'(\mathbf{x}, \mathbf{d})$ and the direction $\mathbf{d}$, we have:

$$\begin{aligned}\Phi'(\mathbf{x}, \mathbf{d}) &= \max_{\mathbf{a} \in \mathcal{A}^*} \langle \mathbf{d}, \nabla_{\mathbf{x}}\phi(\mathbf{x}, \mathbf{a}) \rangle \\ &= \max_{\mathbf{a} \in \mathcal{A}^*} \langle \nabla_{\mathbf{x}}\phi(\mathbf{x}, \mathbf{a}^*), \nabla_{\mathbf{x}}\phi(\mathbf{x}, \mathbf{a}) \rangle \geq \|\nabla_{\mathbf{x}}\phi(\mathbf{x}, \mathbf{a}^*)\|_2^2 > 0.\end{aligned}$$

Then, for each iteration of the min/max problem optimization, we can *(i)* optimize the inner maximization problem, *(ii)* fix the maximizer $\mathbf{a}^* \in \mathcal{A}$ and apply a gradient descent step with the derivative $\nabla_{\mathbf{x}}\phi(\mathbf{x}, \mathbf{a}^*)$. However, as we mentioned in the main paper, the assumptions are not fulfilled in our case to apply Theorem 7 since our inner objective in Equation (8) or its approximation $C^{\mathrm{L}}(e, d) - \mathbf{B}\left[d - 2\sqrt{\min\left(e, \frac{1}{4}\right)} - 2e\right] - \mathbf{B}\left[d - \frac{1}{2}\right] - \mathbf{B}_\lambda\left[\mathrm{kl}\left(e_{\mathcal{S}}(\mathcal{Q}), d_{\mathcal{S}}(\mathcal{Q}) \| e, d\right) - \kappa(\mathcal{Q})\right]$ is not differentiable everywhere in the compact set $[0, \frac{1}{2}]^2$. However, we never encounter problematic cases and this strategy is thus valid for optimizing our proposed approximation. In practice, we have found that it is indeed an efficient and sound strategy.

C Section 5—About Optimizing $2\overline{\mathrm{kl}}\left(r_{\mathcal{S}}(\mathcal{Q}) | \psi_r(\mathcal{Q})\right)$

To minimize the bound $2(\overline{\mathrm{kl}}\left(r_{\mathcal{S}}(\mathcal{Q}) \mid \psi_r(\mathcal{Q})\right))$, we adopt the algorithm (denoted as 2R in the setting description of the experiments of Section 5) similar to Algorithm 2. Indeed, we use instead the objective function:

$$\min_{\mathcal{Q}} 2(\overline{\mathrm{kl}}\left(r_{\mathcal{S}}(\mathcal{Q}) \mid \psi_r(\mathcal{Q})\right)). \tag{10}$$

The algorithm is described in Algorithm 4 below.

Algorithm 4 Minimization of Equation (10) by GD

Given: learning sample $\mathcal{S}$, prior distribution $\mathcal{P}$ on $\mathcal{H}$, update function UPDATE-$\mathcal{Q}$
Hyperparameters: number of iterations T
function MINIMIZE-$\mathcal{Q}$
  $\mathcal{Q} \leftarrow \mathcal{P}$
  for $t \leftarrow 1$ to T **do**
    Compute $\overline{\mathrm{kl}}\left(r_{\mathcal{S}}(\mathcal{Q}) \mid \psi_r(\mathcal{Q})\right)$ using COMPUTE-$\overline{\mathrm{kl}}(q|\psi)$.
    $\mathcal{Q} \leftarrow$UPDATE-$\mathcal{Q}(\overline{\mathrm{kl}}\left(r_{\mathcal{S}}(\mathcal{Q}) \mid \psi_r(\mathcal{Q})\right))$ (thanks to Equation (6))
return $\mathcal{Q}$

D Section 5—Additional Experiments

We report in Figure 2 and Figure 3, the empirical joint error and disagreement obtained on the different datasets. As for Table 1 and Figure 1, this illustrates that the solutions found by `Alg.3`, MASEGOSA and CB-BOOST are similar while MINCQ and 2R provide very different solutions.

