PAC-Bayesian Computation

Contributions to learning and certification strategies for randomised classification algorithms

Omar Rivasplata

A dissertation submitted in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

in

Statistical Learning Theory

Department of Computer Science
University College London

March 25, 2022

I, Omar Rivasplata, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

Risk bounds, which are also called generalisation bounds in the statistical learning literature, are important objects of study because they give some information on the expected error that a predictor may incur on randomly chosen data points. In classical statistical learning, the analyses focus on individual hypotheses, and the aim is deriving risk bounds that are valid for the data-dependent hypothesis output by some learning method. Often, however, such risk bounds are valid uniformly over a hypothesis class, which is a consequence of the methods used to derive them, namely the theory of uniform convergence of empirical processes. This is a source of looseness of these classical kinds of bounds which has lead to debates and criticisms, and motivated the search of alternative methods to derive tighter bounds.

The PAC-Bayes analysis focuses on distributions over hypotheses and randomised predictors defined by such distributions. Other prediction schemes can be devised based on a distribution over hypotheses, however, the randomised predictor is a typical starting point. Lifting the analysis to distributions over hypotheses, rather than individual hypotheses, makes available sharp analysis tools, which arguably account for the tightness of PAC-Bayes bounds. Two main uses of PAC-Bayes bounds are (1) risk certification, and (2) cost function derivation. The first consists of evaluating numerical risk certificates for the distributions over hypotheses learned by some method, while the second consists of turning a PAC-Bayes bound into a training objective, to learn a distribution by minimising the bound. This thesis revisits both kinds of uses of PAC-Bayes bounds. We contribute results on certifying the risk of randomised kernel and neural network classifiers, adding evidence to the success of PAC-Bayes bounds at delivering tight certificates. This thesis proposes the name "PAC-Bayesian Computation" as a generic name to encompass the class of methods that learn a distribution over hypotheses by minimising a PAC-Bayes bound (i.e. the second use case described above: cost function derivation), and reports an interesting case of PAC-Bayesian Computation leading to self-certified learning: we develop a learning and certification strategy that uses all the available data to produce a predictor together with a tight risk certificate, as demonstrated with randomised neural network classifiers on two benchmark data sets (MNIST, CIFAR-10).

Impact Statement

The contents of the introductory Chapter 1 and the background Chapter 2 are of interest to the theoreticians and possibly also to some practitioners in the machine learning community. In particular, the presentation of PAC-Bayes bounds given in Section 2.2 might be of interest to anyone seeking to familiarise themselves with these kinds of bounds and the related literature.

The contents of Chapter 3, Chapter 4, and Chapter 5, which are based on my papers Rivasplata et al. [2020], Rivasplata et al. [2018], and Pérez-Ortiz et al. [2021b], respectively, may be of interest to theoreticians and practitioners in the machine learning community. These works could influence machine learning research inside and outside the academia by (1) clarifying the ability of PAC-Bayes bounds to deliver tight risk certificates for randomised classifiers, and (2) showing the advantages of using these kinds of bounds as learning objectives. In particular, the work of Chapter 5 is likely to be of high relevance, considering that this work deals with training strategies and risk certificates for deep learning models, which are building blocks of many machine learning systems and are increasingly used to make decisions that have a strong impact on society, industry, and individuals. However, considering that the contents of these chapters deal with either theoretical results or experimental results on benchmark data sets, it is fair to say that their impact on society necessarily must be linked to the particular user-specific applications where the results may be instantiated.

Acknowledgements

I gratefully acknowledge the generous sponsorship from DeepMind for carrying out research studies on machine learning at University College London.

I had the privilege of being affiliated with DeepMind as a research scientist intern for three years in parallel with my research studies at University College London.

I had the privilege of being hosted by the the Department of Computer Science and the Centre for Artificial Intelligence at University College London.

I warmly thank John Shawe-Taylor for generously sharing knowledge, advice, guidance, encouragement, and overall a positive outlook on work and life.

I warmly acknowledge the help and support I received from many people at the UCL AI Centre and UCL Computer Science, and from many people at DeepMind.

During these four years of research studies at University College London, and the overlapping three-year internship at DeepMind, I have met outstanding researchers and wonderful people that I am happy to call friends.

Contents

List of Figures

List of Tables

Chapter 1

Prologue

A common question arising in learning theory aims to explain the generalisation ability of a learner: how can a learner ensure good ‘performance’ on unseen data?

Under statistical assumptions on the data-generating process, the question about generalisation of a learner may be approached in terms of a suitable measure of performance defined at population level, such as the expected error on a random data point from the same distribution that generated the data on which a classifier was trained in the case of supervised classification, or in other settings the ‘expected loss’ corresponding to a choice of loss function that makes sense for the learning task being considered. The zero-one loss makes sense for classification tasks, because it counts classification errors; while for regression a typical choice is the squared loss, which measures squared deviations from the mean. Whatever the case may be, when thinking of expected loss as measure of performance, ‘expected’ refers to expectation with respect to the distribution that generates random data points, which is called the *data-generating distribution*. Thus, the expected loss, also called the *risk* or *population loss* (defined formally in Section 2.1), is a measure of performance that is valid at population level, in the sense that it is valid for any random data point from a given distribution. Unfortunately, the population loss is inaccessible in most learning problems of interest, because the data-generating distribution is unknown.

One way to answer the question about generalisation of a learner is by studying upper bounds on the population loss. Such bounds are called *generalisation bounds* or *risk bounds* in the statistical learning literature. Intuitively speaking, if the upper bound is small, then this ensures that the quantity being upper-bounded by it—namely, the population loss—must also be small. Often the focus is on bounding the gap (i.e. the difference) between the population loss (the risk) and its empirical counterpart (the empirical risk) evaluated on a given data set of a given finite size. Then, giving upper bounds on the gap is equivalent to giving upper bounds on the risk in terms of the empirical risk and some other quantities. Due to its importance, this gap has received its own designation: it is usually referred to as the *generalisation gap* or the *risk gap*. The methods for bounding the gap are typically based on probability inequalities for sums of independent random variables, which bound the deviations of such sums from their means.

There are several types of generalisation bounds we care about in learning theory, with variations in the way they depend on the training data and the data-generating distribution. Specifically, this dependence refers to the quantities that upper-bound the generalisation gap. In the classical bounds (such as bounds based on the VC dimension[1]) such quantities depend neither on the data nor the distribution that generates the data. *Distribution-dependent* bounds are expressed in terms of quantities related to the data-generating distribution (e.g. functionals associated to this distribution) and possibly other quantities, but not the data in any way. These kinds of bounds can be helpful to study the behaviour of a learning method on different distributions—for example, some data-generating distributions might give faster convergence rates than others. Finally, *data-dependent* bounds are expressed in terms of empirical quantities that can be computed directly from data. These kinds of bounds can be useful for building and comparing predictors [Catoni, 2007], and also for self-bounding learning methods [Freund, 1998, Langford and Blum, 2003], among possibly other purposes.

In classical statistical learning the analyses focus on individual hypotheses, and the aim is bounding the risk of the hypothesis output by some learning method. The term *hypothesis* is just an alternative name for *predictor* or *classifier* according to the task being considered, for instance, a function that maps input feature vectors to class labels in the case of a classification task. Often, however, the risk bounds of classical statistical learning are valid uniformly over a hypothesis class, which is a consequence of the methods used to derive them. This is a source of limitations of the classical bounds which has lead to debates and criticisms, most notably among which is their notorious looseness when applied to complex models such as deep learning models. This has motivated the search of alternative methods to derive tighter bounds.

PAC-Bayes bounds are a newer kind of generalisation bounds whose study started with the works of McAllester [1998, 1999]. These seminal works contributed the first PAC-Bayes theorems and in fact coined the name 'PAC-Bayes' arguably taking inspiration from the work of Shawe-Taylor and Williamson [1997] who carried out a Probably Approximately Correct (PAC) analysis of a Bayesian-style predictor, which was so called because the predictor was defined by a data-dependent distribution over hypotheses. The usual PAC-Bayes analysis (details of which are outlined in Section 2.2 below) introduces a reference 'data-free' probability distribution on the hypothesis space, commonly called a 'prior' distribution, while the learned data-dependent distribution that is used to construct a prediction rule is called a 'posterior' distribution. This

[1] VC stands for Vapnik-Chervonenkis. The VC dimension was introduced by Vapnik and Chervonenkis [1971]. Generalisation bounds based on the VC dimension are presented with detailed proofs by Vapnik [1998], see also e.g. Devroye et al. [1997], Shalev-Shwartz and Ben-David [2014], Mohri et al. [2018].

terminology is well-established in the literature on PAC-Bayes bounds. However, in contrast to Bayesian learning, the PAC-Bayes prior acts as an analytical device and may or may not be used by the learner, and the PAC-Bayes posterior is unrestricted and so it may be different from the posterior that would be obtained through Bayesian inference. This pinpoints essential differences between the PAC-Bayes and Bayesian learning schools that are important to keep in mind.

PAC-Bayes analyses allow to derive distribution- or data- dependent generalisation bounds. There are many application areas that have used PAC-Bayes bounds, but there are essentially two ways that a PAC-Bayes bound is typically applied: either (1) use the bound to evaluate a risk certificate for a predictor defined by a distributions over hypotheses learned by some method, or (2) turn the bound into a learning method by searching a distribution over hypotheses that minimises the bound. The first use case may be referred to as *risk certification*, since this use case consists of producing a numerical risk certificate. For the second use case I propose[2] the name *PAC-Bayesian Computation* and the corresponding acronym PBC. This generic name may be a convenient way to refer to any learning method that converts a PAC-Bayes bound into a ‘cost function’ or ‘optimisation objective’ for learning a distribution over hypotheses.

The use cases of PAC-Bayes bounds just described define the two central themes of this thesis: risk certification for randomised predictors (where the numerical certificates are evaluated based on a PAC-Bayes bound); and PAC-Bayesian Computation (PBC) for learning data-dependent distributions over a given hypothesis class.

Then, by a combination of these two central themes, we develop *self-certified learning methods* [Pérez-Ortiz et al., 2021b], which are learning methods that use all the available data to produce simultaneously a predictor and a reasonably tight risk certificate that is valid at population level, i.e. the certificate is valid for any unseen data from the same distribution that generated the training data. Self-certified learning is a re-branding of self-bounding learning [Freund, 1998], and in this thesis we make a case for it using PAC-Bayes bounds. In particular, Chapter 5 below reports a promising instance of self-certified learning with randomised neural network classifiers, where a distribution over network weights is obtained via PAC-Bayesian Computation.

Outline of this thesis and its main contributions

- Chapter 2 covers background on statistical learning and PAC-Bayes bounds. The content of this chapter consists of previous knowledge that my research builds on. In particular, this chapter sets the framework and the mathematical notation for the thesis. This chapter also gives an overview of the related literature.

[2]Use of the first person is to highlight individual contributions or ideas of mine, and personal opinions; as opposed to contributions that emerged in collaborations (such as the works reported in Chapter 3, Chapter 4 and Chapter 5), and opinions or ideas held collectively.

- Chapter 3 is based on my paper Rivasplata et al. [2020], which reported original contributions of my research done during my research studies. This work proposed an extension of the PAC-Bayes analysis to stochastic kernels, and revisiting the usual assumptions on which PAC-Bayes bounds are based. In this work I am fully responsible for the generic PAC-Bayes theorem with stochastic kernels, and for the insight that this theorem is valid with (a) data-dependent priors and (b) unbounded loss functions. I am largely responsible for the results presented in Section 2.1, Section 2.2 and Section 2.3 of the paper, which were developed in collaboration with Ilja Kuzborskij. I am almost entirely responsible for writing the paper.

- Chapter 4 is based on my paper Rivasplata et al. [2018], which reported original contributions of my research done during my research studies. This work combined algorithmic stability and a PAC-Bayes bound in order to obtain a tight bound for randomised support vector machine classifiers. In this work I am largely responsible for the proof of the stability-based PAC-Bayes bound presented in this paper, which was developed in collaboration with John Shawe-Taylor and Csaba Szepesvári; and for designing the framework for the experiments, which were carried out by Emilio Parrado-Hernández and Csaba Szepesvári. Also I am fully responsible for the section on Gaussian distributions over infinite-dimensional Hilbert spaces, which elicited positive reactions from the reviewers who requested to move this section from an appendix into the main paper. I am almost entirely responsible for writing the paper.

- Chapter 5 is based on my paper Pérez-Ortiz et al. [2021b], which reported original contributions of my research done during my research studies. This work studied three (probabilistic) neural network training objectives derived from PAC-Bayes bounds, and gave an empirical exploration of the properties of the classifiers obtained by optimising these objectives. In particular, I am fully responsible for the derivation of the PAC-Bayes-quadratic bound, which is the basis for a new training objective proposed by this work. I am largely responsible for developing the theoretical framework of this work and for designing the training strategies implemented in the empirical studies, which were carried out by María Pérez-Ortiz, with whom I share credit as "main contributor" since María was also largely involved in designing the training strategies and writing the paper.

Chapter 2

Background on Statistical Learning and PAC-Bayes bounds

Chapter Layout. This chapter is organised as follows. Section 2.1 defines important concepts and sets the notation to be used throughout the thesis. PAC-Bayes bounds are presented in Section 2.2. In particular, we recall a general PAC-Bayes theorem (see Theorem 2.1) and we discuss how several known PAC-Bayes bounds may be obtained as consequences of it. Also, in this section we discuss the two main uses of PAC-Bayes bounds: risk certification (Section 2.2.1) and cost function derivation (Section 2.2.2). We also discuss some of the related literature (Section 2.2.3). Finally, the technical Section 2.3 gives a proof of the general PAC-Bayes theorem.

2.1 Definitions and Notation

The context of this thesis is the statistical learning model where the learner observes training data $S = (Z_1, Z_2, \dots, Z_n)$ randomly drawn from a distribution over the space of size-n samples $\mathcal{Z}^n =: \mathcal{S}$ for some fixed positive integer n. The basic example space is of the form $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ for a supervised learning problem where the input space is $\mathcal{X} \subset \mathbb{R}^d$ and the label set is $\mathcal{Y}$. The latter is a finite set in classification tasks, and it is an interval in prediction tasks. Then, in this setting, each $Z_i = (X_i, Y_i)$ encodes an input-label pair available to the learner.

The goal of the learning process is finding a function $\hat{h} : \mathcal{X} \to \mathcal{Y}$ that describes the relationship between feature vectors $x \in \mathcal{X}$ and labels $y \in \mathcal{Y}$. It is well-known that searching over all possible functions is computationally intractable and at the same time it may lead to finding a function that fits the training data perfectly but does very poorly on unseen data. Therefore, the search is restricted to a suitably chosen function class $\mathcal{H}$. To find a good function $\hat{h}$, we can define a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ that penalises the discrepancies between predicted labels $\hat{h}(x)$ and true labels y. Then, a classical learning rule aims to find a function that minimises the empirical loss evaluated on the data set $S = ((X_1,Y_1),\dots,(X_n,Y_n))$ that is available for training:

$$\hat{h} \in \operatorname*{arg\,min}_{h\in\mathcal{H}} \sum_{i=1}^{n} \ell(h(X_i),Y_i)$$

This is called Empirical Risk Minimisation (ERM) [Vapnik, 1992], and is the basis of many learning strategies. It might be worthwhile to recall at this point that, in Vapnik's learning framework, the tripe $(\mathcal{H},\mathcal{Z},\ell)$ characterises a learning problem [Vapnik, 1995, 1998]. Moreover, in this general framework the loss function is defined as a function $\ell : \mathcal{H}\times\mathcal{Z} \to [0,\infty)$ that acts on each pair (h,z) of hypothesis and data point. The choice of loss function ℓ is problem-dependent, as are the form of the hypothesis space $\mathcal{H}$ and the example space $\mathcal{Z}$. As already mentioned, in supervised learning problems we have $\mathcal{Z} = \mathcal{X}\times\mathcal{Y}$ where $\mathcal{X} \subset \mathbb{R}^d$ is the set of inputs, and $\mathcal{Y}$ is the set of labels. For instance, a regression task is defined as the problem when $\mathcal{Y} = \mathbb{R}$ and the loss function is the squared loss, namely $\ell(h,z) = (y-h(x))^2$, where $z = (x,y)$ encodes the input-label pair. A binary classification task is the problem where $\mathcal{Y} = \{0,1\}$ (or $\mathcal{Y} = \{-1,+1\}$) and the loss is set to be the zero-one loss: $\ell(h,z) = \mathbb{I}[y \neq h(x)]$, where $\mathbb{I}[\cdot]$ is an indicator function which equals 1 when its predicate is true, and equals 0 otherwise.

The hypothesis space characterises the kinds of functions that the learner is allowed to search. Throughout this thesis we focus on parametric function classes $\mathcal{H} = \{h_w \mid w \in \mathcal{W}\}$ and we assume that the parameter set is $\mathcal{W} \subset \mathbb{R}^p$. The parameter dimension p may or may not coincide with the input dimension d. We think of $w \in \mathcal{W}$ as a *weight vector*, and it is understood that each possible weight vector $w \in \mathcal{W}$ maps to a predictor function $h_w : \mathcal{X} \to \mathcal{Y}$. A classical example is that of predictors $h_w(x) = \sigma(w^\top x - b)$ based on linear functions (if $b = 0$) or their affine translations (if $b \neq 0$ in general) and a threshold function $\sigma : \mathbb{R} \to \mathcal{Y}$ that maps reals to the corresponding labels. Another example of interest is the case of neural network functions, whose form is more involved, but suffice it to say here that in this case the 'weight vector' corresponds to a flattened version of the matrices of connection weights. Since the focus is on a parametric function class, the loss function is presented as a function $\ell : \mathcal{W}\times\mathcal{Z} \to [0,\infty)$ acting on each possible weight vector w and each possible data point z. Thus, $\ell(w,z)$ is interpreted as the loss of the hypothesis h_w corresponding to weight vector w, on a given data point z.

In order to connect performance on training data with that on unseen data, some assumption on the data-generating process needs to be made. At an extreme case, if there are no restrictions on how the data are generated, then it is easy to create problems in which learning is impossible. For instance, we could consider a prediction problem aiming to learn to predict the values of a cubic function, combined with the artificial data-generating example where all training data is generated by a quadratic function; then the lack of a sensible connection between training and testing data implies that any learner trained on the former is doomed to do poorly on the latter. In this thesis we rely on statistical assumptions to connect training and unseen data.

Typically it is assumed that $Z_1, \dots, Z_n$ are independent and share a common distribution[1] $P_1 \in \mathcal{M}_1(\mathcal{Z})$. This is called the i.i.d. data assumption, which means that the random data points are *independent and identically distributed*. The subscript in the notation P_1 is meant to suggest that this is the distribution that generates one random example. We write $Z \sim P_1$ to indicate that the random variable Z is drawn from P_1. Accordingly, we may write $Z_1, \dots, Z_n \overset{\text{i.i.d.}}{\sim} P_1$ to indicate the i.i.d. data assumption. In a general setting, it could be assumed that the size-n random sample $S = (Z_1, Z_2, \dots, Z_n)$ is generated by a distribution $P_n \in \mathcal{M}_1(\mathcal{Z}^n)$, written $S \sim P_n$, where the distribution P_n over $\mathcal{Z}^n$ may in general correlate the components. The i.i.d. data assumption is equivalent to assuming that $P_n = P_1^{\otimes n}$ is the product distribution of n copies of P_1. Throughout this thesis we rely on the i.i.d data assumption unless explicitly stated otherwise.

In principle, a learner should aim to find a weight $w \in \mathcal{W}$ that minimises the *expected loss* on random examples, also called the *population loss* or the *risk*:

$$L(w) = \mathbb{E}[\ell(w,Z)] = \int_{\mathcal{Z}} \ell(w,z) P_1(dz)\,. \tag{2.1}$$

The name ‘population loss’ is meant to suggest that $L(w)$ is a population-level quantity, in the sense that it is defined as the expected loss under a given population distribution P_1. However, the distribution P_1 is unknown in most practical scenarios, which implies that the risk functional $L(\cdot)$ defined in Eq. (2.1) is an unobservable objective. This justifies why in practice the expected loss is replaced with its empirical counterpart, the average loss over the sample. Replacing the expected loss with the average loss on the data gives rise to an observable objective called the *empirical risk* functional:

$$\hat{L}_S(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(w, Z_i)\,. \tag{2.2}$$

Empirical risk minimisation (ERM) is minimisation of the empirical risk functional. This is approached in a problem-dependent manner, but in practice the minimisation of $\hat{L}_S(\cdot)$ is often done with some version of gradient descent. Then, to justify that ERM produces a solution w with a small risk, something must be said about the connection between $\hat{L}_S(w)$ and $L(w)$.

From the trivial identity $L(w) = \hat{L}_S(w) + L(w) - \hat{L}_S(w)$ it is clear that $L(w)$ is small whenever the empirical risk $\hat{L}_S(w)$ and the gap $L(w) - \hat{L}_S(w)$ are both small. As already mentioned, the empirical risk is an observable quantity, in the sense that it is computable directly from data; hence a small $\hat{L}_S(w)$ can be ensured in a data-dependent way. Indeed, the goal of ERM is finding a weight vector w for which $\hat{L}_S(w)$ is as small as possible. However, the gap $L(w) - \hat{L}_S(w)$ is not observable, because $L(w)$ is defined in terms of the data-generating distribution, which is inaccessible. Then, a reasonable way to ensure a small gap is via upper-bounding it.

[1]We write $\mathcal{M}_1(\mathcal{X})$ to denote the family of probability measures over a set $\mathcal{X}$.

Using deviation inequalities which essentially say that the event $L(w) - \hat{L}_S(w) > \varepsilon$ has exponentially small probability, statistical learning theory has derived risk bounds of the form $L(w) \leq \hat{L}_S(w) + \varepsilon(\delta, n)$ that hold with probability of at least $1 - \delta$ over the draw of size-n i.i.d. random samples, for a given choice of confidence parameter $\delta \in (0,1)$. Notice that $\varepsilon(\delta, n)$ is an upper bound on the gap. The basic analysis gives such a bound for an arbitrary hypothesis, while the most important results of the theory are risk bounds that hold simultaneously for all hypotheses in a given hypothesis class, in which case the upper bound on the gap is of the form $\varepsilon(\delta, n, C)$ where C is some notion of complexity of the hypothesis class, such as VC dimension [Vapnik and Chervonenkis, 1971] or Rademacher complexity [Koltchinskii and Panchenko, 2000].

A given risk bound may be applied to the solution found by a particular learning strategy, e.g. ERM over the given class, since the bound holds for any member of the class. Alternatively, a risk bound could be turned into a learning strategy, by searching a hypothesis (weight vector) that minimises the bound. This strategy involves using the bound repeatedly, but this does not affect the form of the bound because the bound holds uniformly over the class being searched. The bound-minimisation strategy is the basis of other learning strategies such as Structural Risk Minimisation (SRM) and self-bounding learning algorithms.

N.B.: The previous paragraphs are based on knowledge that I have accumulated over the years, likely references are the survey Boucheron et al. [2005] and books such as Devroye et al. [1997], Shalev-Shwartz and Ben-David [2014], and Mohri et al. [2018]; but there are other sources.

2.2 PAC-Bayes bounds

A randomised predictor is defined by a probability distribution over the weight space. While in principle an arbitrarily chosen distribution over weights could be used for defining a randomised predictor, the most interesting case is that of a data-dependent distribution Q_S which is obtained as the outcome of the training process based on the data set S. The notation Q_S is meant to indicate the data-dependence[2] of this distribution over weights. Then, given a fresh input X, the randomised predictor predicts its label by drawing a weight vector W at random according to Q_S and applying the predictor h_W to X. Each new prediction requires a fresh draw. Other prediction rules can be devised based on a distribution over hypotheses, however, the randomised predictor is a typical starting point in the PAC-Bayes literature.

[2]Formally, a data-dependent distribution over $\mathcal{W}$ is a stochastic kernel from $\mathcal{S}$ to $\mathcal{W}$. The definition of stochastic kernel is presented in Chapter 3 below, where discussions and an illustration are also given. This formalisation of data-dependent distributions over predictors was covered recently in my paper Rivasplata et al. [2020], where references were also pointed out for the concept of stochastic kernel, which is well-known in the literature on probability and stochastic processes.

One way, which we adopt in this thesis, to measure the performance of the randomised predictor corresponding to the distribution Q over $\mathcal{W}$ is to use the Q-weighted losses, since these are the expected losses over the random draws of weights defining the randomised predictor Q. Accordingly, the population loss of Q becomes

$$L(Q) = \int_{\mathcal{W}} L(w) Q(dw) \,, \tag{2.3}$$

and the empirical loss of Q becomes

$$\hat{L}_S(Q) = \int_{\mathcal{W}} \hat{L}_S(w) Q(dw) \,. \tag{2.4}$$

This notation extends the loss notions $L(w)$ and $\hat{L}_S(w)$ previously defined for a given weight w, to corresponding notions $L(Q)$ and $\hat{L}_S(Q)$ for a given distribution Q over weights. This notation is well-suited for PAC-Bayes bounds (given below) in view of its intuitive and readable nature: $L(Q)$ and $\hat{L}_S(Q)$ are losses of the randomised predictor defined by the distribution Q.

To introduce the promised PAC-Bayes bounds we need to recall some further definitions. Given two probability distributions $Q, Q' \in \mathcal{M}_1(\mathcal{W})$, the Kullback-Leibler (KL) divergence of Q from Q', also known as relative entropy of Q given Q', is defined as follows:

$$\mathrm{KL}(Q \| Q') = \int_{\mathcal{W}} \log\left(\frac{dQ}{dQ'}\right) dQ \,.$$

This equation defines $\mathrm{KL}(Q\|Q')$ when dQ/dQ', the Radon-Nikodym derivative of Q with respect to Q', is defined; otherwise $\mathrm{KL}(Q\|Q') = \infty$.

For $q, q' \in [0,1]$ we define

$$\mathrm{kl}(q\|q') = q \log\left(\frac{q}{q'}\right) + (1-q)\log\left(\frac{1-q}{1-q'}\right) , \tag{2.5}$$

which is called the binary KL divergence, and is the divergence of the Bernoulli distribution with parameter q from the Bernoulli distribution with parameter q'.

As acknowledged before, the PAC-Bayes literature started with the works of McAllester who coined the name ‘PAC-Bayes’ [McAllester, 1998] and proved a classical form of the PAC-Bayes bound [McAllester, 1999]. This line of work owes a lot to others: Langford and Seeger [2001], Seeger [2002] obtained a new form of the bound now called PAC-Bayes-kl; Maurer [2004] clarified that the analysis holds for any loss function with range $[0,1]$ (not only the zero-one loss) and established the optimal dependence on the sample size; Catoni [2004, 2007] clarified the form of the optimal distributions and in particular Catoni [2007] presented a new PAC-Bayes bound. We present next a general theorem from which all these PAC-Bayes bounds can be derived as special cases. This theorem follows that of Germain et al. [2009] with a minor modification which is explained in the *nota bene* after the theorem.

Theorem 2.1 (general PAC-Bayes theorem)**.** *Let the triple $(\mathcal{W}, \mathcal{Z}, \ell)$ consist of a weight space $\mathcal{W} \subset \mathbb{R}^p$, an example space $\mathcal{Z}$, and a loss function $\ell : \mathcal{W} \times \mathcal{Z} \to [0,\infty)$. Let n be a fixed positive integer, and $\mathcal{S} = \mathcal{Z}^n$. Let $\hat{L} : \mathcal{S} \times \mathcal{W} \to \mathbb{R}$ be the empirical risk functional defined as $\hat{L}(s,w) = n^{-1}\sum_{i=1}^{n} \ell(w,z_i)$ for $s = (z_1,\dots,z_n) \in \mathcal{S}$; and write $\hat{L}_s(w) = \hat{L}(s,w)$. Let $P_1 \in \mathcal{M}_1(\mathcal{Z})$ and let $L : \mathcal{W} \to [0,\infty)$ be the risk functional $L(w) = \mathbb{E}[\ell(w,Z)]$ with $Z \sim P_1$.*

For any convex function $F : \mathbb{R}^2 \to \mathbb{R}$, define $f(s,w) = F(L(w),\hat{L}_s(w))$ for $(s,w) \in \mathcal{S} \times \mathcal{W}$. For any $P_n \in \mathcal{M}_1(\mathcal{S})$ and $Q^0 \in \mathcal{M}_1(\mathcal{W})$, let ξ be the exponential moment:

$$\xi = \int_{\mathcal{S}} \int_{\mathcal{W}} e^{f(s,w)} Q^0(dw) P_n(ds)\,. \tag{2.6}$$

Then for any $\delta \in (0,1)$, with probability at least $1-\delta$ over the random draw of size-n sample $S \sim P_n$, simultaneously for all distributions Q over $\mathcal{W}$ we have

$$F(L(Q),\hat{L}_S(Q)) \leq \mathrm{KL}(Q\|Q^0) + \log(\xi/\delta)\,. \tag{2.7}$$

N.B.: The minor difference with Germain et al. [2009, Theorem 2.1] is that in our Theorem 2.1 the sample size is absorbed into the convex function F.

The proof is outlined in Section 2.3 below. This theorem is a general template for deriving PAC-Bayes bounds. Conceptually, the derivation of a PAC-Bayes bound via Theorem 2.1 consists of two components: (i) choose a function F (and a prior Q^0) to use in Eq. (2.7), and (ii) obtain an upper bound on the exponential moment ξ defined in Eq. (2.6). Notice that ξ depends on the chosen F and the chosen prior Q^0. In fact, ξ also depends on P_n but we do not stress this dependence, considering that the inequality of Eq. (2.7) is distribution-free. Also notice that in Theorem 2.1 the prior Q^0 is a data-free distribution, i.e. Q^0 has no dependence on the data S on which the empirical term $\hat{L}_S$ is evaluated.[3] The cost of the generality of Theorem 2.1 is that for each specific choice of the bound (technically, a choice of a function F and a prior Q^0) we need to study the exponential moment ξ and, in particular, provide a reasonable upper bound on it.

Similar to Germain et al. [2009], the exponential moment is presented in Eq. (2.6) with the 'un-swapped' order for the expectations, which is the default order by definition. Notice that the 'swapped' order is equivalent to the un-swapped one in the current case where the prior Q^0 is a data-free distribution:

$$\xi = \int_{\mathcal{S}} \int_{\mathcal{W}} e^{f(s,h)} Q^0(dw) P(ds) = \int_{\mathcal{W}} \int_{\mathcal{S}} e^{f(s,h)} P(ds) Q^0(dw) =: \xi_{\mathrm{swap}}\,.$$

The order has an impact on the techniques used to upper-bound the exponential moment. In general, the un-swapped order is the default (Cf. Section 2.3.1 below, and Chapter 3).

Next we discuss the derivation of some PAC-Bayes bounds from Theorem 2.1.

[3]The extension of Theorem 2.1 to data-dependent priors was considered by Rivasplata et al. [2020]. Some highlights of this extension are given in Chapter 3 below.

Assuming i.i.d. data ($P_n = P_1^{\otimes n}$) and boundedness of the loss function, namely that its range is the interval $[0,1]$, we recover the usual PAC-Bayes bounds by different choices of the convex function F and a data-free distribution Q^0 over $\mathcal{W}$. In particular, we may use the function $F(x,y) = n\,\mathrm{kl}(y\|x)$ for which Maurer [2004] showed that the exponential moment satisfies $\xi \leq 2\sqrt{n}$. This gives the **PAC-Bayes-kl** inequality, originally called the PAC-Bayes relative entropy bound [Langford and Seeger, 2001, Seeger, 2002], which concludes that for any $\delta \in (0,1)$, with probability of at least $1-\delta$ over size-n i.i.d. random samples S, simultaneously for all distributions Q over $\mathcal{W}$ it holds that

$$\mathrm{kl}(\hat{L}_S(Q)\|L(Q)) \leq \frac{\mathrm{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{n}. \tag{2.8}$$

The assumption that Q^0 is a data-free distribution over $\mathcal{W}$ means that Q^0 is fixed without any dependence on the data on which the bound is evaluated (to be very specific, Q^0 cannot depend on the sample S on which $\hat{L}_S$ is evaluated). The original form of this bound that was derived by Langford and Seeger [2001] had a slightly different dependence on n, the form presented here has the sharp dependence on n as clarified by Maurer [2004].

The PAC-Bayes-kl bound can be relaxed in various ways to obtain other PAC-Bayes bounds [see e.g. Tolstikhin and Seldin, 2013]. For instance, using the well-known version of Pinsker's inequality $\mathrm{kl}(\hat{p}\|p) \geq 2(p-\hat{p})^2$ one can lower-bound the binary KL divergence, and then solve the resulting inequality for $L(Q)$, which leads to a PAC-Bayes bound of equivalent form to that of the classic bound of McAllester [1999], hence we shall call it the **PAC-Bayes-classic** bound: for any $\delta \in (0,1)$, with probability of at least $1-\delta$ over size-n i.i.d. random samples S, simultaneously for all distributions Q over $\mathcal{W}$ we have

$$L(Q) \leq \hat{L}_S(Q) + \sqrt{\frac{\mathrm{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{2n}}. \tag{2.9}$$

Notice that the PAC-Bayes-classic bound is an upper bound on $L(Q)$ that holds simultaneously for all distributions Q over weights, with high probability (over samples). In particular, the upper bound may be optimised to choose a distribution Q in a data-dependent manner. This is the idea behind the generic theme that I call PAC-Bayesian Computation (PBC). Repeated use of the bound during optimisation over Q is legitimate, without affecting the form of the bound, because the bound holds uniformly for all distributions Q.

An alternative way to relax the PAC-Bayes-kl bound is using the refined version of Pinsker's inequality $\mathrm{kl}(\hat{p}\|p) \geq (p-\hat{p})^2/(2p)$ valid for $\hat{p} < p$ [see e.g. Boucheron et al., 2013, Lemma 8.4], which is tighter than the former version when $p < 1/4$, and this refined inequality gives

$$L(Q) \leq \hat{L}_S(Q) + \sqrt{2L(Q)\frac{\mathrm{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{n}}\,. \qquad (\star)$$

The difference to the result one gets from the well-known version of Pinsker's inequality is the appearance of $L(Q)$ under the square root on the right hand side. This, in particular, tells us that the inequality is tighter than Eq. (2.9) when the population loss, $L(Q)$, is smaller (specifically when $L(Q) < 1/4$). But it is exactly because of the appearance of $L(Q)$ on the right-hand side that this bound is not immediately useful for optimisation purposes. However, one can view the above inequality as a quadratic inequality on $\sqrt{L(Q)}$. Solving this inequality for $L(Q)$ leads to the **PAC-Bayes-quadratic** bound [Rivasplata et al., 2019, Pérez-Ortiz et al., 2021b]: for any $\delta \in (0,1)$, with probability of at least $1-\delta$ over size-n i.i.d. random samples S, simultaneously for all distributions Q over $\mathcal{W}$ we have

$$L(Q) \leq \left(\sqrt{\hat{L}_S(Q) + \frac{\mathrm{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{2n}} + \sqrt{\frac{\mathrm{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{2n}}\right)^2 . \tag{2.10}$$

Similarly to the PAC-Bayes-classic bound, the PAC-Bayes-quadratic bound is an inequality that holds uniformly over all Q; hence the upper bound may be optimised with respect to Q in order to obtain a data-dependent distribution over weights.

Alternatively, one could combine $(\star)$ with the inequality $\sqrt{ab} \leq \frac{1}{2}(\lambda a + \frac{b}{\lambda})$ which is valid for all $\lambda > 0$. Then substituting $a = L(Q)$ and restricting to $\lambda \in (0,2)$, after some rearrangement one obtains the **PAC-Bayes-λ** bound of Thiemann et al. [2017]: for any $\delta \in (0,1)$, with probability of at least $1-\delta$ over size-n i.i.d. random samples S, simultaneously for all distributions Q over $\mathcal{W}$ and $\lambda \in (0,2)$ we have

$$L(Q) \leq \frac{\hat{L}_S(Q)}{1-\lambda/2} + \frac{\mathrm{KL}(Q\|Q^0) + \log(2\sqrt{n}/\delta)}{n\lambda(1-\lambda/2)}\,. \tag{2.11}$$

Notice that λ is independent from the function $F(x,y) = n\,\mathrm{kl}(y\|x)$ chosen to derive the PAC-Bayes-kl bound. Then the fact that Eq. (2.11) holds for all $\lambda \in (0,2)$ is explained by the *post hoc* use of the inequality $\sqrt{ab} \leq \frac{1}{2}(\lambda a + \frac{b}{\lambda})$ which is valid for all $\lambda > 0$, while the restriction $\lambda < 2$ is to preserve of the inequality sign.

An interesting feature of the PAC-Bayes-λ bound is that this bound holds uniformly over all Q and $\lambda \in (0,2)$, hence in principle this bound is optimisable over both these quantities. The work of Thiemann et al. [2017] discussed conditions under which the PAC-Bayes-λ bound can be optimised alternatingly over Q and λ. Since this bound holds uniformly over $\lambda \in (0,2)$, it is possible to search a grid of λ-values without worsening the bound. If the bound was not uniform over λ but instead it was valid for a fixed λ, then for a search over a number K of λ-values the bound would worsen by an additive term of $\log(K)$ in the numerator of the second term.

Next we discuss some bounds that, unlike the previous discussed bounds, are not relaxations of the PAC-Bayes-kl bound.

An important contribution to the PAC-Bayesian literature was made by Catoni. In particular, Catoni [2007] derived a new PAC-Bayes bound: for any $\delta \in (0,1)$ and any given $\beta > 0$, with probability of at least $1-\delta$ over size-n i.i.d. random samples S, simultaneously for all distributions Q over $\mathcal{W}$ we have

$$L(Q) \leq \frac{1}{1-e^{-\beta}}\left[1-\exp\left\{-\left(\beta\hat{L}_S(Q)+\frac{\mathrm{KL}(Q\|Q^0)+\log(1/\delta)}{n}\right)\right\}\right]. \tag{2.12}$$

This inequality can be derived as a consequence of Theorem 2.1 by using the convex function $F(x,y) = n\log\left(\frac{1}{1-x(1-e^{-\beta})}\right) - \beta ny$ with a fixed $\beta > 0$, for which Catoni [2007] showed that the exponential moment term satisfies $\xi \leq 1$. An important fact to keep in mind is that the high-probability inequality of Eq. (2.12) holds for a fixed $\beta > 0$ only, not uniformly, which is explained by the fact that the chosen function F depends on a fixed value of β. This implies that the bound cannot be optimised over β for free.

McAllester [2013] showed that it is possible to obtain a 'linearised' form of Catoni's bound while restricting the free parameter in the bound: for any $\delta \in (0,1)$ and any $\beta > 1/2$, with probability of at least $1-\delta$ over size-n i.i.d. random samples S, simultaneously for all distributions Q over $\mathcal{W}$ we have

$$L(Q) \leq \frac{1}{1-\frac{1}{2\beta}}\left[\hat{L}_S(Q)+\beta\frac{\mathrm{KL}(Q\|Q^0)+\log(1/\delta)}{n}\right]. \tag{2.13}$$

This bound was recently studied by Dziugaite et al. [2021] in the context of studying the role of data in PAC-Bayes bounds, in particular in learning the PAC-Bayes prior from data.

Unlike the PAC-Bayes-λ bound which holds uniformly over all values of its parameter λ in its domain, the bound of Catoni [2007] in Eq. (2.12), and likewise that of McAllester [2013] in Eq. (2.13), holds for a fixed β in its corresponding domain. Because of this fact, a search over several β-values necessarily worsens the bound. In particular, by a simple use of the union bound, a grid search over say K possible values for β would worsen the bound by an additive $\log(K)$.

There are many application areas that have used PAC-Bayes bounds, but there are essentially two ways that a PAC-Bayes bound is typically applied: either (1) use the bound to give a risk certificate for a predictor based on a distribution over hypotheses learned by some method, or (2) turn the bound into a learning method by searching a distribution that minimises the bound. The next two sections discuss further these two uses of PAC-Bayes bounds. As discussed earlier, these two use cases are the central themes of this thesis. The first use case, which can be called *risk certification*, is discussed further in Section 2.2.1. The second use case, which I propose to call *PAC-Bayesian computation (PBC)*, is discussed further in Section 2.2.2. Additional discussion and some related literature are covered in Section 2.2.3

2.2.1 Risk certification via PAC-Bayes bounds

PAC-Bayes bounds, as well as other kinds of generalisation bounds, have been used for certifying the risk of various prediction rules. Since the PAC-Bayes bounds involve distributions over a given hypothesis class, these kinds of bounds are particularly suited for prediction rules defined by a distribution over the hypothesis class. This includes the randomised prediction rule and the weighted majority vote rule, which are two important cases. The risk certificates are obtained evaluating numerical values of the bounds. Then, a desirable feature is for these numerical values to be tight, so that the numerical value of the risk certificate can be considered to be informative of the (unobservable) value of the population risk.

In this context, the interest in PAC-Bayes bounds is their ability to deliver tight certificates, which has been observed across a variety of learning problems with various learning models, such as linear classifiers [Germain et al., 2009], support vector machines [Ambroladze et al., 2007, Parrado-Hernández et al., 2012], decisions trees [Lorenzen et al., 2019, Masegosa et al., 2020], among others. Recently, the ability of PAC-Bayes bounds to give non-vacuous numerical bound values for neural network models was reported by Dziugaite and Roy [2017], in the regime where the model has many more parameters than training data; while their ability to give tight risk certificates for deep learning models was shown by Pérez-Ortiz et al. [2021b] who also pointed out that learning the PAC-Bayes prior is crucial for achieving tight values. The questions of tightness of risk certificates for neural networks and the role of data in learning the PAC-Bayes prior have been considered by Dziugaite et al. [2021].

2.2.2 PAC-Bayesian computation

As mentioned earlier, I propose to call PAC-Bayesian Computation (PBC) the generic approach to learning a data-dependent distribution over hypotheses by minimising a PAC-Bayes bound. At a high level, using a generalisation bound as a learning objective is a strategy that makes sense considering that the minimiser obtained by this approach will be a predictor whose risk is guaranteed to be bounded by the minimal value of the learning objective. This approach has been explored in the literature with learning objectives based on generalisation bounds that hold uniformly over a hypothesis class [Freund, 1998, Langford and Blum, 2003].

The approach of using a PAC-Bayes bound as learning objective was mentioned already by McAllester [1999], credit for this approach in various contexts is due also to Germain et al. [2009], Seldin and Tishby [2010], Keshet et al. [2011], Noy and

Crammer [2014], Keshet et al. [2017], possibility among others. Recently, the use of this approach has also found success in training neural networks, see Dziugaite and Roy [2017]. In fact, the recent resurgence of interest in PAC-Bayes bounds has been to a large extent motivated by the interest in generalisation guarantees for neural networks. Langford and Caruana [2001] used McAllester [1999]'s classical PAC-Bayesian bound to evaluate the error of a (stochastic) neural network classifier. Dziugaite and Roy [2017] obtained numerically non-vacuous generalisation bounds by optimising the same bound. Subsequent studies (e.g. Rivasplata et al. [2019], Pérez-Ortiz et al. [2021b]) continued this approach, sometimes with links to the generalisation of stochastic optimisation methods (e.g. London [2017], Neyshabur et al. [2018], Dziugaite and Roy [2018b]) or algorithmic stability.

The data-dependent distributions obtained by optimising a PAC-Bayes bound are usually called 'posterior' distributions in the PAC-Bayesian literature. However, these PAC-Bayes posterior distributions should not be confused with Bayesian posteriors. In the frequentist PAC-Bayes analysis, what is called 'prior' is a reference distribution and what is called 'posterior' is an unrestricted distribution, in the sense that there is no likelihood-type factor connecting these two distributions. When learning 'posteriors' via PBC, asking whether learned distribution approximates the true posterior is meaningless, as the notion of 'true PAC-Bayes posterior' is nonexistent. The prior is used by a learner only via the KL term in the bound. For this reason, PAC-Bayesian computation affords an extra level of flexibility in the choice of distributions, even compared to generalised Bayesian approaches [Bissiri et al., 2016].

Having said that, in practice the PAC-Bayes bounds are optimised over a restricted class of distributions. The restriction to a specific class of distributions (e.g. Gaussian) has desirable effects such as explicit analytic expressions for the KL term in the bound, and computational tractability. Gaussian and Laplace distributions are two typical choices of classes of distributions over which to optimise a PAC-Bayes bound [Noy and Crammer, 2014, Blundell et al., 2015, Dziugaite and Roy, 2017, Pérez-Ortiz et al., 2021b]. Arguably, the posterior Gibbs distributions [Dziugaite and Roy, 2018a] are a sensible choice to explore, considering that these distributions are optimal for the PAC-Bayes bounds [Catoni, 2007, Alquier et al., 2016, Guedj, 2019].

2.2.3 Additional discussion and related literature

The literature on PAC-Bayes bounds and their applications is vast. We briefly mention the usual references McAllester [1999], Langford and Seeger [2001], Seeger [2002], and Catoni [2007]; but see also Maurer [2004], and Keshet et al. [2011]. Note that McAllester [1998, 1999] proved the first PAC-Bayesian theorems, and in fact coined the name "PAC-Bayes" arguably taking inspiration from the work of Shawe-Taylor and Williamson [1997] who carried out a PAC analysis of a Bayesian-style estimator. We acknowledge the tutorials of Langford [2005] and McAllester [2013], the mini-tutorial of van Erven [2014], and the primer of Guedj [2019].

The general PAC-Bayes theorem presented in Theorem 2.1 is essentially equivalent to the one of Germain et al. [2009], which was later re-used by Bégin et al. [2014, 2016]. A similar general theorem was given before by Audibert [2004]. The general PAC-Bayes theorem presented in Theorem 2.1 makes it clear that there are essentially two steps in deriving a PAC-Bayes bound: (i) Choose a function F and a prior Q^0 to use in Theorem 2.1, and (ii) find an upper bound for the corresponding exponential moment ξ. The latter step, controlling the exponential moment, usually takes a considerable part of the analysis, and typically this step informs the choices of F that are viable (i.e. those for which bounds on ξ can be obtained).

Recently Rivasplata et al. [2020] reformulated the PAC-Bayes analysis in terms of stochastic kernels, from which they derived a novel PAC-Bayes bound with a data-dependent Gibbs prior. This work built on that of Dziugaite and Roy [2018a]. Another salient contribution of Rivasplata et al. [2020] was revisiting the usual assumptions on which PAC-Bayes bounds are derived, namely (a) data-free priors, (b) bounded losses, and (c) i.i.d. data; and discussing ways to relax these assumptions. (Rivasplata et al. [2020] gave examples relaxing (a) and (b).)

A line of work related to connecting PAC-Bayes priors to data was explored by Catoni [2007], Lever et al. [2013], Pentina and Lampert [2014] and more recently by Rivasplata et al. [2018], who assumed that priors are *distribution-dependent*. In that setting the priors are still ‘data-free’ but in a less agnostic fashion (compared to an arbitrary fixed prior), which allows to demonstrate improvements for “nice” data-generating distributions. Data-dependent priors were investigated recently by Awasthi et al. [2020], who relied on tools from the empirical process theory and controlled the capacity of a data-dependent hypothesis class (see also Foster et al. [2019]).

The PAC-Bayes literature does contain a line of work that investigates relaxing the restriction of bounded loss functions. A straightforward way to extend PAC-Bayes inequalities to unbounded loss functions is to make assumptions on the tail behaviour of the loss [Alquier et al., 2016, Germain et al., 2016a] or its moments [Alquier and Guedj, 2018, Holland, 2019], leading to interesting bounds in special cases. Recent work has also looked into the analysis for heavy-tailed losses. For example, Alquier and Guedj [2018] proposed a polynomial moment-dependent bound with f-divergence replacing the KL divergence, while Holland [2019] devised an exponential bound assuming that the second moment of the loss is bounded uniformly across hypotheses. An alternative approach was explored by Kuzborskij and Szepesvári [2019], who proposed a stability-based approach by controlling the Efron-Stein variance proxy of the loss. Least squares regression (squared loss) was studied by Shalaeva et al. [2020] who improved results of Germain et al. [2016a] and also relaxed the data-generation assumption to non-iid data. It is worth mentioning other works related to extending PAC-Bayes bounds to statistically dependent data, see e.g. Alquier and Wintenberger [2012] who applied Rio [2000]'s version of Hoeffding's inequality, then derived PAC-Bayes bounds for non-i.i.d. data, and used them in model selection for time series.