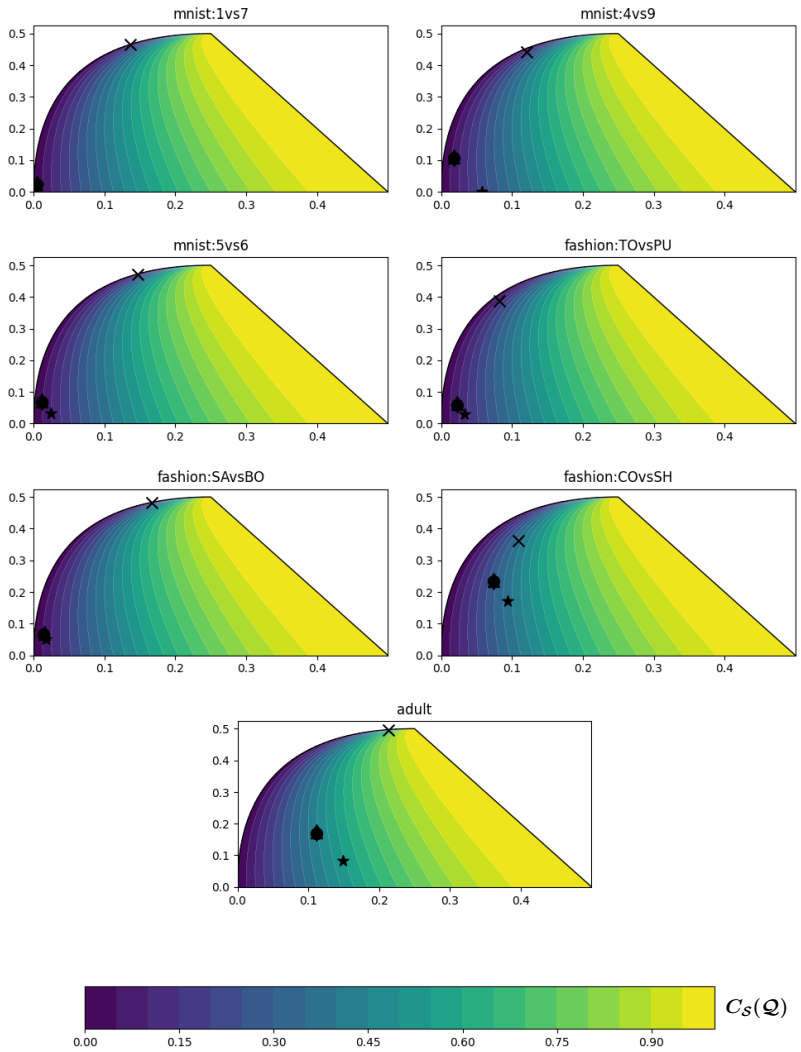

Fig. 2. Representation of all the possible values of the empirical C-Bound $C_S(\mathcal{Q})$ in function of the disagreement $d_S(\mathcal{Q})$ (y-axis) and joint error $e_S(\mathcal{Q})$ (x-axis). We report the values obtained on different datasets by `Alg.3` (⧫), MASEGOSA (▲), 2R (★), `CB-Boost` (•), and `MinCq` (×).