As we mentioned earlier, besides randomised predictions, other prediction schemes may be derived from a distribution over hypotheses. Aggregation by exponential weighting was considered by Dalalyan and Tsybakov [2007, 2008], weighted majority vote for ensembles of decision trees were considered by Lorenzen et al. [2019], Masegosa et al. [2020] and others, and the weighted majority vote in other settings has been studied by Lacasse et al. [2006], Germain et al. [2015] and others. This list is far from being complete.

Finally, it is worth mentioning that the PAC-Bayesian analysis extends beyond bounds on the gap between population and empirical losses: A large body of literature has also looked into upper and lower bounds on the *excess risk*, namely, $Q_S[L] - \inf_{h \in \mathcal{H}} L(h)$, we refer e.g. to Catoni [2007], Alquier et al. [2016], Grünwald and Mehta [2019], Kuzborskij et al. [2019], Mhammedi et al. [2019]. The approach of analyzing the gap (for randomised predictors), which we follow in this work, is generally complementary to such excess risk analyses.

N.B.: The discussion of the related literature just presented is borrowed from the corresponding section of my paper Rivasplata et al. [2020].

2.3 Mathematical technicalities

Different communities use different notations to represent expectations, which at the very bottom are integrals with respect to probability measures. In mathematics and probability theory, the notations $\rho[f]$ and $\langle f, \rho \rangle$ are often used as shorthand for the integral $\int_{\mathcal{W}} f(w)\rho(dw)$ whenever ρ is a measure over $\mathcal{W}$ and the function $f : \mathcal{W} \to \mathbb{R}$ is integrable. Then if ρ is a probability, these notations are alternative ways to write the expectation $\mathbb{E}[f(W)]$ with respect to the random $W \sim \rho$. In the PAC-Bayesian literature, and more generally in the machine learning literature, it is common to write the latter in the form $\mathbb{E}_{W \sim \rho}[f(W)]$ arguably to show explicitly the random variable and the distribution with respect to which the expectation is taken.

The population loss $L(Q)$ and the empirical loss $\hat{L}_S(Q)$, as defined in Eq. (2.3) and Eq. (2.4), are written alternatively as $Q[L]$ and $Q[\hat{L}_S]$, respectively, using the notation just described above. The latter was the notation used by Rivasplata et al. [2020] as it is convenient for the proofs, by compressing the notation; while the former is convenient for presenting the PAC-Bayes bounds in view of its intuitive and readable nature, as discussed in Section 2.2.

In this thesis we follow the convention that random quantities are denoted with capitals, while the possible values they may take are denoted with their lower case counterparts, and their corresponding spaces (of all possible values) with calligraphic.

Thus, for instance, we use W for the random weight vector; we use w for an arbitrary realisation (possible value) of this weight vector, and we use $\mathcal{W}$ for the space of all possible weight vectors. Similarly, the tripples $Z,z,\mathcal{Z}$ and $X,x,\mathcal{X}$ and $Y,y,\mathcal{Y}$ follow this convention, as does the triple $S,s,\mathcal{S}$ related to samples.

When $f:\mathcal{S}\times\mathcal{W}\to\mathbb{R}$ is a sample- and weight-dependent function, we use the shortcut $f_s(w)=f(s,w)$; then we may regard f_s as a weight-dependent function for a given $s\in\mathcal{S}$. Note that if ρ_S is a data-dependent distribution, where S is random data, and $f:\mathcal{S}\times\mathcal{W}\to\mathbb{R}$ is integrable, then $\rho_S[f_S]=\int_{\mathcal{W}} f_S(w)\rho_S(dw)$ should be interpreted as the conditional expectation $\mathbb{E}[f(S,W)|S]$. As a general rule, conditional expectation will be made explicit as just indicated, while when using the expectation operator $\mathbb{E}[\cdot]$ we assume that it integrates over all the enclosed random variables. Similar comments for the use of the probability operator $\mathbb{P}[\cdot]$.

2.3.1 Proof of the general PAC-Bayes theorem

We start with a technical lemma that recalls a well-known inequality which can be traced back to Csiszár [1975] and Donsker and Varadhan [1975].

Lemma 2.2 (Change of measure inequality)**.** *For any measurable function $f:\mathcal{W}\to\mathbb{R}$ and any distributions Q,Q^0 over $\mathcal{W}$, the following inequality holds:*

$$Q[f]\leq \mathrm{KL}(Q\|Q^0)+\log Q^0[e^f]\,.$$

In particular, for any measurable function $f:\mathcal{S}\times\mathcal{W}\to\mathbb{R}$ and any distributions Q,Q^0 over $\mathcal{W}$, and for any given $s\in\mathcal{S}$, we have $Q[f_s]\leq \mathrm{KL}(Q\|Q^0)+\log Q^0[e^{f_s}]$.

Proof. Let $f:\mathcal{W}\to\mathbb{R}$ be measurable, and let Q,Q^0 be any distributions over $\mathcal{W}$. By a simple change of measure, and Jensen's inequality, we have:

$$\begin{aligned}\log\int_{\mathcal{W}}\exp(f)\,dQ^0 &= \log\int_{\mathcal{W}}\exp(f)\frac{dQ^0}{dQ}\,dQ\\ &\geq \int_{\mathcal{W}}\log\Big(\exp(f)\frac{dQ^0}{dQ}\Big)\,dQ\\ &= \int_{\mathcal{W}} f\,dQ+\int_{\mathcal{W}}\log\Big(\frac{dQ^0}{dQ}\Big)\,dQ\,.\end{aligned}$$

Rewriting in terms of the notation introduced before, the inequality between the first and last expressions is $\log Q^0[e^f]\geq Q[f]-\mathrm{KL}(Q\|Q^0)$, which is equivalent to the inequality of the lemma.

The proof for $f:\mathcal{S}\times\mathcal{W}\to\mathbb{R}$ is similar. In particular, for any given $s\in\mathcal{S}$ we have:

$$\log\int_{\mathcal{W}}\exp(f_s)\,dQ^0\geq\int_{\mathcal{W}} f_s\,dQ+\int_{\mathcal{W}}\log\Big(\frac{dQ^0}{dQ}\Big)\,dQ\,,$$

which translates into $\log Q^0[e^{f_s}]\geq Q[f_s]-\mathrm{KL}(Q\|Q^0)$. □

Next we present the proof of Theorem 2.1. The theorem was stated for $\mathcal{S} = \mathcal{Z}^n$ the space of size-n samples, and a distribution $P_n \in \mathcal{M}_1(\mathcal{S})$ that generates the size-n random samples. For simplicity (and generality) in the proof given below we consider an abstract measurable space $\mathcal{S}$ and a distribution $P \in \mathcal{M}_1(\mathcal{S})$.

Proof of Theorem 2.1. Let $F : \mathbb{R}^2 \to \mathbb{R}$ be a convex function, and let $f : \mathcal{S} \times \mathcal{W} \to \mathbb{R}$ be the function defined by $f(s,w) = F(L(w), \hat{L}_s(w))$ for $(s,w) \in \mathcal{S} \times \mathcal{W}$.

Fix a distribution Q^0 over $\mathcal{W}$, and consider the function $\phi : \mathcal{S} \to [0,\infty]$ defined by

$$\phi(s) = \sup_Q \left\{ \int_{\mathcal{W}} f_s \, dQ - \mathrm{KL}(Q\|Q^0) \right\}$$

where the supremum is over all distributions Q over $\mathcal{W}$.

Let $P \in \mathcal{M}_1(\mathcal{S})$. By Lemma 2.2,

$$\mathbb{E}[e^{\phi(S)}] = \int_{\mathcal{S}} e^{\phi} \, dP \leq \int_{\mathcal{S}} \int_{\mathcal{W}} e^{f(s,w)} \, Q^0(dw) P(ds) =: \xi \, .$$

Then, applying Markov's inequality, for any $\delta \in (0,1)$ we have:

$$\mathbb{P}[\phi(S) > \log(\xi/\delta)] = \mathbb{P}[e^{\phi(S)} > \xi/\delta] \leq \frac{\mathbb{E}[e^{\phi(S)}]}{\xi/\delta} \leq \frac{\xi}{\xi/\delta} = \delta \, .$$

Therefore, with probability of at least $1-\delta$ over the random draw of $S \sim P$ we have the inequality $\phi(S) \leq \log(\xi/\delta)$, i.e. simultaneously for all Q we have

$$\int_{\mathcal{W}} f_S(w) \, Q(dw) \leq \mathrm{KL}(Q\|Q^0) + \log(\xi/\delta) \, .$$

Finally, by the convexity of F and Jensen's inequality, we have

$$\begin{aligned}
\int_{\mathcal{W}} f_S(w) \, Q(dw) &= \int_{\mathcal{W}} F(L(w), \hat{L}_S(w)) \, Q(dw) \\
&\geq F\Big(\int_{\mathcal{W}} L(w) \, Q(dw), \int_{\mathcal{W}} \hat{L}_S(w) \, Q(dw) \Big) = F(L(Q), \hat{L}_S(Q)) \, .
\end{aligned}$$

This completes the proof of Theorem 2.1 upon replacing P with P_n. □

Notice, in particular, the part of the proof that applies Markov's inequality to the random variable $\phi(S)$. Since the function ϕ is defined by taking a supremum over Q, this explains why the high-probability inequality of Theorem 2.1, namely Eq. (2.7), holds *uniformly* over all distributions Q over $\mathcal{W}$, which was expressed equivalently by the phrase "simultaneously for all distributions Q over $\mathcal{W}$" in the conclusion of Theorem 2.1 as well as the statement of PAC-Bayes bounds derived from it.

Chapter 3

PAC-Bayes analysis with stochastic kernels

N.B.: The content of this chapter comes from my paper Rivasplata et al. [2020]. Minor updates were made to improve the clarity.

Chapter Layout. This chapter is organised as follows. Section 3.1 discusses the definition of stochastic kernels and proposes to use it to formalise data-dependent distributions over weights. Section 3.2 presents the main results of this work, namely, the basic inequalities for stochastic kernels of Theorem 3.1 and the general PAC-Bayes theorem for stochastic kernels of Theorem 3.2. Then, Section 3.3 gives a discussion of the implications of these results, including how from Theorem 3.1 one may derive various PAC-Bayes style bounds that have similar forms to the well-known PAC-Bayes bounds in the literature, and also new kinds of PAC-Bayes bounds. Finally, Section 3.4 concludes the chapter and discusses future work.

3.1 Data-dependent distributions as stochastic kernels

The formal definition presented in this section requires explicit reference to the sigma algebras on the spaces of interest (sample space and weight space). Accordingly, the space of size-n samples $\mathcal{S}$ is equipped with a sigma algebra[1] that we denote $\Sigma_{\mathcal{S}}$, and the weight space $\mathcal{W}$ is equipped with a sigma algebra that we denote $\Sigma_{\mathcal{W}}$.

A data-dependent (or sample-dependent) distribution over $\mathcal{W}$ is formalised as a *stochastic kernel* from $\mathcal{S}$ to $\mathcal{W}$, which is defined as a mapping $Q : \mathcal{S} \times \Sigma_{\mathcal{W}} \to [0,1]$ such that (i) for each $B \in \Sigma_{\mathcal{W}}$ the function $s \mapsto Q(s,B)$ is measurable; and (ii) for each $s \in \mathcal{S}$ the function $Q_s : B \mapsto Q(s,B)$ is a probability measure over $\mathcal{W}$. We write $\mathcal{K}(\mathcal{S},\mathcal{W})$ to denote the set of all stochastic kernels from $\mathcal{S}$ to —distributions over— $\mathcal{W}$, and we reserve the notation $\mathcal{M}_1(\mathcal{W})$ for the set of 'data-free' distributions over $\mathcal{W}$. Notice that $\mathcal{M}_1(\mathcal{W}) \subset \mathcal{K}(\mathcal{S},\mathcal{W})$, since every 'data-free' distribution can be regarded as a constant kernel.

[1]Recall that a sigma-algebra on a space $\mathcal{X}$ is a collection of subsets of $\mathcal{X}$ that contains the set $\mathcal{X}$ itself, and is stable under complements and countable unions and intersections.

In the definition of stochastic kernel, the condition (ii) says that for each fixed sample s one has an associated distribution over $\mathcal{W}$, and in fact this condition defines the notation Q_s for the distribution associated to a given sample s under the kernel Q. On the other hand, condition (i) is mainly technical and its role is ensuring measurability of the mapping that takes in samples and outputs distributions.

Stochastic kernels are well-known in the literature on stochastic processes, where they are also called *transition kernels* or *probability kernels*; see for instance Kallenberg [2017], Meyn and Tweedie [2009] and Ethier and Kurtz [1986]. The name 'kernel' is arguably justified by the fact that these objects are the basic building blocks for constructing the finite-dimensional distributions of some kinds of stochastic processes. Stochastic kernels have appeared in the learning theory literature under the names of regular conditional probabilities [Catoni, 2004, 2007, Alquier, 2008] or Markov kernels [Xu and Raginsky, 2017].

To have a tangible example of stochastic kernel, we may consider a Gaussian kernel which takes in an arbitrary sample $s \in \mathcal{S}$ and produces a Gaussian distribution over $\mathcal{W} = \mathbb{R}^p$ with sample-dependent mean $m(s) \in \mathbb{R}^p$ and sample-dependent covariance $\Sigma(s) \in \mathbb{R}^{p\times p}$. This kernel is the function $Q : \mathcal{S} \times \Sigma_{\mathcal{W}} \to [0,1]$ with formula

$$Q(s,B) = C \int_B \exp\big\{ - \|w - m(s)\|^2_{\Sigma(s)^{-1}} \big\} \mu(dw)\,,$$

where $C > 0$ is the normalisation factor, μ is the Lebesgue measure on $\mathbb{R}^p$, and we used the notation $\|w\|^2_M = w^\top M w$ so that $\|\cdot\|_M$ is the norm induced by the matrix $M \in \mathbb{R}^{p\times p}$ when this matrix is positive definite. Then, conditions (i) and (ii) are met. Furthermore, following the notation convention declared in condition (ii), for any given sample s we then denote by Q_s the probability measure (over $\mathcal{W}$) defined by the right-hand side of the preceding equation; this measure acts on sets $B \in \Sigma_{\mathcal{W}}$ as follows:

$$Q_s(B) = C \int_B \exp\big\{ - \|w - m(s)\|^2_{\Sigma(s)^{-1}} \big\} \mu(dw)\,.$$

Moreover, given a *random* sample S drawn from some distribution over $\mathcal{S}$, we then have an associated random measure Q_S over $\mathcal{W}$.

In general, if $Q \in \mathcal{K}(\mathcal{S},\mathcal{W})$ is a stochastic kernel from $\mathcal{S}$ to $\mathcal{W}$, then the notation Q_s stands for the distribution over $\mathcal{W}$ corresponding to an arbitrary sample s, and accordingly the notation Q_S stands for the distribution over $\mathcal{W}$ corresponding to a randomly drawn sample S. One well-known example of data-dependent distributions, which will be used in Chapter 4 and Chapter 5, is that of Gaussian distributions with a data-dependent mean vector and a possibly data-dependent covariance matrix, such as described in the previous paragraph. Another important example is the data-dependent *Gibbs* distribution, where Q_S is of the form $Q_S(dw) \propto e^{-\gamma \hat{L}_S(w)} \mu(dw)$ for some $\gamma > 0$, with μ a base measure over $\mathcal{W}$.

In this work we present a basic inequality for stochastic kernels (Theorem 3.1 below) and a general PAC-Bayes style theorem for stochastic kernels (Theorem 3.2 below). These results are applicable to data-dependent distributions over a hypothesis space, which can be represented as stochastic kernels. To make a case for the usefulness of this approach, we show that Theorem 3.2 encompasses PAC-Bayes style bounds of similar forms to the usual PAC-Bayes bounds in the literature [McAllester, 1998, 1999, Seeger, 2002, Catoni, 2007, Thiemann et al., 2017], while at the same time Theorem 3.2 enables new PAC-Bayes bounds. Importantly, this study takes a critical stand on the "usual assumptions" on which PAC-Bayes inequalities are based, namely, (a) data-free prior, (b) bounded loss, and (c) i.i.d. data observations. We aim to clarify the role of these assumptions and to illustrate how to obtain PAC-Bayes inequalities in cases where these assumptions are removed. As we will soon see, the analysis leading to our Theorem 3.2 shows that the PAC-Bayes priors can be data-dependent by default, and also that the underlying loss function can be unbounded by default. Furthermore, the proof of our Theorem 3.2 does not rely on the assumption of i.i.d. data observations, which may enable new results for statistically dependent data in future research.

3.2 Main results: Two theorems for stochastic kernels

The following results involve data- and weight-dependent functions $f : \mathcal{S} \times \mathcal{W} \to \mathbb{R}$. Notice that the order $\mathcal{S} \times \mathcal{W}$ is immaterial—functions $\mathcal{W} \times \mathcal{S} \to \mathbb{R}$ are treated the same way. It will be convenient to define $f_s(w) = f(s,w)$.

If $\rho \in \mathcal{M}_1(\mathcal{W})$ is a 'data-free' distribution, we will write $\rho[f_s]$ to denote the ρ-weighted average of $f_s(\cdot)$ for fixed s, that is, $\rho[f_s] = \int_{\mathcal{W}} f_s(w)\rho(dw)$. When ρ is data-dependent, that is, $\rho \in \mathcal{K}(\mathcal{S},\mathcal{W})$ is a stochastic kernel, we will write ρ_s for the distribution over $\mathcal{W}$ corresponding to a fixed s, so $\rho_s(B) = \rho(s,B)$ for $B \in \Sigma_{\mathcal{W}}$, and $\rho_s[f_s] = \int_{\mathcal{W}} f_s(h)\rho_s(dw)$.

The joint distribution over $\mathcal{S} \times \mathcal{W}$ defined by $P \in \mathcal{M}_1(\mathcal{S})$ and $Q \in \mathcal{K}(\mathcal{S},\mathcal{W})$ is the measure denoted by $P \otimes Q$ that acts on functions $\phi : \mathcal{S} \times \mathcal{W} \to \mathbb{R}$ as follows:

$$(P \otimes Q)[\phi] = \int_{\mathcal{S}} P(ds) \int_{\mathcal{W}} Q(s,dw)[\phi(s,w)] = \int_{\mathcal{S}} \int_{\mathcal{W}} \phi(s,w) Q_s(dw) P(ds) \,.$$

The notation $P \otimes Q$ is standard for this construction, see e.g. Kallenberg [2017]. Drawing a random pair $(S,W) \sim P \otimes Q$ is equivalent to drawing $S \sim P$ and drawing $W \sim Q_S$. In this case, with $\mathbb{E}$ denoting the expectation under the joint distribution $P \otimes Q$, the previous display takes the form $\mathbb{E}[\phi(S,W)] = \mathbb{E}[\mathbb{E}[\phi(S,W)|S]]$.

The joint distribution over $\mathcal{S} \times \mathcal{W}$ defined by $P \in \mathcal{M}_1(\mathcal{S})$ and $Q \in \mathcal{K}(\mathcal{S},\mathcal{W})$, as just constructed above and denoted $P \otimes Q$, corresponds to what in Bayesian learning is commonly written $Q_{W|S}P_S$.

The first main result is the following theorem.

Theorem 3.1 (Inequalities for stochastic kernels)**.** *Fix a probability $P \in \mathcal{M}_1(\mathcal{S})$, a stochastic kernel $Q^0 \in \mathcal{K}(\mathcal{S},\mathcal{W})$, and a measurable function $f : \mathcal{S} \times \mathcal{W} \to \mathbb{R}$, and let*

$$\xi = \int_{\mathcal{S}} \int_{\mathcal{W}} e^{f(s,w)} Q^0_s(dw) P(ds)\,.$$

(i) For any stochastic kernel $Q \in \mathcal{K}(\mathcal{S},\mathcal{W})$, for any $\delta \in (0,1)$, with probability of at least $1-\delta$ over the random draw of a pair $(S,W) \sim P \otimes Q$ we have

$$f(S,W) \leq \log\left(\frac{dQ_S}{dQ^0_S}(W)\right) + \log(\xi/\delta)\,.$$

(ii) For any stochastic kernel $Q \in \mathcal{K}(\mathcal{S},\mathcal{W})$, for any $\delta \in (0,1)$, with probability of at least $1-\delta$ over the random draw of $S \sim P$ we have

$$Q_S[f_S] \leq \mathrm{KL}(Q_S \| Q^0_S) + \log(\xi/\delta)\,.$$

Proof. Let $Q^0 \in \mathcal{K}(\mathcal{S},\mathcal{W})$, and let $\mathbb{E}^0$ denote expectation under the joint distribution $P \otimes Q^0$. Thus if $S \sim P$ and $W \sim Q^0_S$ we then have $\xi = \mathbb{E}^0[\mathbb{E}^0[e^{f(S,W)}|S]]$.

Let $Q \in \mathcal{K}(\mathcal{S},\mathcal{W})$ and let us denote by $\mathbb{E}$ the expectation with respect to the joint distribution $P \otimes Q$. Then, by a change of measure, we may re-write $\xi = \mathbb{E}^0[e^{f(S,W)}]$ as $\xi = \mathbb{E}[e^{\tilde{f}(S,W)}] = \mathbb{E}[e^D]$ with

$$D = \tilde{f}(S,W) = f(S,W) - \log\left(\frac{dQ_S}{dQ^0_S}(W)\right)\,.$$

Recall that when Y is a positive random variable, by Markov's inequality, for any $\delta \in (0,1)$, with probability of at least $1-\delta$ we have: $\log Y \leq \log \mathbb{E}[Y] + \log(1/\delta)$.

(i) Applying the previous inequality to $Y = e^D$, with probability of at least $1-\delta$ over the random draw of the pair $(S,W) \sim P \otimes Q$ we get $D \leq \log \mathbb{E}[e^D] + \log(1/\delta)$.

(ii) Recall $f_S(W) = f(S,W)$. Notice that the conditional expectation given S satisfies $\mathbb{E}[D|S] = Q_S[f_S] - \mathrm{KL}(Q_S \| Q^0_S)$. By Jensen's inequality, $\mathbb{E}[D|S] \leq \log \mathbb{E}[e^D|S]$, while from the previous inequality applied to $Y = \mathbb{E}[e^D|S]$, with probability of at least $1-\delta$ over the random draw of $S \sim P$ we have $\log \mathbb{E}[e^D|S] \leq \log \mathbb{E}[e^D] + \log(1/\delta)$. $\square$

Notice that in Theorem 3.1 the 'prior' Q^0 is a stochastic kernel from $\mathcal{S}$ to $\mathcal{W}$. Hence, given a data set S, the corresponding Q^0_S is by default a data-dependent distribution over weights. Also note that the function f is unrestricted, and the distribution $P \in \mathcal{M}_1(\mathcal{S})$ is unrestricted, except for integrability conditions to ensure that the exponential moment ξ is finite.

Suppose the function f is a composition of the form $f = F \circ A$ with $A : \mathcal{S} \times \mathcal{W} \to \mathbb{R}^2$ given by $A(s,w) = (L(w), \hat{L}_s(w))$ and $F : \mathbb{R}^2 \to \mathbb{R}$ a convex function. In this case, by Jensen's inequality we have $F(Q_s[A_s]) \leq Q_s[F(A_s)]$ and Theorem 3.1(ii) then gives our second main result:

Theorem 3.2 (general PAC-Bayes for stochastic kernels)**.** *Let the triple $(\mathcal{W},\mathcal{Z},\ell)$ consist of a weight space $\mathcal{W} \subset \mathbb{R}^p$, an example space $\mathcal{Z}$, and a loss function $\ell : \mathcal{W} \times \mathcal{Z} \to [0,\infty)$. Let n be a fixed positive integer, and $\mathcal{S} = \mathcal{Z}^n$. Let $\hat{L} : \mathcal{S} \times \mathcal{W} \to \mathbb{R}$ be the empirical risk functional defined as $\hat{L}(s,w) = n^{-1}\sum_{i=1}^{n} \ell(w,z_i)$ for $s = (z_1,\dots,z_n) \in \mathcal{S}$; and write $\hat{L}_s(w) = \hat{L}(s,w)$. Let $P_1 \in \mathcal{M}_1(\mathcal{Z})$ and let $L : \mathcal{W} \to [0,\infty)$ be the risk functional $L(w) = \mathbb{E}[\ell(w,Z)]$ with $Z \sim P_1$.*

For any convex function $F : \mathbb{R}^2 \to \mathbb{R}$, define $f(s,w) = F(L(w),\hat{L}_s(w))$ for $(s,w) \in \mathcal{S} \times \mathcal{W}$. For any $P_n \in \mathcal{M}_1(\mathcal{S})$ and $Q^0 \in \mathcal{K}(\mathcal{S},\mathcal{W})$, let $\xi = (P \otimes Q^0)[e^f]$ be the exponential moment:

$$\xi = \int_{\mathcal{S}} \int_{\mathcal{W}} e^{f(s,w)} Q_s^0(dw) P_n(ds)\,. \tag{3.1}$$

Then for any $Q \in \mathcal{K}(\mathcal{S},\mathcal{H})$ and any $\delta \in (0,1)$, with probability of at least $1-\delta$ over the random draw of $S \sim P_n$ we have

$$F(Q_S[L], Q_S[\hat{L}_S]) \le \mathrm{KL}(Q_S \| Q_S^0) + \log(\xi/\delta)\,. \tag{3.2}$$

Notice the similarity with Theorem 2.1. There are, however, two important differences: First, Theorem 2.1 is for a data-free distribution Q^0, which means that Q^0 cannot depend on the data set S on which $\hat{L}_S$ is evaluated, whereas Theorem 3.2 is for a stochastic kernel Q^0 (from $\mathcal{S}$ to $\mathcal{W}$), which in particular implies that Q^0 can depend on the same data set S used for evaluating $\hat{L}_S$. Second, the conclusion of Theorem 2.1 is a high-probability inequality that holds simultaneously for all distributions Q; while that of Theorem 3.2 holds for a fixed stochastic kernel Q.

In view that the 'PAC-Bayes prior' in Theorem 3.2 is a stochastic kernel, this theorem may enable new bounds for data-dependent priors, while at the same time it can be used to recover PAC-Bayes bounds of similar form to the usual ones which were proved for 'data-free' priors, with the caviat that in this case the bounds hold for a given 'PAC-Bayes posterior' (a given stochastic kernel). This implies that PAC-Bayes bounds derived from Theorem 3.2 are not suitable for optimisation. Nevertheless, these PAC-Bayes bounds that hold for a given stochastic kernel are applicable to the case of risk certification for data-dependent distributions.

We emphasise that a 'data-free' distribution is equivalent to a constant stochastic kernel: $Q_s^0 = Q_{s'}^0$ for all $s,s' \in \mathcal{S}$. Hence $\mathcal{M}_1(\mathcal{W}) \subset \mathcal{K}(\mathcal{S},\mathcal{W})$, which implies that Theorem 3.2 does cover the usual case of data-free priors considered before in the literature. However, an attempt to 'recover' the usual bounds from Theorem 3.2 would give weaker conclusions that hold for a fixed posterior. Nevertheless, Theorem 3.2 can be regarded as a general template for deriving PAC-Bayes style bounds, not just with the usual 'data-free' priors such as those considered previously in the literature, but also more generally with data-dependent priors.

3.3 Discussion on implications of the main results

From Theorem 3.2 it is possible to derive PAC-Bayes style bounds of similar forms to the usual bounds (namely, those discussed in Section 2.2). This is done by suitable choices of the function F and a data-free prior Q^0, with the clarification that in this case the bounds hold for a given kernel (PAC-Bayes posterior) Q. For instance, $F(x,y) = 2n(x-y)^2$ yields a PAC-Bayes style bound akin to the classical bound of McAllester [1999], $F(x,y) = n\,\mathrm{kl}(y\|x)$ gives a bound akin to the bound of Langford and Seeger [2001], Seeger [2002] (which is also known as the PAC-Bayes-kl bound), and for a fixed $\beta > 0$ the function $F(x,y) = n\log\left(\frac{1}{1-x(1-e^{-\beta})}\right) - \beta ny$ gives a bound of similar form to the bound of Catoni [2007]. Furthermore, the choice $F(x,y) = n(x-y)^2/(2x)$ leads to a bound akin to the PAC-Bayes-quadratic bound of Rivasplata et al. [2019], or to one of similar form to the PAC-Bayes-λ bound of Thiemann et al. [2017].

An important role is played by ξ, the exponential moment of the chosen function under the joint distribution $P_n \otimes Q^0$. As discussed above in Section 2.2, there are essentially two main steps involved in obtaining a PAC-Bayes bound: (i) choose a function F (and a prior Q^0) to use in Theorem 2.1, and (ii) upper-bound the exponential moment ξ. The same two steps are needed for obtaining bounds based on Theorem 3.2.

We emphasise that the "usual assumptions" on which PAC-Bayes bounds were based, namely, (a) data-free prior, (b) bounded loss, and (c) i.i.d. data, played a role only in the technique for controlling the exponential moment ξ. This is because with a data-free Q^0 we may swap the order of integration:

$$\xi = \int_{\mathcal{S}}\int_{\mathcal{W}} e^{f(s,h)} Q^0(dw)P(ds) = \int_{\mathcal{W}}\int_{\mathcal{S}} e^{f(s,h)} P(ds)Q^0(dw) =: \xi_{\mathrm{swap}}\,.$$

Then bounding ξ proceeds by calculating or bounding ξ_{swap} for which there are readily available techniques under the assumptions of bounded loss and i.i.d. data (see e.g. Maurer [2004], Germain et al. [2009], van Erven [2014]). However, in principle these restrictions may be relaxed if other techniques are used to control ξ that do not rely on them, which is one of the important points that this work tries to highlight.

The novelty of Theorem 3.2 is in enabling PAC-Bayes style bounds where the PAC-Bayes prior Q^0 a data-dependent distribution. We discus below in Section 3.3.1 a novel PAC-Bayes bound with a data-dependent Gibbs prior, which is a first example of the new kinds of generalisation bounds that may be enabled by Theorem 3.2. The trade-off is that the bounds based on Theorem 3.2 are for a fixed stochastic kernel Q, while those based on Theorem 2.1 hold simultaneously for all distributions Q. In particular, the bounds based on Theorem 2.1, such as those discussed in Section 2.2, are suitable for optimisation over Q (because the bound holds uniformly for all Q). By contrast, the bounds derived from Theorem 3.2 are suitable for risk certification (for the data-dependent distribution Q_S found by some method).

Interestingly, Theorem 3.2 does not impose any restrictions on the loss function ℓ that is used in defining $L(w)$ and $\hat{L}_s(w)$. This feature is shared by Theorem 2.1. Hence, these theorems are, in principle, valid for *any* loss function: convex or non-convex, bounded or unbounded. Previous PAC-Bayes bounds imposed a restriction of bounded loss functions but, as pointed out above (and in Section 2.2), such restriction only played a role in the technique used to control ξ.

Notice also that Theorem 3.2 and Theorem 2.1 hold for any data-generating distribution $P_n \in \mathcal{M}_1(\mathcal{Z}^n)$, i.e. without restrictions on the data-generating process. In particular, at the level of generality at which these general theorems are presented, they hold beyond the i.i.d. data assumption. Hence, these theorems could potentially enable new generalisation bounds for statistically dependent data.

The next two subsections try to illustrate the novel kinds of PAC-Bayes style bounds with data-dependent priors that are enabled by Theorem 3.2.

3.3.1 A PAC-Bayes bound with a data-dependent Gibbs prior

We discuss a PAC-Bayes bound with a data-dependent prior that was given by Rivasplata et al. [2020]: Choosing the function $F(x,y) = \sqrt{n}(x-y)$ and choosing as 'prior' an empirical Gibbs distribution $Q_s^0(dw) \propto e^{-\gamma \hat{L}(w,s)}\mu(dw)$ for some fixed $\gamma > 0$ and base measure μ over $\mathcal{W}$, we proved that for any kernel $Q \in \mathcal{K}(\mathcal{S},\mathcal{W})$ and $\delta \in (0,1)$, with probability of at least $1-\delta$ over size-n i.i.d. random samples S we have

$$Q_S[L] - Q_S[\hat{L}_S] \leq \frac{1}{\sqrt{n}}\left(\mathrm{KL}(Q_S \| Q_S^0) + 2\Big(1+\frac{2\gamma}{\sqrt{n}}\Big) + \log\Big(\frac{1+\sqrt{e}}{\delta}\Big)\right) . \tag{3.3}$$

This bound follows from Theorem 3.2 and a stability analysis for the empirical Gibbs distribution to control the exponential moment (the details are in [Rivasplata et al., 2020, Appendix B]). To the best of my knowledge, this was the first work to extend the PAC-Bayes analysis to stochastic kernels. The PAC-Bayes bound with a data-dependant (Gibbs) prior just discussed shows the versatility of the approach. This framework appears to be a promising theoretical tool to obtain new results, which potentially could lead to tighter bounds.

Notice that this prior allowed to remove 'log(n)' from the usual PAC-Bayes bounds (Cf. Section 2.2, the PAC-Bayes-kl bound and its relaxations). This was one of the important contributions of Catoni [2007], who also used a data-dependent Gibbs distribution, see Catoni [2007, Theorem 1.2.4, Theorem 1.3.1, & corollaries]. Interestingly, the choice $Q = Q^0$ gives the smallest right-hand side in Eq. (3.3) (however, it does not necessarily minimize the bound on $Q_S[L]$) which leads to the following for the Gibbs learner: $Q_S[L] - Q_S[\hat{L}_S] \lesssim 1/\sqrt{n} + \gamma/n$. Notice that this latter bound has an additive $1/\sqrt{n}$ compared to the bound in expectation of Raginsky et al. [2017].

3.3.2 PAC-Bayes bounds with d-stable data-dependent priors

Next we discuss an approach to convert any PAC-Bayes bound that is valid for a usual 'data-free' prior into a bound with a stable data-dependent prior, which is accomplished by generalising a technique from Dziugaite and Roy [2018a]. We need a definition: When we say that $\pi \in \mathcal{K}(\mathcal{S},\mathcal{W})$ satisfies the DP property with $\varepsilon > 0$ (written DP(ε) for short) we mean that whenever s and s' differ only at one element, the corresponding distributions over $\mathcal{W}$ satisfy:

$$\frac{d\pi_s}{d\pi_{s'}} \leq e^{\varepsilon}\,.$$

This condition on the Radon-Nikodym derivative is equivalent to the condition that, whenever s and s' differ at one entry, the ratio $\pi(s,A)/\pi(s',A)$ is upper bounded by e^{ε}, for all sets $A \in \Sigma_{\mathcal{W}}$. Thus, the property entails stability of the data-dependent distribution π_s with respect to small changes in the composition of the n-tuple s. This definition goes back to the literature on privacy-preserving methods for data analysis [Dwork et al., 2015b]; however, we are interested in its formal properties only. It captures a kind of 'distributional stability' which we refer to as 'd-stability' for short.

Essentially, Dziugaite and Roy [2018a] show that for any fixed 'data-free' distribution $Q^* \in \mathcal{M}_1(\mathcal{W})$ and stochastic kernel $Q^0 \in \mathcal{K}(\mathcal{S},\mathcal{W})$ satisfying the DP(ε) property, one can turn the inequality $F(Q_S[L],Q_S[\hat{L}_S]) \leq \mathrm{KL}(Q_S\|Q^*) + \log(\xi(Q^*)/\delta)$ into

$$F(Q_S[L],Q_S[\hat{L}_S]) \leq \mathrm{KL}(Q_S\|Q^0_S) + \log(2\xi(Q^*)/\delta) + \frac{n\varepsilon^2}{2} + \varepsilon\sqrt{\frac{n}{2}\log(\frac{4}{\delta})}\,. \tag{3.4}$$

In other words, if Eq. (3.2) holds with a data-free prior Q^*, then Eq. (3.4) holds with a data-dependent prior that is distributionally stable (i.e. satisfies DP(ε)). Note that different choices of F would lead to different bounds on $\xi(Q^*)$ —essentially, upper bounds on the exponential moment typically considered in the PAC-Bayesian literature. For example, taking $F(x,y) = n\,\mathrm{kl}(y\|x)$ one can show that $\xi(Q^*) \leq 2\sqrt{n}$ [Maurer, 2004], and this leads to Theorem 4.2 of Dziugaite and Roy [2018a]: if $Q^0 \in \mathcal{K}(\mathcal{S},\mathcal{W})$ satisfies the DP(ε) property, then for any kernel $Q \in \mathcal{K}(\mathcal{S},\mathcal{W})$ and $\delta \in (0,1)$, with probability at least $1-\delta$ over size-n i.i.d. samples S we have

$$\mathrm{kl}(Q_S[\hat{L}_S]\|Q_S[L]) \leq \frac{1}{n}\left(\mathrm{KL}(Q_S\|Q^0_S) + \log(\frac{4\sqrt{n}}{\delta}) + \frac{n\varepsilon^2}{2} + \varepsilon\sqrt{\frac{n}{2}\log(\frac{4}{\delta})}\right)\,.$$

Eq. (3.4) is a general version of this result, whose derivation uses the notion of *max-information* [Dwork et al., 2015a] and, in particular, an inequality that upper-bounds this quantity. The details of the general conversion recipe (from 'data-free' prior to 'data-dependent d-stable' prior) are given in [Rivasplata et al., 2020, Appendix C].

N.B.: Rivasplata et al. [2020, Section 2.3] also presented an inequality for randomised linear classifiers with the unbounded square loss and a data-dependent prior. That result is not reproduced here because it is a bit of a detour from the topics of this thesis.

3.4 Conclusion and Future Work

This chapter is based on my paper Rivasplata et al. [2020]. This work presented two basic high-probability inequalities for stochastic kernels, and a general PAC-Bayes theorem for stochastic kernels, from which one may derive PAC-Bayes style bounds similar to the various known PAC-Bayes bounds, as well as novel bounds. The chapter discussed the generality of the new framework, and its trade-offs compared to the standard PAC-Bayes bounds. Most importantly, the new framework enabled PAC-Bayes style bounds where the prior is a data-dependent distribution (kernel) by default, at the price of conclusions that hold with high probability for a fixed 'posterior' distribution, whereas the classical PAC-Bayes bounds hold uniformly for all distributions.

Notice that despite their variety and attractive properties, the results in the vast majority of the previous literature on PAC-Bayes bounds shared two crucial limitations: the prior Q^0 cannot depend on the training data S and the loss function has to be bounded. These limitations had been removed in the PAC-Bayesian literature in special cases under strong assumptions. Additionally, the i.i.d. data assumption could be seen as a third crucial limitation, since it prevents the applicability of PAC-Bayes bounds to learning problems with statistically dependent data.

The work presented in this chapter (and in Section 2.2) clarified the role of the requirements of fixed 'data-free' priors, bounded losses, and i.i.d. data, highlighting that those requirements were used in the techniques to upper-bound an exponential moment term, while the essential part of the PAC-Bayes argument remains valid without those restrictions. This is an important insight indicating that to develop bounds that hold without those classical requirements, one needs to find techniques for upper-bounding the exponential moment corresponding to suitably chosen functions.

The chapter presented two bounds that illustrate the use of data-dependent priors. The first example, that of Section 3.3.1, involved a data-dependent Gibbs distribution and its proof used a stability analysis to upper-bound the exponential moment of a suitably chosen function, which led to a novel PAC-Bayes style bound (Eq. (3.3)). The second example, that of Section 3.3.2, generalised a technique from Dziugaite and Roy [2018a] thus allowing to convert any PAC-Bayes bound which is valid for a data-free prior into a bound for a d-stable data-dependent prior (Eq. (3.4)).

Future work should aim to derive novel PAC-Bayes bounds with data-dependent priors. This may require novel techniques to control the exponential moments of suitably chosen functions. Of particular interest is deriving tight bounds which could be helpful for risk certification. Additional interesting questions concern deriving novel bounds with unbounded losses, and bounds for statistically dependent data.

Chapter 4

PAC-Bayes bounds for stable algorithms with distribution-dependent priors

N.B.: The content of this chapter is my paper Rivasplata et al. [2018], with minor modifications to make the content consistent with other chapters of this thesis.

This work combined two directions of research: algorithmic stability, and PAC-Bayes bounds for algorithms that randomise with a data-dependent distribution. The combination of these ideas enabled the development of risk bounds that exploit stability of the learned hypothesis but are independent of the complexity of the hypothesis class. Specifically, Rivasplata et al. [2018] derived PAC-Bayes bounds for Hilbert space valued algorithms whose output is stable with respect to small changes in the composition of the training set (the precise definition of stability is given in Definition 4.1 below). This is an example of using a PAC-Bayes bound for risk certification. The PAC-Bayes-kl bound is used here with 'priors' defined in terms of the data-generating distribution, as introduced by Catoni [2007] and developed further e.g. by Lever et al. [2010, 2013] and Parrado-Hernández et al. [2012].

The stability analysis carried out by Bousquet and Elisseeff [2002] followed and extended the work of Lugosi and Pawlak [1994] and was further developed by Celisse and Guedj [2016], Abou-Moustafa and Szepesvári [2017], Liu et al. [2017], Bousquet et al. [2020], among others. This analysis shows that stability can be used to give bounds on the generalisation of the learned functions if the learning algorithm is stable, in the sense that slightly different training sets give similar solutions. Intuitively, this is because stable learning should ensure that the output is not too sensitive to small changes in the training set, and stability-based bounds depend on a quantity that measures this sensitivity.

In this work stability is measured by the sensitivity coefficients (see Definition 4.1 below) of a Hilbert space valued algorithm. This setting encompasses learning algorithms whose output is a 'weight vector' in a Hilbert space of arbitrary dimension. We provide an analysis leading to a PAC-Bayes bound for randomised classifiers under Gaussian randomisation. As a by-product of the stability analysis we derive a concentration inequality for the learned weight vector. Applying it to the solution of a Support Vector Machine (SVM) we deduce a concentration bound for the SVM weight vector, and a PAC-Bayes bound for the randomised predictor defined by Gaussian randomisation centered at the SVM solution. Experimental results compare our new bound with other stability-based generalisation bounds, and with a more standard PAC-Bayes bound for randomised Gaussian classifiers. We also report experiments with their use in model selection.

Chapter Layout. This chapter is organised as follows. Section 4.1, after briefly going over some chapter-specific notations, defines the meaning of weight stability, based on weight sensitivity coefficients. This is the stability notion on which the main results (Theorem 4.2 and Corollary 4.3) are based. Then, Section 4.2 gives a side-by-side comparison to other generalisation bounds. Section 4.3 presents the detailed proofs of the main results. The technical Section 4.4 is about Gaussian distributions over an infinite-dimensional Hilbert space. Section 4.5 shows the results of the experiments. Finally, Section 4.6 concludes the chapter and discusses future work.

4.1 Definition and Main Results

N.B.: This chapter presents the results of Rivasplata et al. [2018] reformulated in terms of the notation introduced in Chapter 2, for the sake of consistency.

A supervised learning algorithm is a function that maps finite lists of labelled examples to hypotheses, where the latter are functions that belong to some hypothesis class $\mathcal{H}$. As discussed in Chapter 2, we focus on function classes of the form $\mathcal{H} = \{h_w \mid w \in \mathcal{W}\}$ corresponding to a weight space $\mathcal{W}$. Each possible weight vector $w \in \mathcal{W}$ maps to a predictor function $h_w : \mathcal{X} \to \mathcal{Y}$. For the sake of simplicity we may consider learning algorithms that take in a set of labelled examples and output a weight vector. Writing $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$ for the space of input-label pairs, such a learning algorithm can be formalised as a mapping $\mathsf{A} : \cup_n \mathcal{Z}^n \to \mathcal{W}$.

We consider classification tasks, in which case it is natural to use the zero-one loss as the loss function for defining the performance measures (risk, empirical risk). We use the notation $\ell_{01}(y', y) = \mathbf{1}[y' \neq y]$ for the zero-one loss, where $\mathbf{1}[\cdot]$ is an indicator function equal to 1 when the argument is true and equal to 0 when the argument is false. Under this loss function the risk of w (see Eq. (2.1)) takes the form $L^{01}(w) = \mathbb{E}[\mathbf{1}[h_w(X) \neq Y]] = \mathbb{P}[h_w(X) \neq Y]$, i.e., $L^{01}(w)$ equals the probability of misclassifying the random example $(X,Y) \sim P$ when using w; and the empirical risk of w (see Eq. (2.2)) over the list of examples $s = ((x_1,y_1),\dots,(x_n,y_n))$ is given by $\hat{L}^{01}_s(w) = \frac{1}{n}\sum_{i=1}^n \mathbf{1}[h_w(x_i) \neq y_i]$, i.e., $\hat{L}^{01}_s(w)$ is the in-sample proportion of misclassified examples when using w.

The main theorem of Rivasplata et al. [2018] concerns Hilbert space valued algorithms, in the sense that the hypothesis space is a Hilbert space. Again, for the sake of simplicity we consider algorithms whose output is a weight vector w from a weight space $\mathcal{W}$. In this case the norm $\|w\| = \sqrt{\langle w,w\rangle}$ is defined by the inner product $\langle \cdot,\cdot \rangle$, and we may use this norm to measure the difference between the algorithm's outputs corresponding to different samples.

Some notation related to lists will be used to shorten the notation. We define $[n] := \{1,\dots,n\}$ as the initial segment consisting of the first n positive integers. For a list $\xi_1,\xi_2,\xi_3,\dots$ where the list members may belong to any given set, and for any indexes $i < j$, we define $\xi_{i:j} := (\xi_i,\dots,\xi_j)$, i.e. $\xi_{i:j}$ is a shortcut for the segment of the list from ξ_i to ξ_j.

Definition 4.1. *Consider a learning algorithm* $\mathsf{A} : \cup_n \mathcal{Z}^n \to \mathcal{W}$ *where* $\mathcal{W}$ *is a separable Hilbert space, and let* $\|\cdot\|$ *be the norm induced by the inner product of this space. We define the* weight sensitivity coefficient *of* A *at sample size n as follows:*

$$\beta_n = \sup_{i\in[n]} \sup_{z_i,z_i'} \|\mathsf{A}(z_{1:i-1},z_i,z_{i+1:n}) - \mathsf{A}(z_{1:i-1},z_i',z_{i+1:n})\| \,.$$

This definition is close in spirit to what is called "uniform stability" in the literature, except that our definition concerns stability of the learned weight vector (measured by a distance on the weight space), while e.g. Bousquet and Elisseeff [2002] deal with stability of the loss functional. The latter could be called "loss stability" (in terms of "loss sensitivity coefficients") for the sake of informative names. A more sensible definition than our definition of weight stability could be to consider a notion of *hypothesis stability* quantified by *hypothesis sensitivity coefficients* where a suitable notion of distance is assumed in the hypothesis space directly. The latter definition could be better suited to account for the existence of symmetries where different weight vectors may encode the same hypothesis.

Writing $z_{1:n} \approx z'_{1:n}$ when these n-tuples differ at one entry (at most), an equivalent formulation to the above is $\beta_n = \sup_{z_{1:n}\approx z'_{1:n}} \|\mathsf{A}(z_{1:n}) - \mathsf{A}(z'_{1:n})\|$. In particular, if two random samples S_n and S_n' differ only on one example, then $\|\mathsf{A}(S_n) - \mathsf{A}(S_n')\| \leq \beta_n$. Thus our definition implies stability with respect to replacing one example with an independent copy. Alternatively, one could define $\beta_n = \operatorname{ess\,sup}_{S_n\approx S_n'} \|\mathsf{A}(S_n) - \mathsf{A}(S_n')\|$, which corresponds to the "uniform argument stability" of Liu et al. [2017]. We avoid the 'almost-sure' technicalities by defining our β_n's as the maximal difference (in norm) with respect to all n-tuples $z_{1:n} \approx z'_{1:n}$. The extension to sensitivity when changing several examples is natural: $\|\mathsf{A}(z_{1:n}) - \mathsf{A}(z'_{1:n})\| \leq \beta_n \sum_{i=1}^n \mathbf{1}[z_i \neq z_i']$. Note that β_n is a Lipschitz factor with respect to the Hamming distance. The "total Lipschitz stability" of Kontorovich [2014] is a similar notion for stability of the loss functional. The "collective stability" of London et al. [2013] is not comparable to ours (different setting) despite the similar look.