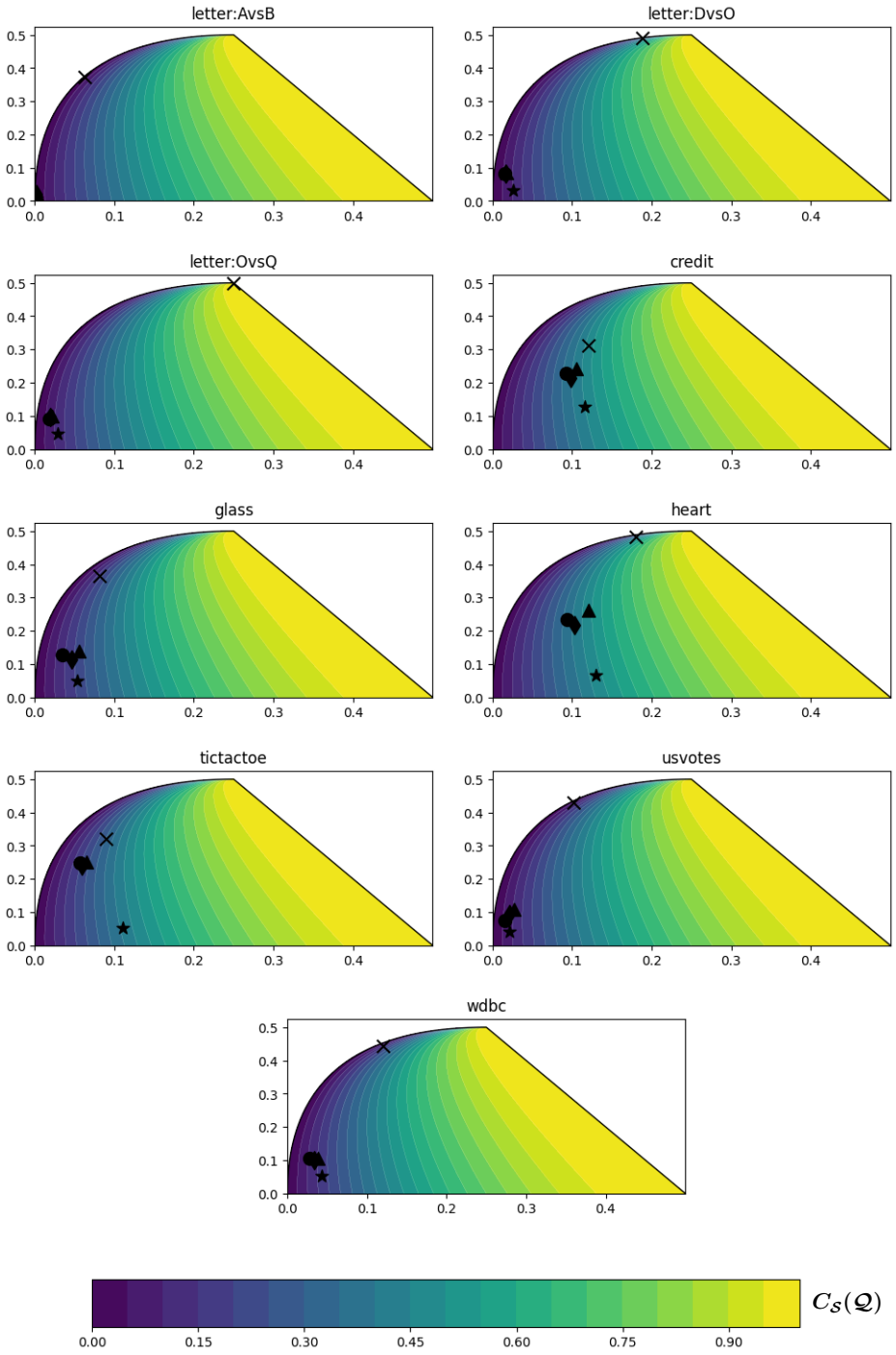

Fig. 3. Representation of all the possible values of the empirical C-Bound $C_{\mathcal{S}}(\mathcal{Q})$ in function of the disagreement $d_{\mathcal{S}}(\mathcal{Q})$ (y-axis) and joint error $e_{\mathcal{S}}(\mathcal{Q})$ (x-axis). We report the values obtained on different datasets by `Alg.3` (⧫), MASEGOSA (▲), 2R (★), `CB-Boost` (•), and `MinCq` (×).