We consider randomised classifiers that operate as follows. Let $Q \in \mathcal{M}_1(\mathcal{W})$ be a probability distribution over the weight space. To make a prediction the randomised classifier picks a random $W \in \mathcal{W}$ according to Q and predicts a label with h_W. Each prediction is made with a fresh draw of W. For simplicity we use the same label Q for the probability distribution and for the corresponding randomised classifier. The risk measures $L(w)$ and $\hat{L}_s(w)$ are extended to randomised classifiers by averaging: $L(Q) \equiv \int_{\mathcal{W}} L(w)Q(dw)$ is the *average true risk* of Q, and $\hat{L}_s(Q) \equiv \int_{\mathcal{W}} \hat{L}_s(w)Q(dw)$ is its empirical counterpart.

Recall that given two distributions $Q, Q_0 \in \mathcal{M}_1(\mathcal{W})$, the Kullback-Leibler divergence (a.k.a. relative entropy) of Q with respect to Q_0 is

$$\mathrm{KL}(Q \| Q_0) = \int_{\mathcal{W}} \log\Big(\frac{dQ}{dQ_0}\Big)\, dQ\,.$$

Of course this makes sense when Q is absolutely continuous with respect to Q_0, which ensures that the Radon-Nikodym derivative dQ/dQ_0 exists. For Bernoulli distributions with parameters q and q_0 we write $\mathrm{kl}(q\|q_0) = q\log(\frac{q}{q_0}) + (1-q)\log(\frac{1-q}{1-q_0})$, and $\mathrm{kl}_+(q\|q_0) = \mathrm{kl}(q\|q_0)\mathbf{1}[q < q_0]$.

4.1.1 Main theorem: a PAC-Bayes bound for stable algorithms with Gaussian randomisation

Theorem 4.2. *Let $\mathsf{A} : \cup_n \mathcal{Z}^n \to \mathcal{W}$ be a Hilbert space valued algorithm. Suppose that the post-training predictions are to be randomised according to Gaussian distributions $Q = \mathcal{N}(\mathsf{A}(S), \sigma^2 I)$. If A has weight stability coefficient β_n at sample size n, then for any randomisation variance $\sigma^2 > 0$, for any $\delta \in (0,1)$, with probability of at least $1-2\delta$ over size-n i.i.d. samples S we have*[1]

$$\mathrm{kl}(\hat{L}_S(Q)\|L(Q)) \le \frac{\frac{n\beta_n^2}{2\sigma^2}\left(1+\sqrt{\frac{1}{2}\log\big(\frac{1}{\delta}\big)}\,\right)^2 + \log(\frac{2\sqrt{n}}{\delta})}{n}\,.$$

The proof, which is given in Section 4.3 below, combines stability of the learned weight vector (as in our Definition 4.1) and the PAC-Bayes-kl bound (see Eq. (2.8) in Section 2.2). Notice that the randomising distribution Q is data-dependent, since it is a Gaussian distribution centered at the data-dependent weight vector output by the learning algorithm. Notice also that the scale parameter σ, which is the standard deviation of Q in each direction, is fixed independently of the learning algorithm. This, in particular, implies that one cannot optimise σ for free: Using the bound to find the best out of K possible values for σ would worsen the bound by an additive $\log(K)$ in the numerator of the upper bound.

[1]The log term in the corresponding result of Rivasplata et al. [2018] followed the form of the bound presented by Langford [2005]. The form presented here is with the sharp dependence on n clarified by Maurer [2004].

4.1.2 Application: a PAC-Bayes bound for SVM with Gaussian randomisation

For a Support Vector Machine (SVM) with feature map $\varphi : \mathcal{X} \to \mathcal{W}$ into a separable Hilbert space $\mathcal{W}$, we may identify[2] a linear classifier $c_w(\cdot) = \mathrm{sign}(\langle w, \varphi(\cdot)\rangle)$ with a vector $w \in \mathcal{W}$. With this identification we can regard an SVM as a Hilbert space[3] valued mapping that based on a training sample S learns a weight vector $W = \mathrm{SVM}(S) \in \mathcal{W}$. In this context, stability of the SVM's solution then reduces to stability of the learned weight vector.

To be specific, let $\mathrm{SVM}_\lambda(S)$ be the SVM that regularises the empirical risk over the sample $S = ((X_1,Y_1),\dots,(X_n,Y_n))$ by solving the following optimisation problem:

$$\arg\min_w \left(\frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_{i=1}^n \ell(h_w(X_i),Y_i) \right). \tag{4.1}$$

Our stability coefficients in this case satisfy $\beta_n \leq \frac{2}{\lambda n}$ (Example 2 of Bousquet and Elisseeff [2002], adapted to our setting). Then a direct application of our Theorem 4.2 together with a concentration argument for the SVM weight vector (see our Corollary 4.8 below) gives the following:

Corollary 4.3. *Let $W = \mathrm{SVM}_\lambda(S)$. Suppose that the post-training predictions are randomised according to Gaussian[4] distributions $Q = \mathcal{N}(W, \sigma^2 I)$. For any randomisation variance $\sigma^2 > 0$, for any $\delta \in (0,1)$, with probability of at least $1-2\delta$ over size-n i.i.d. samples S we have*

$$\mathrm{kl}(\hat{L}_S(Q)\|L(Q)) \leq \frac{\frac{2}{\sigma^2\lambda^2 n}\left(1+\sqrt{\frac{1}{2}\log\left(\frac{1}{\delta}\right)}\right)^2 + \log\left(\frac{2\sqrt{n}}{\delta}\right)}{n}.$$

In closing this section we mention that our main theorem is general in the sense that it covers any Hilbert space valued algorithm. This result may be specialised to any regularised ERM algorithm [Liu et al., 2017]. We applied it to SVM's whose weight sensitivity coefficients (as in our Definition 4.1) are known. It can be argued that neural networks (NN's) fall under this framework as well. Then an appealing future research direction, with deep learning in view, is to figure out the sensitivity coefficients of NN's trained by Stochastic Gradient Descent. Then our main theorem could be applied to provide non-vacuous bounds for the performance of NN's, which we believe is very much needed.

[2] Riesz representation theorem is behind this identification.

[3] $\mathcal{W}$ may be infinite-dimensional (e.g. Gaussian kernel).

[4] See Section 4.4 about the interpretation of Gaussian randomisation for a Hilbert space valued algorithm.

4.2 Comparison to other bounds

For reference we list several risk bounds (including ours). They are in the context of binary classification ($\mathcal{Y} = \{-1,+1\}$). For clarity, risks under the 0-1 loss are denoted by L^{01} and risks with respect to the (clipped) hinge loss are denoted by L^{hi}. Bounds requiring a Lipschitz loss function do not apply to the 0-1 loss. However, the 0-1 loss is upper bounded by the hinge loss, allowing us to upper bound the risk with respect to the former in therms of the risk with respect to the latter. On the other hand, results requiring a bounded loss function do not apply to the regular hinge loss. In those cases the clipped hinge loss is used, which enjoys boundedness and Lipschitz continuity.

4.2.1 P@EW: Our new instance-dependent PAC-Bayes bound

Our Corollary 4.3, with $Q = \mathcal{N}(W_n, \sigma^2 I)$, a Gaussian centered at $W_n = \mathrm{SVM}_\lambda(S_n)$ with a fixed randomisation variance σ^2, gives the following risk bound which holds with probability $\geq 1-2\delta$:

$$\mathrm{kl}(\hat{L}_S^{01}(Q)\|L^{01}(Q)) \leq \frac{2}{\sigma^2\lambda^2 n^2}\left(1+\sqrt{\frac{1}{2}\log\left(\frac{1}{\delta}\right)}\right)^2 + \frac{1}{n}\log\left(\frac{2\sqrt{n}}{\delta}\right).$$

As will be clear from the proof (see Section 4.3 below), this bound is obtained from the PAC-Bayes-kl bound (see Eq. (2.8)) using a prior $Q_0 = \mathcal{N}(\mathbb{E}[W_n], \sigma^2 I)$ centered at the expected weight. This bound is with λ as in our formulation of SVM given in Eq. (4.1) above.

4.2.2 P@O: Prior at the origin PAC-Bayes bound

The PAC-Bayes-kl bound Eq. (2.8) again with $Q = \mathcal{N}(W_n, \sigma^2 I)$, gives the following risk bound which holds with probability $\geq 1-\delta$:

$$\forall \sigma^2, \quad \mathrm{kl}(\hat{L}_S^{01}(Q)\|L^{01}(Q)) \leq \frac{1}{2\sigma^2 n}\|W_n\|^2 + \frac{1}{n}\log\left(\frac{2\sqrt{n}}{\delta}\right).$$

This is the PAC-Bayes bound for SVM given by Corollary 5.4 of Langford [2005], which is obtained by using a prior $Q_0 = \mathcal{N}(0, \sigma^2 I)$ centered at the origin.

4.2.3 Bound of Liu et al. [2017]

From Corollary 1 of Liu et al. [2017] (but with λ as in the formulation of Eq. (4.1)) we get the following risk bound which holds with probability $\geq 1-2\delta$:

$$L^{01}(W_n) \leq L^{\mathrm{hi}}(W_n) \leq \hat{L}_S^{\mathrm{hi}}(W_n) + \frac{\mathbf{8}}{\lambda n}\sqrt{2\log\left(\frac{2}{\delta}\right)} + \sqrt{\frac{1}{2n}\log\left(\frac{1}{\delta}\right)}.$$

We use Corollary 1 of Liu et al. [2017] with $B = 1$, $L = 1$ and $M = 1$ (clipped hinge loss).

4.2.4 Bound of Bousquet and Elisseeff [2002]

From Example 2 of Bousquet and Elisseeff [2002] (but with λ as in the formulation of Eq. (4.1)) we get the following risk bound which holds with probability $\geq 1-\delta$:

$$L^{01}(W_n) \leq L^{\mathrm{hi}}(W_n) \leq \hat{L}^{\mathrm{hi}}_S(W_n) + \frac{\mathbf{2}}{\lambda n} + \Big(1+\frac{\mathbf{4}}{\lambda}\Big)\sqrt{\frac{1}{2n}\log\big(\frac{1}{\delta}\big)}\,.$$

We use Example 2 and Theorem 17 (based on Theorem 12) of Bousquet and Elisseeff [2002] with $\kappa = 1$ (normalised kernel) and $M = 1$ (clipped hinge loss).

In Section 4.2.5 below there is a list of different SVM formulations, and how to convert between them. We found it useful when implementing code for experiments.

There are obvious differences in the nature of these bounds: the last two (Liu et al. [2017] and Bousquet and Elisseeff [2002]) are risk bounds for the (un-randomised) classifiers, while the first two (P@EW, P@O) give an upper bound on the KL-divergence between the average risks (empirical to theoretical) of the randomised classifiers. Of course inverting the KL-divergence we get a bound for the average theoretical risk in terms of the average empirical risk and the (square root of the) right hand side. Also, the first two bounds have an extra parameter, the randomisation variance (σ^2), and one may naturally ask when and how this parameter could be optimised. Note that P@O bound is not based on stability, while the other three bounds are based on stability notions. Next let us comment on how these bounds compare quantitatively.

Our P@EW bound and the P@O bound are similar except for the first term on the right hand side. This term comes from the KL-divergence between the Gaussian distributions (i.e. $\mathrm{KL}(Q\|Q_0)$). Our P@EW bound's first term improves with larger values of λ, which in turn penalise the norm of the weight vector of the corresponding SVM, resulting in a small first term in P@O bound. Note that P@O bound is equivalent to the setting of $Q = \mathcal{N}(\mu W_n, I)$, a Gaussian with identity covariance and with center along the direction of W_n at distance $\mu = 1/\sigma$ from the origin, and $Q_0 = \mathcal{N}(0, I)$ a Gaussian with identity covariance and center at the origin of the system of coordinates (as discussed by Langford [2005] and implemented by Parrado-Hernández et al. [2012]). This also explains why the P@O bound holds uniformly over σ, which is an important difference with P@EW since the latter holds for a fixed σ.

The first term on the right hand side of our P@EW bound comes from the concentration of the weight vector (see our Corollary 4.8). Lemma 1 of Liu et al. [2017] implies a similar concentration inequality for the weight vector, but it is not hard to see that our concentration bound is slightly better.

Finally, in the experiments we compare numerically our P@EW bound with Bousquet and Elisseeff [2002].

4.2.5 SVM weight vector: clarification about formulations

We have a sample of size n. In the standard implementation the weight vector $W_n(C)$ found by SVM is a solution of the following optimisation problem:

$$W_n(C) = \arg\min_w \left(\frac{1}{2} \|w\|^2_{\mathcal{H}} + C \sum_{i=1}^n \xi_i \right). \tag{svm1}$$

In our work the weight vector $W_n^{\text{OURS}}(\lambda)$ found by SVM is a solution of the following optimisation problem (cf. Eq. (4.1)):

$$W_n^{\text{OURS}}(\lambda) = \arg\min_w \left(\frac{\lambda}{2} \|w\|^2_{\mathcal{H}} + \frac{1}{n} \sum_{i=1}^n \xi_i \right). \tag{svm2}$$

In Bousquet and Elisseeff [2002] and Liu et al. [2017] the weight vector $W_n^{\text{B\&E}}(\lambda)$ found by SVM is a solution of the following optimisation problem:

$$W_n^{\text{B\&E}}(\lambda) = \arg\min_w \left(\lambda \|w\|^2_{\mathcal{H}} + \frac{1}{n} \sum_{i=1}^n \xi_i \right). \tag{svm3}$$

The minimum is over $w \in \mathcal{H}$ (an appropriate Hilbert space) and subject to some constrains for the ξ_i's in all cases. The relation between them is:

- $W_n^{\text{OURS}}(\lambda) = W_n^{\text{B\&E}}(\lambda/2)$
- $W_n^{\text{B\&E}}(\lambda) = W_n(C)$ with $C = \frac{1}{2n\lambda}$
- $W_n^{\text{OURS}}(\lambda) = W_n(C)$ with $C = \frac{1}{n\lambda}$

4.3 Proofs

As we said before, the proof of our Theorem 4.2 combines stability of the learned weight vector (in the sense of our Definition 4.1) and the PAC-Bayes-kl bound (see Eq. (2.8) in Section 2.2), quoted next for reference: For any data-free distribution $Q_0 \in \mathcal{M}_1(\mathcal{W})$, and for any $\delta \in (0,1)$, with probability of at least $1-\delta$ over the random draw of size-n i.i.d. samples S, simultaneously for all distributions $Q \in \mathcal{M}_1(\mathcal{W})$ we have

$$\mathrm{kl}(\hat{L}_S(Q) \| L(Q)) \leq \frac{\mathrm{KL}(Q\|Q_0) + \log(\frac{2\sqrt{n}}{\delta})}{n}.$$

Consider a learning algorithm $\mathsf{A} : \cup_n \mathcal{Z}^n \to \mathcal{W}$ We use the PAC-Bayes-kl bound with a Gaussian 'posterior' distribution $Q = \mathcal{N}(\mathsf{A}(S_n), \sigma^2 I)$ centered at the random output $\mathsf{A}(S_n)$, and a Gaussian 'prior' $Q_0 = \mathcal{N}(\mathbb{E}[\mathsf{A}(S_n)], \sigma^2 I)$ centered at the expected output, both with covariance operator $\sigma^2 I$. The KL-divergence between those Gaussians scales with $\|\mathsf{A}(S_n) - \mathbb{E}[\mathsf{A}(S_n)]\|^2$. More precisely:

$$\mathrm{KL}(Q\|Q_0) = \frac{1}{2\sigma^2} \|\mathsf{A}(S_n) - \mathbb{E}[\mathsf{A}(S_n)]\|^2.$$

Therefore, bounding $\|\mathsf{A}(S_n) - \mathbb{E}[\mathsf{A}(S_n)]\|$ will give (via the PAC-Bayes-kl bound) a corresponding bound on the divergence between the average empirical risk $\hat{L}_S(Q)$ and the average population risk $L(Q)$ of the randomised classifier Q. Then inverting $\mathrm{kl}(\hat{L}_S(Q)\|L(Q))$ with respect to its second argument we get an upper bound for $L(Q)$ in terms of the empirical $\hat{L}_S(Q)$ and the stability coefficients, which holds with high probability. Stability (in the form of our Definition 4.1) implies a concentration inequality for $\|\mathsf{A}(S_n) - \mathbb{E}[\mathsf{A}(S_n)]\|$. This is done in our Corollary 4.7 (see Section 4.3.3 below) and completes the circle of ideas to prove our main theorem. The proof of our concentration inequality is based on an extension of the bounded differences theorem of McDiarmid [1989] to vector-valued functions discussed next.

4.3.1 McDiarmid's inequality for real-valued functions of the sample

The training sample is $Z_{1:n} = (Z_1, \ldots, Z_n)$ where each example Z_i is a random variable taking values in the (measurable) space $\mathcal{Z}$. We quote a well-known theorem of McDiarmid [1989]:

Theorem 4.4. (McDiarmid's inequality) *Let $Z_1, \ldots, Z_n$ be independent $\mathcal{Z}$-valued random variables, and $f : \mathcal{Z}^n \to \mathbb{R}$ a real-valued function such that for each i and for each list of 'complementary' arguments $z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_n$ we have*

$$\sup_{z_i, z_i'} |f(z_{1:i-1}, z_i, z_{i+1:n}) - f(z_{1:i-1}, z_i', z_{i+1:n})| \leq c_i \,.$$

Then for every $\varepsilon > 0$, $\mathbb{P}\{f(Z_{1:n}) - \mathbb{E}[f(Z_{1:n})] > \varepsilon\} \leq \exp\left(\frac{-2\varepsilon^2}{\sum_{i=1}^n c_i^2}\right)$.

McDiarmid's inequality applies to a *real-valued* function of independent random variables. Next we present an extension to *vector-valued* functions of independent random variables.

4.3.2 McDiarmid's inequality for vector-valued functions of the sample

Let $Z_1, \ldots, Z_n$ be independent $\mathcal{Z}$-valued random variables and $f : \mathcal{Z}^n \to \mathcal{W}$ a function into a separable Hilbert space. We will prove that bounded differences *in norm*[5] implies concentration of $f(Z_{1:n})$ around its mean *in norm*, i.e., that $\|f(Z_{1:n}) - \mathbb{E}f(Z_{1:n})\|$ is small with high probability.

Notice that McDiarmid's theorem can't be applied directly to $f(Z_{1:n}) - \mathbb{E}f(Z_{1:n})$ when f is vector-valued. We will apply McDiarmid to the real-valued $\|f(Z_{1:n}) - \mathbb{E}f(Z_{1:n})\|$, which will give an upper bound for $\|f - \mathbb{E}f\|$ in terms of $\mathbb{E}\|f - \mathbb{E}f\|$. The next lemma upper bounds $\mathbb{E}\|f - \mathbb{E}f\|$ for a function f with bounded differences in

[5]The Hilbert space norm, induced by the inner product of $\mathcal{W}$.

norm. The proof follows the steps of the proof of the classic result of McDiarmid [1989] quoted above, we give the details in Section 4.3.4.

Lemma 4.5. *Let $Z_1,\dots,Z_n$ be independent $\mathcal{Z}$-valued random variables, and $f : \mathcal{Z}^n \to \mathcal{W}$ a function into a Hilbert space $\mathcal{W}$ satisfying the bounded differences property: for each i and for each list of 'complementary' arguments $z_1,\dots,z_{i-1},z_{i+1},\dots,z_n$ we have*

$$\sup_{z_i,z_i'} \|f(z_{1:i-1},z_i,z_{i+1:n}) - f(z_{1:i-1},z_i',z_{i+1:n})\| \leq c_i\,.$$

Then $\mathbb{E}\|f(Z_{1:n}) - \mathbb{E}[f(Z_{1:n})]\| \leq \sqrt{\sum_{i=1}^n c_i^2}$.

If the vector-valued function $f(z_{1:n})$ has bounded differences in norm (as in Lemma 4.5) and $C \in \mathbb{R}$ is any constant, then the real-valued function $\|f(z_{1:n}) - C\|$ has the bounded differences property (as in McDiarmid's theorem). In particular this is true for $\|f(z_{1:n}) - \mathbb{E}f(Z_{1:n})\|$ (notice that $\mathbb{E}f(Z_{1:n})$ is constant over replacing Z_i by an independent copy Z_i') so applying McDiarmid's inequality to it, combining with Lemma 4.5, we get the following theorem:

Theorem 4.6. *Under the assumptions of Lemma 4.5, for any $\delta \in (0,1)$, with probability of at least $1-\delta$ we have*

$$\|f(Z_{1:n}) - \mathbb{E}[f(Z_{1:n})]\| \leq \sqrt{\sum_{i=1}^n c_i^2} + \sqrt{\frac{\sum_{i=1}^n c_i^2}{2}\log\left(\frac{1}{\delta}\right)}\,.$$

Notice that the vector $c_{1:n} = (c_1,\dots,c_n)$ of difference bounds appears in the above inequality only through its Euclidean norm $\|c_{1:n}\|_2 = \sqrt{\sum_{i=1}^n c_i^2}$.

4.3.3 Stability implies concentration

The weight sensitivity coefficients give concentration of the learned weight vector:

Corollary 4.7. *Let A be a Hilbert space valued algorithm. Suppose A has weight sensitivity coefficient β_n at sample size n. Then for any $\delta \in (0,1)$, with probability $\geq 1-\delta$ we have*

$$\|\mathsf{A}(S) - \mathbb{E}[\mathsf{A}(S)]\| \leq \sqrt{n}\,\beta_n\left(1 + \sqrt{\frac{1}{2}\log\left(\frac{1}{\delta}\right)}\right)\,.$$

This is a consequence of Theorem 4.6 since $c_i \leq \beta_n$ for $i = 1,\dots,n$, hence $\|c_{1:n}\| \leq \sqrt{n}\,\beta_n$.

Last (not least) we deduce concentration of the weight vector $W = \mathrm{SVM}_\lambda(S)$.

Corollary 4.8. *Let $W = \mathrm{SVM}_\lambda(S)$ be the weight vector output by SVM with regularisation λ (see Eq. (4.1)) based on a random sample $S = Z_{1:n} = (Z_1,\dots,Z_n)$. Suppose* *that the kernel used by SVM is bounded by B. For any $\lambda > 0$, for any $\delta \in (0,1)$, with probability $\geq 1-\delta$ we have*

$$\|W - \mathbb{E}[W]\| \leq \frac{2B}{\lambda\sqrt{n}}\left(1+\sqrt{\frac{1}{2}\log\left(\frac{1}{\delta}\right)}\right).$$

Under these conditions we have hypothesis sensitivity coefficients $\beta_n \leq \frac{2B}{\lambda n}$ (we follow Bousquet and Elisseeff [2002], Example 2 and Lemma 16, adapted to our setting). Then apply Corollary 4.7.

4.3.4 Proof of Lemma 4.5

Let $M_n = f(Z_1,\ldots,Z_n)$ be a function of the independent $\mathcal{Z}$-valued random variables $Z_1,\ldots,Z_n$, where the function $f : \mathcal{Z}^n \to \mathcal{W}$ maps into a separable Hilbert space $\mathcal{W}$. Let us write $M_n - \mathbb{E}[M_n]$ as the telescopic sum[6]

$$M_n - \mathbb{E}[M_n] = D_n + D_{n-1} + \cdots + D_1$$

where

$$D_i = \mathbb{E}[M_n|\mathcal{F}_i] - \mathbb{E}[M_n|\mathcal{F}_{i-1}]$$

and $\mathcal{F}_k = \sigma(Z_1,\ldots,Z_k)$ the σ-algebra generated by the first k examples. Thus

$$\|M_n - \mathbb{E}[M_n]\|^2 = \sum_{i=1}^{n}\|D_i\|^2 + 2\sum_{i<j}\langle D_i, D_j\rangle.$$

We need $\mathbb{E}\|M_n - \mathbb{E}[M_n]\|^2$. Taking the expectation above makes the second sum disappear since for $i<j$ we have

$$\mathbb{E}[\langle D_i, D_j\rangle] = \mathbb{E}\big[\mathbb{E}[\langle D_i, D_j\rangle|\mathcal{F}_i]\big] = \mathbb{E}\big[\langle D_i, \mathbb{E}[D_j|\mathcal{F}_i]\rangle\big]$$

and clearly $\mathbb{E}[D_j|\mathcal{F}_i] = 0$ for $j > i$. Thus we have

$$\mathbb{E}\,\|M_n - \mathbb{E}[M_n]\|^2 = \mathbb{E}\sum_{i=1}^{n}\|D_i\|^2. \tag{4.2}$$

Also recall the notation $\xi_{k:l} = (\xi_k,\ldots,\xi_l)$ for $k<l$. It will be used extensively in what follows.

Let us write the conditional expectations in terms of regular conditional probabilities:

$$\mathbb{E}[f(Z_{1:n})|\mathcal{F}_i] = \int f(Z_{1:i}, z_{i+1:n})\, dP_{Z_{i+1:n}|Z_{1:i}}(z_{i+1:n}|Z_{1:i}).$$

[6]The Doob decomposition: D_i are martingale differences and their sum $M_n - \mathbb{E}[M_n]$ is a martingale.

The random variables are labelled with capitals. The lower case letters are for the variables of integration. We write P_X for the distribution (probability law) of X.

Similarly

$$\begin{aligned}\mathbb{E}[f(Z_{1:n})|\mathcal{F}_{i-1}] &= \int f(Z_{1:i-1}, z_{i:n})\, dP_{Z_{i:n}|Z_{1:i-1}}(z_{i:n}|Z_{1:i-1}) \\ &= \int f(Z_{1:i-1}, z_{i:n})\, dP_{Z_i|Z_{1:i-1}}(z_i|Z_{1:i-1}) \cdot dP_{Z_{i+1:n}|Z_{1:i}}(z_{i+1:n}|Z_{1:i-1}, z_i)\,.\end{aligned}$$

By independence, $P_{Z_{i+1:n}|Z_{1:i}} = P_{Z_{i+1:n}}$ and $P_{Z_i|Z_{1:i-1}} = P_{Z_i}$ (this latter is not really needed in the proof, but shortens the formulae). Hence,

$$\begin{aligned}D_i = \mathbb{E}[f(Z_{1:n})|\mathcal{F}_i] - \mathbb{E}[f(Z_{1:n})|\mathcal{F}_{i-1}] &= \int f(Z_{1:i}, z_{i+1:n})\, dP_{Z_{i+1:n}}(z_{i+1:n}) \\ &\quad - \int f(Z_{1:i-1}, z_{i:n})\, dP_{Z_i}(z_i)\, dP_{Z_{i+1:n}}(z_{i+1:n})\,.\end{aligned}$$

Then D_i is equal to the integral w.r.t. $P_{Z_{i+1:n}}(z_{i+1:n})$ of

$$\int [f(Z_{1:i-1}, Z_i, z_{i+1:n}) - f(Z_{1:i-1}, z_i, z_{i+1:n})]\, dP_{Z_i}(z_i)\,.$$

Notice that only the ith argument of f differs in the integrand. Therefore, if

$$\|f(Z_{1:i-1}, Z_i, z_{i+1:n}) - f(Z_{1:i-1}, z_i, z_{i+1:n})\| \leq c$$

then $\|D_i\| \leq c$. This shows that bounded differences for $f(Z_{1:n})$ implies bounded martingale differences (in norm).

Finally, using Jensen's inequality and Eq. (4.2), and the bounded differences assumption:

$$\mathbb{E}\|M_n - \mathbb{E}[M_n]\| \leq \sqrt{\mathbb{E}\|M_n - \mathbb{E}[M_n]\|^2} \leq \sqrt{\sum_{i=1}^n c_i^2}\,.$$

4.3.5 The average empirical error for randomised linear classifiers with Gaussian randomisation

This section is about the calculation of the empirical term $\hat{L}_S^{01}(Q)$ when Q is a Gaussian distribution with center $w_0 \in \mathbb{R}^p$ and covariance matrix $\sigma^2 I$ where I is the identity $p \times p$. The task is binary classification, and the learning model consists of linear classifiers. The relevant loss function is the zero-one loss $\mathbf{1}(h_w(x) \neq y)$ and the linear classifiers are of the form $h_w(x) = \text{sign}(\langle w, \phi(x)\rangle)$ for $w \in \mathbb{R}^p$, where $\phi : \mathcal{X} \to \mathbb{R}^p$ is a feature map (the weight space here is $\mathcal{W} = \mathbb{R}^p$).

Writing $G_{(w_0,\sigma^2 I)}$ to denote the Gaussian distribution on $\mathbb{R}^p$ with centre at the vector $w_0 \in \mathbb{R}^p$ and covariance matrix $\sigma^2 I \in \mathbb{R}^{p\times p}$ with some $\sigma > 0$, we shall prove that the empirical term $\hat{L}_S^{01}(G_{(w_0,\sigma^2 I)})$ satisfies

$$\hat{L}_S^{01}(G_{(w_0,\sigma^2 I)}) = \int_{\mathcal{X}\times\mathcal{Y}} \tilde{F}\left(\frac{y\, w_0^\top \phi(x)}{\sigma\|\phi(x)\|}\right) d\hat{P}_n(x,y) \tag{4.3}$$

where $\hat{P}_n = \frac{1}{n}\sum_{i=1}^n \delta_{(X_i,Y_i)}$ is the empirical distribution[7] on $\mathcal{X}\times\mathcal{Y}$ associated to the n-sample $S = ((X_1,Y_1),\ldots,(X_n,Y_n))$; and $\tilde{F} = 1-F$ where F is the cumulative distribution function of the standard Gaussian:

$$F(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-u^2/2}\, du\,. \tag{4.4}$$

In this section we write the derivation of Eq. (4.3). To make things more general let $G_{(w_0,\Sigma)}$ be a Gaussian with center $w_0 \in \mathbb{R}^p$ and covariance matrix $\Sigma \in \mathbb{R}^{p\times p}$. But to make notation simpler, in the calculations we identify the feature vectors $\phi(x)$ with the input vectors x (correspondingly, we think that the dimension of x matches the feature dimension p.). The labels are $y \in \{\pm 1\}$. The classifier $h_w(\cdot) = \mathrm{sign}(\langle w,\cdot\rangle)$ is identified with the weight vector w. The zero-one loss of w on example (x,y) can be written as

$$\mathbf{1}(h_w(x) \neq y) = \frac{1-\mathrm{sign}(y\langle w,x\rangle)}{2}\,.$$

Consider the empirical error of a fixed w, namely $\hat{L}_S^{01}(w) = \int_{\mathcal{X}\times\mathcal{Y}} \mathbf{1}(h_w(x)\neq y)\, d\hat{P}_n(x,y)$. Then the average empirical error when choosing a random W according to $G_{(w_0,\Sigma)}$ is:

$$\hat{L}_S^{01}(G_{(w_0,\Sigma)}) = \int_{\mathbb{R}^p} \hat{L}_S^{01}(w)\, dG_{(w_0,\Sigma)}(w)\,.$$

Plugging in the definition of $\hat{L}_S^{01}(w)$ and swapping the order of the integrals and using the above formula for the zero-one loss of the linear classifier, the right hand side is

$$\begin{aligned}
&\int_{\mathbb{R}^p}\int_{\mathcal{X}\times\mathcal{Y}} \mathbf{1}(h_w(x)\neq y)\, d\hat{P}_n(x,y)\, dG_{(w_0,\Sigma)}(w) \\
&\qquad = \int_{\mathcal{X}\times\mathcal{Y}}\int_{\mathbb{R}^p} \mathbf{1}(h_w(x)\neq y)\, dG_{(w_0,\Sigma)}(w)\, d\hat{P}_n(x,y) \\
&\qquad = \int_{\mathcal{X}\times\mathcal{Y}}\int_{\mathbb{R}^p} \frac{1-\mathrm{sign}(y\langle w,x\rangle)}{2}\, dG_{(w_0,\Sigma)}(w)\, d\hat{P}_n(x,y) \\
&\qquad = \int_{\mathcal{X}\times\mathcal{Y}} \frac{1}{2}\left(1 - A(x,y)\right)\, d\hat{P}_n(x,y) \qquad (4.5)
\end{aligned}$$

where for a fixed pair (x,y) we are writing

$$A(x,y) = \int_{\mathbb{R}^p} \mathrm{sign}(y\langle w,x\rangle)\, dG_{(w_0,\Sigma)}(w)\,.$$

Decompose the last integral into two terms:

$$A(x,y) = \int_{y\langle w,x\rangle>0} dG_{(w_0,\Sigma)}(w) - \int_{y\langle w,x\rangle<0} dG_{(w_0,\Sigma)}(w)\,.$$

Notice that, by the symmetry of the Gaussian distribution, we have

$$\int_{y\langle w,x\rangle<0} dG_{(w_0,\Sigma)}(w) = 1 - \int_{y\langle w,x\rangle>0} dG_{(w_0,\Sigma)}(w)\,.$$

[7]The integral with respect to the empirical distribution $\hat{P}_n$ evaluates as a normalised sum (i.e. a sample average): $\int_{\mathcal{X}\times\mathcal{Y}} f(x,y)\, d\hat{P}_n(x,y) = \frac{1}{n}\sum_{i=1}^n f(X_i,Y_i)$.

For the random vector $W \sim G_{(w_0,\Sigma)}$ the linear functional $y\langle W,x\rangle$ has first moment $\mathbb{E}[y\langle W,x\rangle] = y\langle w_0,x\rangle$ and the second moment $\mathbb{E}[(y\langle W,x\rangle)^2] = \|x\|_\Sigma^2 + (\langle w_0,x\rangle)^2$, where recall that $y \in \{\pm 1\}$ and so $y^2 = 1$. Therefore, the functional $y\langle W,x\rangle$ has a 1-dimensional Gaussian distribution with mean $y\langle w_0,x\rangle$ and variance $\|x\|_\Sigma^2 = \langle \Sigma x,x\rangle$. Then

$$\begin{aligned}\int_{y\langle w,x\rangle>0} dG_{(w_0,\Sigma)}(w) &= \mathbb{P}[y\langle W,x\rangle > 0]\\ &= \mathbb{P}\left[\frac{y\langle W,x\rangle - y\langle w_0,x\rangle}{\|x\|_\Sigma} > \frac{-y\langle w_0,x\rangle}{\|x\|_\Sigma}\right]\\ &= \mathbb{P}\left[\mathcal{N}(0,1) > \frac{-y\langle w_0,x\rangle}{\|x\|_\Sigma}\right]\\ &= \mathbb{P}\left[\mathcal{N}(0,1) < \frac{y\langle w_0,x\rangle}{\|x\|_\Sigma}\right]\\ &= F\left(\frac{y\langle w_0,x\rangle}{\|x\|_\Sigma}\right).\end{aligned}$$

Therefore

$$A(x,y) = 2F\left(\frac{y\langle w_0,x\rangle}{\|x\|_\Sigma}\right) - 1$$

and

$$1 - A(x,y) = 2 - 2F\left(\frac{y\langle w_0,x\rangle}{\|x\|_\Sigma}\right) = 2\tilde{F}\left(\frac{y\langle w_0,x\rangle}{\|x\|_\Sigma}\right).$$

Altogether, plugging back into Eq. (4.5) this gives

$$\hat{L}_S^{01}(G_{(w_0,\Sigma)}) = \int_{\mathcal{X}\times\mathcal{Y}} \tilde{F}\left(\frac{y\langle w_0,x\rangle}{\|x\|_\Sigma}\right) d\hat{P}_n(x,y).$$

Notice that using $\Sigma = \sigma^2 I$ and $\phi(x)$ instead of x this gives Eq. (4.3).

REMARK: Langford [2005] used a Q_μ which is $\mathcal{N}(\mu,1)$ along the direction of a vector w, and $\mathcal{N}(0,1)$ in all directions perpendicular to w. To evaluate the empirical error rate of the stochastic classifier, note that such Q_μ is a Gaussian centered at $w_0 = \mu w/\|w\|$ and covariance the identity $p \times p$ matrix, giving the formula

$$\hat{L}_S^{01}(Q_\mu) = \int_{\mathcal{X}\times\mathcal{Y}} \tilde{F}\left(\mu \frac{y\, w^\top \phi(x)}{\|w\|\ \|\phi(x)\|}\right) d\hat{P}_n(x,y).$$

4.4 Gaussian distributions over a Hilbert space?

This section aims to provide a rigorous explanation for Gaussian randomisation in Hilbert spaces, which has been used here and in several previous machine learning works. For instance in the setting of SVM classifiers with feature map $\phi : \mathcal{X} \to \mathcal{H}$, the output is a weight vector that lives in the Hilbert space $\mathcal{H}$. With the Gaussian kernel in mind, we are facing an infinite-dimensional separable $\mathcal{H}$, which upon the choice of an orthonormal basis $\{e_1, e_2, \dots\}$ can be identified with the space[8] $\ell_2 \subset \mathbb{R}^{\mathbb{N}}$ of square summable sequences of real numbers, via the isometric isomorphism $\mathcal{H} \to \ell_2$ that maps the vector $w = \sum_{i=1}^{\infty} w_i e_i \in \mathcal{H}$ to the sequence $(w_1, w_2, \dots) \in \ell_2$. Thus without loss of generality we may regard the feature map as $\phi : \mathcal{X} \to \ell_2 \subset \mathbb{R}^{\mathbb{N}}$.

The PAC-Bayes approach applied to SVMs says that instead of committing to the weight vector $W_n = \mathrm{SVM}(S_n)$ we will randomise by choosing a fresh $W \in \mathcal{H}$ according to some probability distribution on $\mathcal{H}$ for each prediction. Suppose the randomised classifier is to be chosen according to a Gaussian distribution. Although it commonly appears in the literature, it is worth wondering just what is a Gaussian distribution over the space $\mathcal{H} = \ell_2$.

Two possibilities come to mind for the Gaussian random classifier W: (1) according to a Gaussian measure on ℓ_2, say $W \sim \mathcal{N}(\mu, \Sigma)$ with mean $\mu \in \ell_2$ and covariance operator Σ meeting the requirements (positive, trace-class) for this to be a Gaussian measure on ℓ_2; or (2) according to a Gaussian measure on the bigger $\mathbb{R}^{\mathbb{N}}$, e.g. $W \sim \mathcal{N}(\mu, I)$ by which we mean the measure constructed as the product of a sequence $\mathcal{N}(\mu_i, 1)$ of independent real-valued Gaussians with unit variance. These two possibilities are mutually exclusive since the first choice gives a measure on $\mathbb{R}^{\mathbb{N}}$ whose mass is supported on ℓ_2, while the second choice leads to a measure on $\mathbb{R}^{\mathbb{N}}$ supported outside of ℓ_2. A good reference for this topic is Bogachev [1998].

Let us go with the second choice: $\mathcal{N}(0, I) = \bigotimes_{i=}^{\infty} \mathcal{N}(0, 1)$, a 'standard Gaussian' on $\mathbb{R}^{\mathbb{N}}$. This is a legitimate probability measure on $\mathbb{R}^{\mathbb{N}}$ (by Kolmogorov's Extension theorem). But it is supported outside of ℓ_2, so when sampling a $W \in \mathbb{R}^{\mathbb{N}}$ according to this measure, with probability one such W will be outside of our feature space ℓ_2. Then we have to wonder about the meaning of $\langle W, \cdot \rangle$ when W is not in the Hilbert space carrying this inner product.

Let us write $W = (\xi_1, \xi_2, \dots)$ a sequence of i.i.d. standard (real-valued) Gaussian random variables. Let $x = (x_1, x_2, \dots) \in \ell_2$, and consider the formal expression $\langle x, W \rangle = \sum_{i=1}^{\infty} x_i \xi_i$. Notice that

$$\sum_{i=1}^{\infty} \mathbb{E}[|x_i \xi_i|^2] = \sum_{i=1}^{\infty} |x_i|^2 < \infty .$$

Then (see e.g. Bogachev [1998], Theorem 1.1.4) our formal object $\langle x, W \rangle = \sum_{i=1}^{\infty} x_i \xi_i$ is actually well-defined in the sense that the series is convergent almost surely (i.e. with probability one), although as we pointed out such W is outside ℓ_2.

4.4.1 Predicting with the Gaussian random classifier

Let $W_n = \mathrm{SVM}(S_n)$ be the weight vector found by running SVM on the sample S_n. We write it as $W_n = \sum_{i=1}^{n} \alpha_i Y_i \phi(X_i)$. Let $\kappa(\cdot, \cdot)$ be the kernel doing the "kernel trick."

Also as above let W be a Gaussian random vector in $\mathbb{R}^{\mathbb{N}}$, and we write it as

[8] Just to be sure: $\mathbb{R}^{\mathbb{N}}$ stands for the set of all infinite sequences of real numbers.

$W = \sum_{j=1}^{\infty} \xi_j e_j$ with $\xi_1, \xi_2, \ldots$ i.i.d. standard Gaussians. As usual e_j stands for the canonical unit vectors having a 1 in the jth coordinate and zeros elsewhere.

For an input $x \in \mathcal{X}$ with corresponding feature vector $\phi(x) \in \mathcal{H}$, we predict with

$$\langle W_n + W, \phi(x) \rangle = \sum_{i=1}^{n} \alpha_i Y_i \kappa(X_i, x) + \sum_{j=1}^{\infty} \xi_j \langle e_j, \phi(x) \rangle \,.$$

This is well-defined since

$$\sum_{i=1}^{\infty} \mathbb{E}[(\xi_j \langle e_j, \phi(x) \rangle)^2] = \sum_{i=1}^{\infty} (\langle e_j, \phi(x) \rangle)^2 = \|\phi(x)\|^2 \,,$$

so the series $\sum_{j=1}^{\infty} \xi_j \langle e_j, \phi(x) \rangle$ converges almost surely (Bogachev [1998], Theorem 1.1.4).

4.5 Experiments

N.B.: The results of experiments reported by Rivasplata et al. [2018] were based on using the form of the PAC-Bayes-kl bound as presented by Langford [2005, Theorem 5.1] in which the log term in the upper bound is $\log(n+1)$, whereas in our presentation above we used the sharp dependence on n (as per Maurer [2004]) in which the corresponding term is $\log(2\sqrt{n})$. Notice that the difference in the bounds is $\frac{1}{n}[\log(n+1) - \log(2\sqrt{n})]$, which is negligible for large n.

The purpose of the experiments was to explore the strengths and potential weaknesses of our new bound in relation to the previous alternatives, as well as to explore the bound's ability to help model selection. For this, to facilitate comparisons, taking the setup of Parrado-Hernández et al. [2012], we experimented with the five UCI datasets described there. However, we present results for the datasets Pima (PIM) and Ringnorm (RIN) only, as the results on the other datasets mostly followed the results on these and these two datasets are the most extreme in terms of datase size with 768 and 7400 examples, respectively. Similarly, they are significantly different in terms of their input dimensions with feature spaces of dimension 8 and 20, respectively.

Model and data preparation We used an offset-free SVM classifier with a Gaussian RBF kernel $\kappa(x,y) = \exp(-\|x-y\|_2^2/(2\sigma^2))$ with RBF width parameter $\sigma > 0$. The SVM used the so-called standard SVM-C formulation which multiplies the total (hinge) loss by $C > 0$; the conversion to our formulation Eq. (4.1) is given by $C = \frac{1}{\lambda n}$ where n is the number of training examples and $\lambda > 0$ is the penalty factor, as explained in Section 4.2.5. The datasets were split into a training and a test set using the train_test_split method of the scikit-learn package [Pedregosa et al., 2011], keeping 80% of the data for training and 20% for testing.

Model parameters Following the procedure suggested in Chapelle and Zien [2005, Section 2.3.1], we set up a geometric 7×7 grid over the (C, σ)-parameter space where C ranges between $2^{-8}C_0$ and $2^2 C_0$ and σ ranges between $2^{-3}\sigma_0$ and $2^3\sigma_0$, where σ_0 is

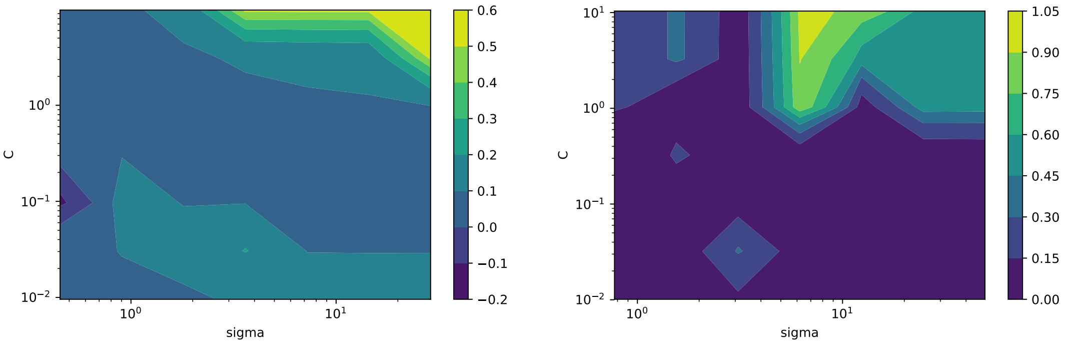

Figure 4.1: Tightness of P@O bound on PIM (left) and RIN (right) shown as the difference between the bound and the test error of the underlying randomised classifier. Smaller values are preferred.

the median of the Euclidean distance between pairs of data points of the training set, and given σ_0, C_0 is obtained as the reciprocal value of the empirical variance of data in feature space underlying the RBF kernel with width σ_0. The grid size was selected for economy of computation. The grid lower and upper bounds for σ were ad-hoc, though they were inspired by the literature, while for the same for C, we enlarged the lower range to focus on the region of the parameter space where the stability-based bounds have a better chance to be effective: In particular, the stability-based bounds grow with C in a linear fashion, with a coefficient that was empirically observed to be close to one.

Computations For each of the (C, σ) pairs on the said grid, we trained an SVM model using a Python implementation of the SMO algorithm of Platt [1999], adjusted to SVMs with no offset (Steinwart and Christmann [2008] argue that "the offset term has neither a known theoretical nor an empirical advantage" for the Gaussian RBF kernel). We then calculated various bounds using the obtained model, as well as the corresponding test error rates (recall that the randomised classifiers' test error is different than the test error of the SVM model that uses no randomisation). The bounds compared were the two mentioned hinge-loss based bounds: The bound by Liu et al. [2017] and that of Bousquet and Elisseeff [2002]. In addition we calculated the P@O bound and (our) P@EW bound. When these latter were calculated we optimised the randomisation variance parameter σ^2_{noise} by minimising error estimate obtained from the respective bound (the binary KL divergence $\text{kl}(\hat{L}_S(Q)||L(Q))$ was inverted numerically). Further details of this can be found in Section 4.5.1.

Results As explained earlier our primary interest is to explore the strengths and weaknesses of the various bounds. In particular, we are interested in their tightness, as well as their ability to support model selection. As the qualitative results were insensitive to the split, results for a single "random" (arbitrary) split are shown only.

Tightness: The hinge loss based bounds gave trivial bounds over almost all pairs of (C, σ). Upon investigating this we found that this is because the hinge loss takes

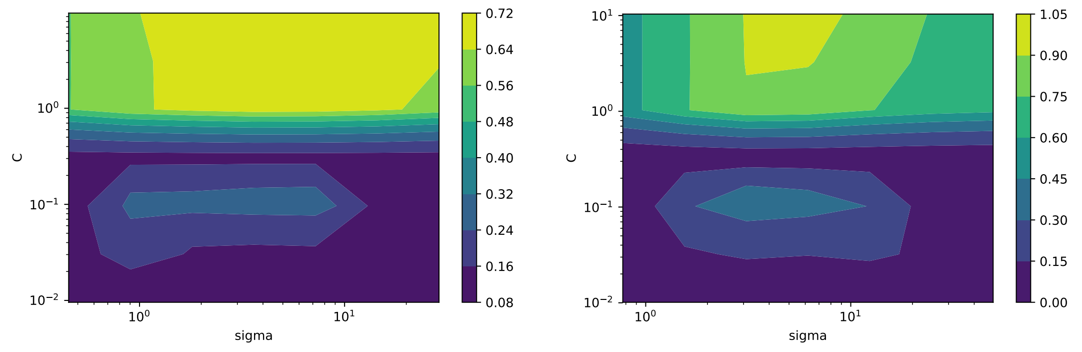

Figure 4.2: Tightness of P@EW bound (the bound derived here) on PIM (left) and RIN (right) shown as the difference between the bound and the test error of the underlying randomised classifier. Smaller values are preferred.

much larger values than the training error rate unless C takes large values (cf. Fig. 4.3 in Section 4.5.2). However, for large values of C, both of the bounds are vacuous. In general, the stability based bounds (Liu et al. [2017], Bousquet and Elisseeff [2002] and our bound) are sensitive to large values of C. Fig. 4.1 shows the difference between the P@O bound and the test error of the underlying respective randomised classifiers as a function of (C, σ) while Fig. 4.2 shows the difference between the P@EW bound and the test error of the underlying randomised classifier. (Figs. 4.7 and 4.9 in Section 4.5.2 show the test errors for these classifiers, while Figs. 4.6 and 4.8 shows the bound.) The meticulous reader may worry about that it appears that on the smaller dataset, PIM, the difference shown for P@O is sometimes negative. As it turns out this is due to the randomness of the test error: Once we add a confidence correction that accounts for the randomness of the small test set ($n_{\text{test}} = 154$) this difference disappears once we correct the test error for this. From the figures the most obvious difference between the bounds is that the P@EW bound is sensitive to the value of C and it becomes loose for larger values of C. This is expected: As noted earlier, stability based bounds, which P@EW is an instance of, are sensitive to C. The P@O bound shows a weaker dependence on C if any. In Section 4.5.2 we show the advantage (or disadvantage) of the P@EW bound over the P@O bound on Fig. 4.10. From this figure we can see that on PIM, P@EW is to be preferred almost uniformly for small values of C ($C \leq 0.5$), while on RIN, the advantage of P@EW is limited both for smaller values of C and a certain range of the RBF width. Two comments are in order in connection to this: *(i)* We find it remarkable that a stability-based bound can be competitive with the P@O bound, which is known as one of the best bounds available. *(ii)* While comparing bounds is interesting for learning about their qualities, the bounds can be used together (e.g., at the price of an extra union bound).

Model selection: To evaluate a bound's capability in helping model selection it is worth comparing the correlation between the bound and test error of the underlying classifiers. By comparing Figs. 4.6 and 4.7 with Figs. 4.8 and 4.9 it appears that perhaps the behavior of the P@EW bound (at least for small values of C) follows more closely the behavior of the corresponding test error surface. This is particularly visible on RIN, where the P@EW bound seems to be able to pick better values both for C and σ, which lead to a much smaller test error (around 0.12) than what one can obtain by using the P@O bound.

4.5.1 Details of optimising σ^2_{noise}

This optimisation is "free" for the P@O bound as the bound is uniform over σ^2_{noise}. In the P@EW bound we adjusted the failure probability δ to accommodate the multiple evaluations during the optimisation by replacing it with $\delta/(\tau(\tau+1))$, where τ is the number of times the P@EW bound is evaluated by the optimisation procedure. A standard union bound argument shows that the adjustment to δ makes the resulting bound hold with probability $1-\delta$ regardless the value of τ. The SLSQP method implemented in SCIPY was used as an optimiser, with an extra outer loop that searched for a suitable initialisation, as SLSQP is a gradient based method and the P@O bound can be quite "flat". The same problem did not appear for the P@EW bound. The attentive reader may be concerned that if τ gets large values, we, in a way, are optimising the "wrong bound". To check whether this is a possibility, we also computed the "union bound penalty" for decreasing δ by the factor $\tau(\tau+1)$ as the difference between the (invalid) bound where δ is unchanged and the bound where δ is decreased and found that the penalty was generally orders of magnitudes smaller than the risk estimate. Nevertheless, this may be a problem when the risk to be estimated is very small, which we think is not very common in practice.

4.5.2 Additional figures

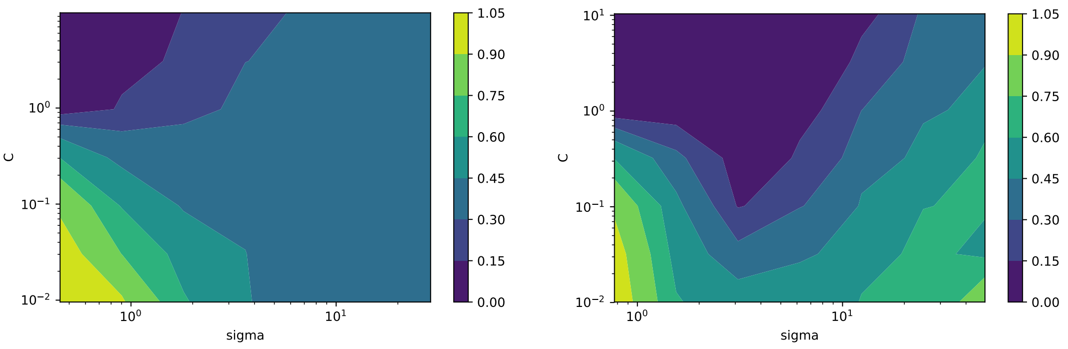

Figure 4.3: Hinge loss on PIM (left) and RIN (right). For large values of C, the hinge loss is reasonable, but this is not the case for small values.

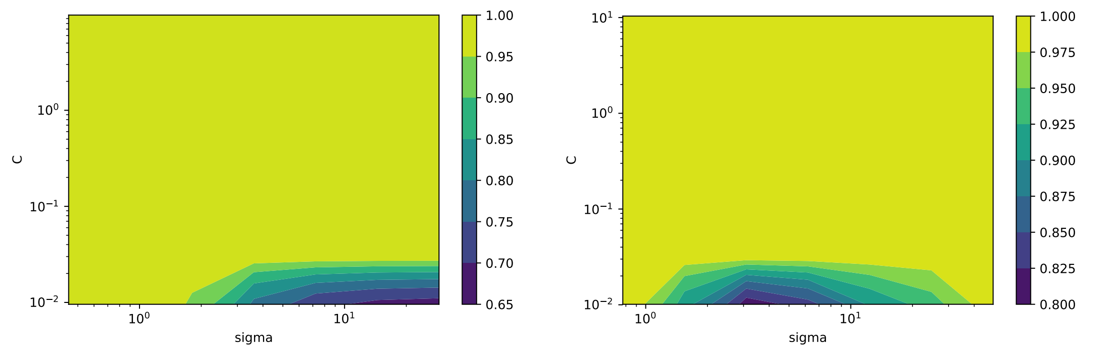

Figure 4.4: The bound of Liu et al. [2017] on PIM (left) and RIN (right). The bound is almost always vacuous for reasons described in the text.

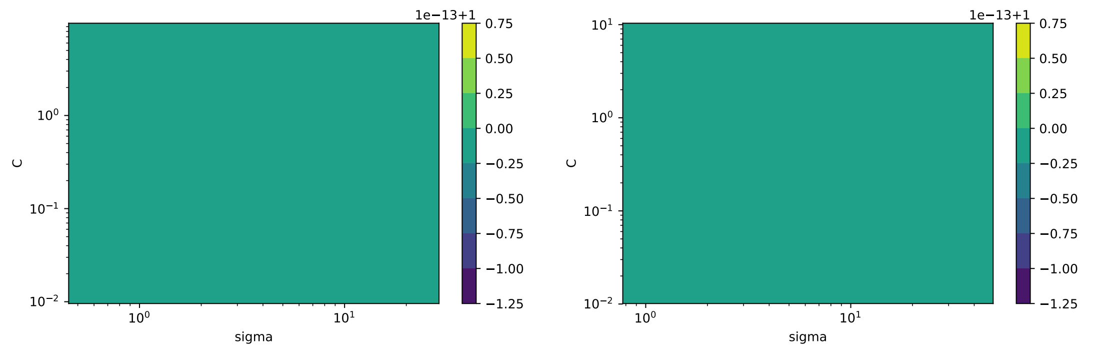

Figure 4.5: The bound of Bousquet and Elisseeff [2002] on PIM (left) and RIN (right). The bound is almost always vacuous for reasons described in the text.

4.6 Conclusion and Future Work

This chapter is based on my paper Rivasplata et al. [2018]. This work developed a stability-based PAC-Bayes bound for randomised classifiers. We proceeded by investigating the stability of the weight vector learned by a Hilbert space valued algorithm, a special case being SVMs. We applied our main theorem (Theorem 4.2) to SVMs, leading to our P@EW bound, and we compared it to other stability-based bounds and to a previously known PAC-Bayes bound. The main finding is that perhaps P@EW is the first nontrivial bound that uses (uniform) weight stability.

As discussed in this chapter (Cf. discussion after Definition 4.1) there are various related notions of algorithmic stability. The notion of 'weight stability' used in this chapter is close in spirit to the notion of 'uniform stability' [Bousquet and Elisseeff, 2002] since it considers the worst-case sensitivity to changing a single input data point. While we considered changes as seen by the norm in the weight space, other stability notions consider changes in the learned classifier, as seen by some distance function on the hypothesis space, or changes in the loss functional applied to the classifiers. Then it is natural to ask what stability notion is preferable for a given learning setting.

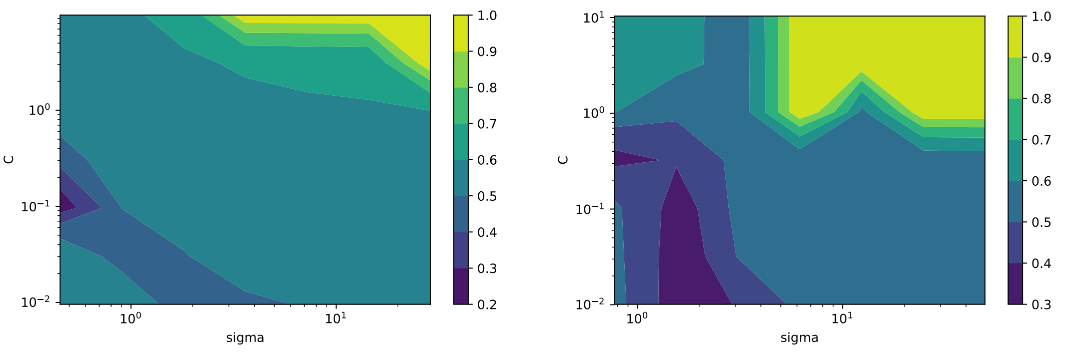

Figure 4.6: The P@O bound on PIM (left) and RIN (right).

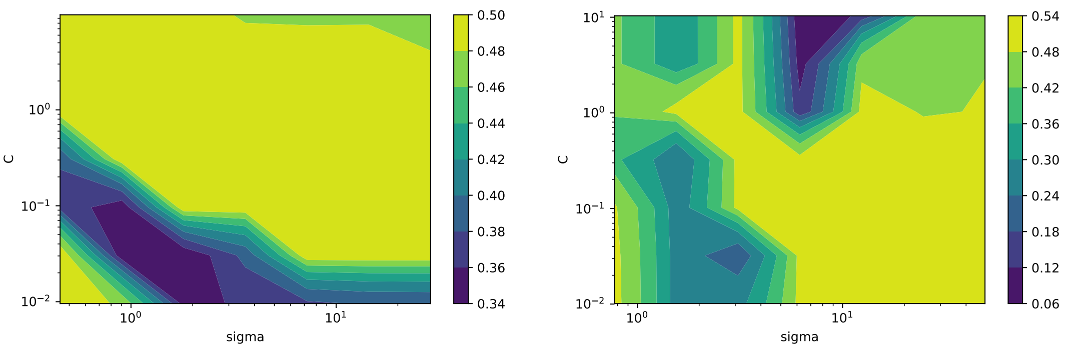

Figure 4.7: Test error of the randomised classifiers underlying the P@O bound on PIM (left) and RIN (right).

The stability-based bound presented here depends on several problem-related quantities, such as the randomisation noise parameter σ, the regularisation factor λ, and the number n of training data points. A natural question is of course whether this dependence is optimal and in what sense. While some comparisons to other bounds were discussed, such comparisons only can answer how one bound compares to another bound with respect to each of the involved quantities. A more interesting question would be to study the optimal dependence on the quantities, and discuss a given bound in terms of how far it is from being optimal with respect to each quantity.

N.B.: Hanneke and Kontorovich [2019] studied the optimality of risk bounds for SVMs, but note the publication date which is posterior to Rivasplata et al. [2018].

The choice of the same variance parameter σ^2 for both the prior and posterior in our PAC-Bayes bound has raised some questions. It is clear that this choice is convenient for computation of the KL term in the bound. An important question to address is about the impact of this choice on the trade-off with the empirical error term $\hat{L}_S^{01}(Q)$, since the latter also depends on σ, as shown in Section 4.3.5. Another question which was left unanswered is about the choice of covariance structure in the Gaussian distributions, and perhaps even the choice of distributions that could give the best results.

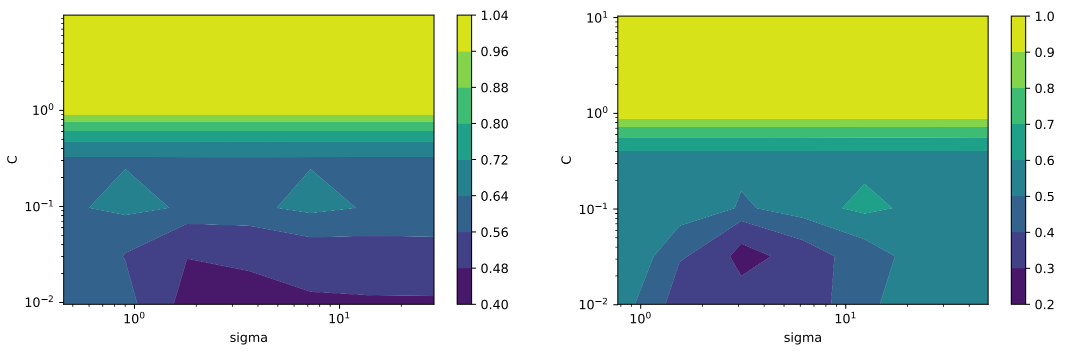

Figure 4.8: The P@EW bound on PIM (left) and RIN (right).

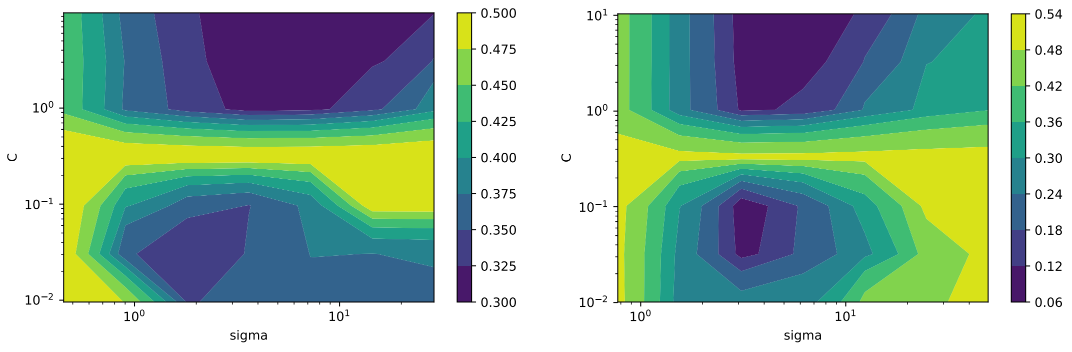

Figure 4.9: Test error of the randomised classifiers underlying the P@EW bound on PIM (left) and RIN (right).

As already mentioned at the end of Section 4.1, one may argue that neural networks (NNs) also fall in the category of algorithms that output a weight vector, and therefore the notion of weight stability (Definition 4.1) may be applied in this setting. Then, a very interesting future research direction, with deep learning in view, is to figure out the sensitivity coefficients of NN's trained by Stochastic Gradient Descent. Then our main theorem could be applied to provide non-vacuous bounds for the performance of NN's. Arguably, the stability of neural networks is a non-trivial problem which may require the combination of theoretical and computational approaches.

This work reported in this chapter can be viewed as contributing a certification strategy for randomised SVM classifiers, where the risk certificate is computed post-training on the same data that was used to train the classifier. This work built on the previous work of Parrado-Hernández et al. [2012] who studied learning and certification strategies for randomised SVM classifiers. It may be connected also to a line of research that aims to develop 'self-bounding algorithms' (Freund [1998], Langford and Blum [2003]) in the sense that besides producing a predictor the algorithm also creates a performance certificate based on the available data. This line of thought inspired further works of mine with collaborators (Cf. Rivasplata et al. [2019], Pérez-Ortiz et al. [2021b])

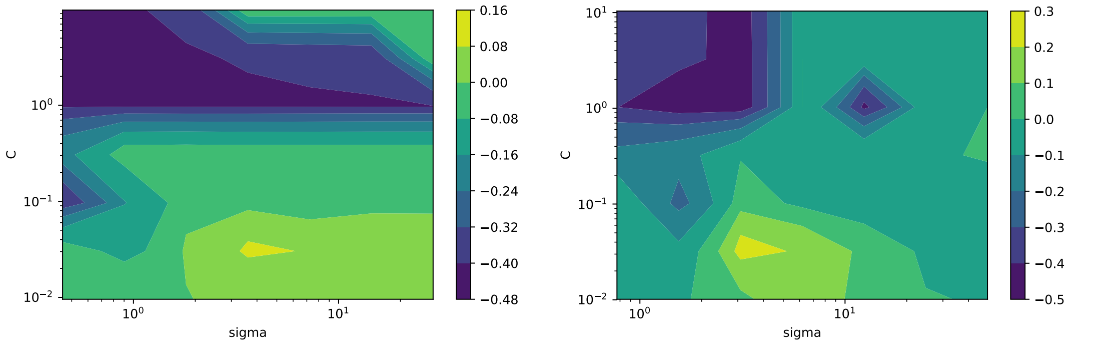

Figure 4.10: Advantage of the P@EW bound to the P@O bound on PIM (left) and RIN (right): The figure shows the difference between the P@O bound and the P@EW bound. Where this is positive, P@EW is to be preferred, while where it is negative, P@O is to be preferred.

and indeed inspired the learning and certification strategies based on PAC-Bayes bounds that are described in Chapter 5 which deals with training and certification of neural network classifiers, and the promising paradigm of self-certified learning.

Chapter 5

Self-certified learning with randomised neural networks and PAC-Bayes with backprop

N.B.: The content of this chapter is borrowed from my paper Pérez-Ortiz et al. [2021b]. Minor modifications were made to make the content consistent with other chapters of this thesis, and some updates were made to improve the clarity.

In a probabilistic neural network, the result of the training process is a distribution over network weights, rather than simply fixed weights. Several prediction schemes can be devised based on a probability distribution over weights. For instance, one may use a randomised predictor, where each prediction is done by randomly sampling the weights from the data-dependent distribution obtained as the result of the training process. Another prediction rule consists of predicting always with the mean of the learned distribution. Yet another prediction rule is the ensemble predictor based on integrating the predictions of all possible parameter settings, weighted according to the learned distribution.

In this chapter we experiment with probabilistic neural networks from a PAC-Bayes approach. We name 'PAC-Bayes with Backprop' (PBB) the family of (probabilistic) neural network training methods derived from PAC-Bayes bounds and optimised through stochastic gradient descent. The work reported here is the result of our empirical studies undertaken to investigate three PBB training objectives. For reference, they are the functions f_{quad}, f_{lambda} and f_{classic}, shown respectively in Eq. (5.2), Eq. (5.3) and Eq. (5.4) below. These objectives are based on PAC-Bayes bounds with similar names, which are relaxations of the PAC-Bayes relative entropy bound [Langford and Seeger, 2001], also known as the PAC-Bayes-kl bound in the literature. The classic PAC-Bayes bound, from which f_{classic} is derived, is that of McAllester [1999], but we use it with the improved dependence on the number of training patterns as clarified by Maurer [2004]. The PAC-Bayes-lambda bound is that of Thiemann et al. [2017]. The

PAC-Bayes-quadratic bound, from which f_{quad} is derived, was originally introduced in the preprint Rivasplata et al. [2019]. Importantly, our work shows tightness of the numerical certificates on the error of the randomised classifiers generated by these training methods. In each case, the computed certificate is valid on unseen examples (from the same data distribution as the training data), and is evaluated using (part of) the data set that was used to learn the predictor for which the certificate is valid. These properties make our work a first example of *self-certified learning*, which proposes to use the whole data set for learning a predictor and certifying its risk on unseen data, without the need for data splitting protocols both for testing and model selection.

This line of research owes credit to previous works that have trained a probability distribution over neural network weights by minimising a PAC-Bayes bound, or used a PAC-Bayes bound to give risk certificates for trained neural networks. Langford and Caruana [2001] developed a method to train a distribution over neural network weights by randomising the weights with Gaussian noise (adjusted in a data-dependent way via a sensitivity analysis), and computed an upper bound on the error using the PAC-Bayes-kl bound.[1] They also suggested that PAC-Bayes bounds might be fruitful for computing non-vacuous generalisation bounds for neural nets. Dziugaite and Roy [2017] used a training objective (essentially equivalent to f_{classic}) which was based on a relaxation of the PAC-Bayes-kl bound. They optimised this objective using stochastic gradient descent (SGD), and computed a confidence bound on the error of the randomised classifier following the same approach that Langford and Caruana [2001] used to compute their error bound. Dziugaite and Roy [2018a] developed a two-stage method, which in the first stage trains a prior mean by empirical risk minimisation via stochastic gradient Langevin dynamics (SGLD) [Welling and Teh, 2011], and in the second stage re-uses the same data used for the prior in order to train a posterior Gibbs distribution over weights; they also evaluate a relaxation of the PAC-Bayes-kl bound, based on ideas from differential privacy [Dwork et al., 2015a,b], which accounts for the data re-use.

In this chapter we report experiments on MINIST and CIFAR-10 with the three training objectives mentioned above. We used by default the randomised predictor scheme (also called the ‘stochastic predictor’ in the PAC-Bayes literature), justified by the fact that PAC-Bayes bounds give high-confidence guarantees on the expected loss of the randomised predictor. Since training is based on a surrogate loss function, optimising a PBB objective gives a high-confidence guarantee on the randomised predictor’s risk under the surrogate loss. Accordingly, to obtain guarantees that are valid

[1]Numerical inversion of the PAC-Bayes-kl bound (we explain this in Section 5.3) gives a certificate (upper bound) on the risk of the randomised predictor, in terms of its empirical error and other quantities. The empirical error term is evaluated indirectly by Monte Carlo sampling, and a bound on the tail of the Monte Carlo evaluation [Langford and Caruana, 2001, Theorem 2.5] is combined with the PAC-Bayes-kl bound to give a numerical risk certificate that holds with high probability over the random draw of data and Monte Carlo samples.

for the classification error (i.e. the zero-one loss), we separately evaluate a confidence bound for the error of the randomised predictor, based on part of the data that was used to learn it (following the procedure that was used by Langford and Caruana [2001] and Dziugaite and Roy [2017]). For comparison we also report test set error for the randomised predictor, and for the other two predictor schemes described above, namely, the posterior mean and the ensemble predictors.

Our work took inspiration from Blundell et al. [2015], whose results showed that randomised weights achieve competitive test set errors; and from Dziugaite and Roy [2017, 2018a], whose results gave randomised neural network classifiers with reasonable test set errors and, more importantly, non-vacuous risk bound values. Our experiments show that PBB training objectives can (a) achieve competitive test set errors (e.g. comparable to Blundell et al. [2015] and empirical risk minimisation), while also (b) deliver risk certificates with reasonably tight values. Our results show as well a significant improvement over those of Dziugaite and Roy [2017, 2018a]: we further close the gap between the numerical risk certificate (bound value) and the risk estimate (test set error rate). As we argue below, this improvement comes from the tightness of the PAC-Bayes bounds we used, which is established analytically and corroborated by our experiments on MNIST and CIFAR-10 with deep fully connected networks and convolutional neural networks.

Regarding the tightness, the training objective of Dziugaite and Roy [2017] (which in our notation takes essentially the form of f_{classic} shown in Eq. (5.4) below) has the disadvantage of being sub-optimal in the regime of small losses. This is because their objective is a relaxation of the PAC-Bayes-kl bound via an inequality that is loose in this regime. The looseness was the price paid for having a computable objective. Note that small losses is precisely the regime of interest in neural network training (although the true loss being small is data set and architecture dependent). By contrast, our proposed training objectives (f_{quad} and f_{lambda} in Eq. (5.2) and Eq. (5.3) below) are based on relaxing the PAC-Bayes-kl bound by an inequality that is tighter in this same regime of small losses, which is one of the reasons explaining our tighter risk certificates in MNIST (not for CIFAR-10, which could be explained by the large empirical loss obtained at the end of the optimisation). Interestingly, our own re-implementation of f_{classic} also gave improved results compared to the results of Dziugaite and Roy, which suggests that besides the training objectives we used, also the training strategies we used are responsible for the improvements.

A clear advantage of PAC-Bayes with Backprop (PBB) methods is being an instance of self-certified[2] learning: We say that a learning method is *self-certified* if it uses all the available data in order to simultaneously output a predictor and a risk certificate that is reasonably tight and valid at population level, i.e. the certificate is

[2]The name *self-certified learning* was originally introduced in my paper Pérez-Ortiz et al. [2021b].

valid for any unseen data from the same distribution that generated the training data. As mentioned in Chapter 1, self-certified learning is a re-branding of self-bounding learning [Freund, 1998], and this chapter makes a case for self-certified learning using PAC-Bayes bounds. For clarity, we emphasise that "using all the available data to simultaneously produce a predictor and a risk certificate" should be understood in the sense that all the data is used by the learning strategy to output a predictor, and therefore the risk certificate must be computed on the same (or part of the) data used for learning the predictor. The value of self-certified learning algorithms is in the possibility of using of all the available data to achieve both goals (learning a predictor and certifying its risk) simultaneously, thus making efficient use of the available data. This chapter shows learning and certification strategies based on PAC-Bayes bounds which could lead us to self-certified learning. Thus, when training probabilistic neural nets by PBB methods the output is not just a predictor but simultaneously a *tight risk certificate* that guarantees the quality of predictions on unseen examples. Note that risk certificates *per se* will not impress until their reported values match or closely follow the classification error rates evaluated on a test set, so that the risk certificate is informative of the out-of-sample error. This is where our work makes a significant contribution, since our PBB training methods lead to risk certificates for neural nets with much tighter values than previous works in the literature. Once again, the solution found by our learning procedure comes together with a high-confidence guarantee that certifies its risk under the surrogate training loss, and to obtain a high-confidence guarantee for the classification error (zero-one loss) we evaluate *post training* a risk bound. A more ambitious goal would be to establish calibration[3] of the surrogate cross-entropy loss, so then minimising it would guarantee minimal classification error.

We would like to highlight the elegant simplicity of the methods presented here: Our results are achieved i) with priors learnt through empirical risk minimisation of the surrogate loss on a subset of the data set (which does not overlap with the data used for computing the risk certificate for the randomised predictor, thus in line with classical PAC-Bayes priors) and ii) via classical SGD optimisation. In contrast, Dziugaite and Roy [2018a] trained a special type of data-dependent PAC-Bayes prior on the whole data set using SGLD optimisation. They justified this procedure arguing that the limit distribution of SGLD satisfies the differential privacy property (but a finite-time guarantee was missing), and relaxed the PAC-Bayes-kl bound with a correction term based on the concept of max-information[4] to account for using the same data to train the prior mean and to evaluate the bound. Furthermore, our methods do not involve tampering with the training objective, as opposed to Blundell et al. [2015], who used a "KL attenuating trick" by inserting a tunable parameter as a factor of the Kullback-

[3]This is akin to results on calibration of the surrogate hinge loss, cf. Steinwart and Christmann [2008].
[4]Dwork et al. [2015a,b] proposed this concept in the context of adaptive data analysis.

Leibler (KL) divergence in their objective. Our work highlights the point that it is worthwhile studying simple methods, not just to understand their scope or for the sake of having a more controlled experimental setup, but also to more accurately assess the real value added by the 'extras' of the more complex methods.

Contributions of this work:

1. We rigorously study and illustrate 'PAC-Bayes with Backprop' (PBB), a generic strategy to derive (probabilistic) neural network training methods from PAC-Bayes bounds.

2. We propose —and experiment with— two new PBB training objectives: one derived from the PAC-Bayes-quadratic bound of Rivasplata et al. [2019], and one derived from the PAC-Bayes-lambda bound of Thiemann et al. [2017].

3. We also re-implement the training objective based on the classic PAC-Bayes bound that was used by Dziugaite and Roy, for the sake of comparing our training objectives and training strategy, both with respect to test set accuracy and risk certificates obtained.

4. We connect PAC-Bayes with Backprop (PBB) methods to the Bayes-by-Backprop (BBB) method of Blundell et al. [2015] which is inspired by Bayesian learning and achieved competitive test set accuracy. Unlike BBB, our training methods require less heuristics and also provide a risk certificate; not just an error estimate based on a test set.

5. We demonstrate via experimental results that PBB methods might be able to achieve self-certified learning with nontrivial certificates: obtaining competitive test set errors and computing non-vacuous bounds with much tighter values than previous works.

Broader Context. Deep learning is a vibrant research area. The success of deep neural network models in several tasks has motivated many works that study their optimisation and generalisation properties, some of the collective knowledge is condensed in a few recent sources such as Montavon et al. [2012], Goodfellow et al. [2016], Aggarwal [2018]. Some works focus on experimenting with methods to train neural networks, others aim at generating knowledge and understanding about these fascinating learning systems. In this chapter we intend to contribute both ways. We focus on supervised classification problems through probabilistic neural networks, and we experiment with training objectives that are principled and consist of interpretable quantities. Furthermore, our work puts an emphasis on certifying the quality of predictions beyond a specific data set.

Note that known neural network training methods range from those that have been developed based mainly on heuristics to those derived from sound principles.

Bayesian learning, for instance, offers principled approaches for learning data-dependent distributions over network weights (see e.g. Buntine and Weigend, 1991, Neal, 1992, MacKay, 1992, Barber and Bishop, 1997), hence probabilistic neural nets arise naturally in this approach. Bayesian neural networks continue to be developed, with notable recent contributions e.g. by Hernández-Lobato and Adams [2015], Martens and Grosse [2015], Blundell et al. [2015], Gal and Ghahramani [2016], Louizos and Welling [2016], Ritter et al. [2018], Khan and Lin [2017], Osawa et al. [2019], Maddox et al. [2019], among others. Our work is complementary of Bayesian learning in the sense that our methods also offer principled training objectives for learning probabilistic neural networks. However, there are differences between the PAC-Bayes and Bayesian learning approaches that are important to keep in mind (see our discussions in Chapter 1 and Section 5.1). It is worth mentioning also that some works have pointed out the resemblance between PAC-Bayes bounds and the evidence lower bound (ELBO) of variational Bayesian inference (Alquier et al., 2016, Achille and Soatto, 2018, Thakur et al., 2019, Pitas, 2020). An insightful connection between Bayesian inference and the frequentist PAC-Bayes approach was discussed by Germain et al. [2016a].

As we pointed out before, we are not the first to train a probabilistic neural network by minimising a PAC-Bayes bound, or to use a PAC-Bayes bound to give risk certificates for randomised neural nets. We already mentioned Langford and Caruana [2001] and Dziugaite and Roy [2017, 2018a], whose works have directly influenced ours.[5] Next, we comment on some other works that connect PAC-Bayes with neural networks. London [2017] approached the generalisation of neural networks by a stability-based PAC-Bayes analysis, and proposed an adaptive sampling algorithm for SGD that optimises its distribution over training instances using multiplicative weight updates. Neyshabur et al. [2017, 2018] examined the connection between some specifically defined complexity measures and generalisation, the part related to our work is that they specialised a form of the classic PAC-Bayes bound and used Gaussian noise on network weights to give generalisation bounds for probabilistic neural networks based on the norms of the weights. Zhou et al. [2019] compressed trained networks by pruning weights to a given target sparsity, and gave generalisation guarantees on the compressed networks, which were based on randomising predictors according to their 'description length' and a specialisation of a PAC-Bayes bound of Catoni [2007].

We would like to point out that the present work builds on Rivasplata et al. [2019]. In the meantime, more works have appeared that connect neural networks with PAC-Bayes bounds in various settings: Letarte et al. [2019], Viallard et al. [2019], Lan et al. [2020], possibly among others. We do not elaborate on these works as they deal with significantly different settings than ours. The recent work by Dziugaite et al. [2021] is

[5]Note that Langford and Caruana [2001] and Dziugaite and Roy [2017] called them *stochastic* neural networks, arguably because the distribution over weights moves during training.

more closely related to ours in that they investigate the use of data to learn a PAC-Bayes prior.

Chapter Layout. The rest of the chapter is organised as follows. Section 5.1 discusses the connection between our work and that of Blundell et al. [2015]. Section 5.2 presents the training objectives derived from the PAC-Bayes bounds that were discussed in Chapter 2. The technical Section 5.3 describes the binary KL inversion strategy and how it is used for risk certification. Section 5.4 presents our experimental results. Finally, Section 5.5 concludes the chapter and discusses future research directions.

5.1 The Bayes by Backprop (BBB) Objective

The ‘Bayes by backprop’ (BBB) method of Blundell et al. [2015] is inspired by a variational Bayes argument [Jordan et al., 1998, Fox and Roberts, 2012], where the idea is to learn a distribution over weights that approximates the Bayesian posterior distribution. Choosing a p-dimensional Gaussian $Q_\theta = \mu + \sigma\mathcal{N}(0,I)$, parametrised by $\theta = (\mu,\sigma) \in \mathbb{R}^p \times \mathbb{R}^p$, the optimum parameters are those that minimise $\mathrm{KL}(Q_\theta \| P(\cdot|S))$, i.e. the KL divergence from Q_θ and the Bayesian posterior $P(\cdot|S)$. By a simple calculation, and using the Bayes rule, one can extract:

$$\mathrm{KL}(Q_\theta \| P(\cdot|S)) = \int_{\mathcal{W}} -\log(P(S|w))Q_\theta(dw) + \mathrm{KL}(Q_\theta \| Q^0)\,,$$

where Q^0 stands here for the Bayesian prior distribution. Thus, minimising $\mathrm{KL}(Q_\theta \| P(\cdot|S))$ is equivalent to minimising the right-hand side, which presents a sum of a data-dependent term (the expected negative log-likelihood) and a prior-dependent term ($\mathrm{KL}(Q_\theta \| Q^0)$). This optimisation problem is analogous to that of minimising a PAC-Bayes bound, since the latter balances a fit-to-data term (the empirical loss) and a fit-to-prior term (the KL).

There is indeed a close connection between the PAC-Bayes and Bayesian learning approaches, as has been pointed out by the work of Germain et al. [2016a], when the loss function is the negative log-likelihood. Beyond this special case, the PAC-Bayes learning approach offers more flexibility in design choices, such as the choice of loss functions and the choice of distributions. This is because the PAC-Bayes ‘prior’ is a reference distribution and the PAC-Bayes ‘posterior’ does not need to be derived from a prior by a likelihood update factor. This is a crucial difference with Bayesian learning, and one that makes the PAC-Bayes framework a lot more flexible in the choice of distributions over parameters, even compared to generalised Bayesian approaches [Bissiri et al., 2016].

As we mentioned before, the training objective proposed by Blundell et al. [2015] is inspired by the variational Bayesian argument outlined above, in particular, in our notation the training objective they proposed and experimented with is as follows:

$$f_{\mathrm{bbb}}(Q) = \hat{L}_S(Q) + \eta \, \frac{\mathrm{KL}(Q \| Q^0)}{n} \,. \tag{5.1}$$

The scaling factor, $\eta > 0$, is introduced in a heuristic manner to make the method more flexible, while the variational Bayes argument gives (5.1) with $\eta = 1$. When η is treated as a tuning parameter, the method can be interpreted as searching in "KL balls" centered at Q^0 of various radii. Thus, the KL term then plays the role of penalising the complexity of the model space searched. Blundell et al. [2015] proposed to optimise this objective (for a fixed η) using stochastic gradient descent (SGD), which randomises over both mini-batches and the weights, and used the pathwise gradient estimate [Price, 1958]. The resulting gradient-calculation procedure can be seen to be only at most twice as expensive as standard backpropagation —hence the name of their method. The hyperparameter $\eta > 0$ is chosen using a validation set, which is also often used to select the best performing model among those that were produced during the course of running SGD (as opposed to using the model obtained when the optimisation procedure finishes).

5.2 Towards Practical PAC-Bayes with Backprop (PBB) Methods

The essential idea of 'PAC-Bayes with Backprop' (PBB) is to train a probabilistic neural network by minimising a PAC-Bayes bound via stochastic gradient descent (SGD) optimisation. Here we present two training objectives, derived from Eq. (2.10) and Eq. (2.11) respectively, in the context of *classification problems* when the loss is the zero-one loss or a surrogate loss. These objectives are used here for the first time to train probabilistic neural networks. We also discuss the training objective derived from Eq. (2.9) for comparison purposes. Besides the training objectives themselves, in this section we also discuss the various details of the optimisation strategy.

To optimise the weights of neural networks the standard idea is to use a form of stochastic gradient descent, which requires the ability to efficiently calculate gradients of the objective to be optimised. When the loss is the zero-one loss, the training loss viewed as a function of the weights, $w \mapsto \hat{L}_S^{01}(w)$, is piecewise constant, which makes simple gradient-based methods fail (since the gradient, whenever it exists, is zero). As such, it is customary to replace the zero-one loss with a smoother "surrogate loss" that plays well with gradient-based optimisation. In particular, the standard loss used on multiclass classification problems is the cross-entropy loss, $\ell^{\text{x-e}} : \mathbb{R}^k \times [k] \to \mathbb{R}$ defined by $\ell^{\text{x-e}}(u,y) = -\log(\sigma(u)_y)$ where $u \in \mathbb{R}^k$, $y \in [k] = \{1,\ldots,k\}$ and $\sigma : \mathbb{R}^k \to [0,1]^k$ is the soft-max function defined by $\sigma(u)_i = \exp(u_i)/\sum_j \exp(u_j)$. This choice can be justified on the grounds that $\ell^{\text{x-e}}(u,y)$ gives an upper bound on the probability of mistake when the label is chosen at random from the distribution produced by applying soft-max on u (e.g., $u =$ the output of the last linear layer of a neural network). Indeed, owning to the inequality $\log(x) \leq x - 1$, which is valid for any $x > 0$, given any $u \in \mathbb{R}^k$ and $y \in [k]$, if $Y \sim \sigma(u)$ then $\mathbb{E}[\mathbb{I}\{Y \neq y\}] = \mathbb{P}(Y \neq y) = 1 - \sigma(u)_y \leq \ell^{\text{x-e}}(u,y)$.

We thus also propose to replace the zero-one loss with the cross-entropy loss in either Eq. (2.10) or Eq. (2.11), leading to the objectives

$$f_{\text{quad}}(Q) = \left(\sqrt{\hat{L}_S^{\text{x-e}}(Q) + \frac{\mathrm{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{2n}} + \sqrt{\frac{\mathrm{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{2n}} \right)^2 \tag{5.2}$$

and

$$f_{\text{lambda}}(Q,\lambda) = \frac{\hat{L}_S^{\text{x-e}}(Q)}{1-\lambda/2} + \frac{\mathrm{KL}(Q\|Q^0) + \log(2\sqrt{n}/\delta)}{n\lambda(1-\lambda/2)}\,. \tag{5.3}$$

For comparison, the training objective derived from Eq. (2.9) takes the following form:

$$f_{\text{classic}}(Q) = \hat{L}_S^{\text{x-e}}(Q) + \sqrt{\frac{\mathrm{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{2n}}\,. \tag{5.4}$$

Here, $\hat{L}_S^{\text{x-e}}(w) = \frac{1}{n}\sum_{i=1}^n \tilde{\ell}_1^{\text{x-e}}(h_w(X_i),Y_i)$ is the empirical error rate under the 'bounded' version of cross-entropy loss, namely the loss $\tilde{\ell}_1^{\text{x-e}}$ described next, and $h_w : \mathcal{X} \to \mathbb{R}^k$ denotes the function implemented by the neural network that uses weights w.

The next issue to address is that the cross-entropy loss is unbounded, while the PAC-Bayes bounds that inspired these objectives require a bounded loss with range $[0,1]$. This is fixed by enforcing an upper bound on the cross-entropy loss by lower-bounding the network probabilities by a value $p_{\min} > 0$ [Dziugaite and Roy, 2018a]. This is achieved by replacing σ in the definition of $\ell^{\text{x-e}}$ by $\tilde{\sigma}(u)_y = \max(\sigma(u)_y, p_{\min})$. This adjustment gives a 'bounded cross-entropy' loss function $\tilde{\ell}^{\text{x-e}}(u,y) = -\log(\tilde{\sigma}(u)_y)$ with range between 0 and $\log(1/p_{\min})$. Finally, re-scaling by $1/\log(1/p_{\min})$ gives a loss function $\tilde{\ell}_1^{\text{x-e}}$ with range [0,1] ready to be used in the PAC-Bayes bounds and training objectives discussed here. The latter ($\tilde{\ell}_1^{\text{x-e}}$) is used as the surrogate loss for training in all our experiments with f_{quad}, f_{lambda}, and f_{classic}.

5.2.1 Optimisation Problem

Optimisation of f_{quad} and f_{classic} (Eq. (5.2) and Eq. (5.4)) entails minimising over Q only, while optimisation of f_{lambda} (Eq. (5.3)) is done by alternating minimisation with respect to Q and λ, similar to the procedure that was used by Thiemann et al. [2017] in their experiments with SVMs. By choosing Q appropriately, in either case we use the pathwise gradient estimator [Price, 1958, Jankowiak and Obermeyer, 2018, Mohamed et al., 2020] as done by Blundell et al. [2015].

In particular, assuming that the parametrisation $Q = Q_\theta$ with $\theta \in \mathbb{R}^q$ is such that $h_W(\cdot)$ with $W \sim Q_\theta$ ($W \in \mathbb{R}^p$) has the same distribution as $h_{f_\theta(V)}(\cdot)$ where $V \in \mathbb{R}^{p'}$ is drawn at random from a *fixed* distribution P_V and $f_\theta : \mathbb{R}^{p'} \to \mathbb{R}^p$ is a smooth map, an unbiased estimate of the gradient of the loss-map $\theta \mapsto Q_\theta[\ell(h_\bullet(x),y)]$ at some θ can be obtained by drawing $V \sim P_V$ and calculating $\frac{\partial}{\partial\theta}\ell(h_{f_\theta(V)}(x),y)$, thereby reducing the efficient computation of the gradient to the application of the backpropagation algorithm on the map $\theta \mapsto \ell(h_{f_\theta(v)}(x),y)$ at $v = V$. Indeed, assuming that the partial derivatives are integrable, which needs to be verified on a case-by-case basis, we have $\frac{\partial}{\partial\theta}\int Q_\theta(dw)\ell(h_w(x),y) = \frac{\partial}{\partial\theta}\int P_V(dv)\ell(h_{f_\theta(v)}(x),y) = \int P_V(dv)\frac{\partial}{\partial\theta}\ell(h_{f_\theta(v)}(x),y)$. The details of this reparametrisation strategy are given e.g. by Ruiz et al. [2016].

In our experiments the PAC-Bayes posterior is parametrised as a diagonal Gaussian distribution over weight space $\mathcal{W} = \mathbb{R}^p$. Then a sample of the posterior can be obtained by sampling a standard Gaussian, scaling each coordinate by a corresponding standard deviation from the vector $\sigma = (\sigma_i)_{i\in[p]} \in \mathbb{R}^p$, and shifting by a mean vector $\mu \in \mathbb{R}^p$. We parametrise σ coordinatewise as $\sigma = \log(1+\exp(\rho))$ so σ is always non-negative. Following Blundell et al. [2015], the reparametrisation we use is $W = \mu + \sigma \odot V$, where the symbol $\odot$ denotes coordinatewise product (i.e. $(\sigma \odot V)_i = \sigma_i V_i$ for each i), with appropriate distribution (Gauss or Laplace) for each coordinate of V, although other reparametrisations are possible [Osawa et al., 2019, Khan and Lin, 2017]. Gradient updates are with respect to vectors μ and ρ, as can be seen in Algorithm 1. Note that after sampling the weights, the gradients for the mean and standard deviation are shared and are exactly the gradients found by the usual backpropagation algorithm on a neural network. More specifically, to learn both the mean and the standard deviation we simply calculate the usual gradients found by backpropagation, and then scale and shift them as done by Blundell et al. [2015]. Note that Algorithm 1 shows the procedure for optimising f_{quad} with Gaussian noise. The procedure with Laplace noise is similar. The procedure for f_{classic} is similar. The procedure for f_{lambda} would be very similar except that f_{lambda} has the additional parameter λ.

As discussed in Chapter 2, the PAC-Bayes bounds from which these training objectives were derived are relaxations of the PAC-Bayes-kl bound (Eq. (2.8)). We refer the reader to Eq. (2.5) for the definition of the binary KL divergence, denoted $\text{kl}(\cdot\|\cdot)$. It was explained that f_{classic} is a relaxation of PAC-Bayes-kl bound obtained by Pinsker's inequality:

$$\text{kl}(\hat{p}\|p) \geq 2(p-\hat{p})^2 \quad \text{for } \hat{p}, p \in (0,1)\,. \tag{5.5}$$

On the other hand, f_{quad} and f_{lambda} are relaxations of the PAC-Bayes-kl bound obtained using the refined version Pinsker's inequality:

$$\text{kl}(\hat{p}\|p) \geq \frac{(p-\hat{p})^2}{2p} \quad \text{for } \hat{p}, p \in (0,1),\ \hat{p} < p\,. \tag{5.6}$$

Algorithm 1 PAC-Bayes with Backprop (PBB)

```
Require:
    $\mu_0$                                  ▷ Prior center parameters
    $\rho_0$                                 ▷ Prior scale hyper-parameter
    $Z_{1:n}$                                ▷ Training examples (inputs + labels)
    $\delta \in (0,1)$                       ▷ Confidence parameter
    $\alpha \in (0,1),\ T$                   ▷ Learning rate; Number of iterations
Ensure: Optimal $\mu \in \mathbb{R}^p$, $\rho \in \mathbb{R}^p$   ▷ Posterior centers and scales
 1: procedure PB_QUAD_GAUSS
 2:     $\mu \leftarrow \mu_0$               ▷ Set initial posterior center to prior center
 3:     $\rho \leftarrow \rho_0$             ▷ Set initial posterior scale to prior scale
 4:     for $t \leftarrow 1:T$ do            ▷ Run SGD for T iterations.
 5:         Sample $V \sim \mathcal{N}(0,I)$
 6:         $W = \mu + \log(1+\exp(\rho)) \odot V$
 7:         $f = f_{\text{quad}}(Z_{1:n}, W, \mu, \rho, \mu_0, \rho_0, \delta)$
 8:         SGD gradient step using $\begin{bmatrix}\nabla_\mu f\\ \nabla_\rho f\end{bmatrix}$,   $\nabla_\mu f = \frac{\partial f}{\partial W} + \frac{\partial f}{\partial \mu}$,   $\nabla_\rho f = \frac{\partial f}{\partial W} \cdot \frac{V}{1+\exp(-\rho)} + \frac{\partial f}{\partial \rho}$
 9:     end for
10:     return $\mu, \rho$
11: end procedure
```

One can compare these two inequalities, to find regime of $p, \hat{p}$ in which one is better than the other. The result of the comparison is that Eq. (5.5) is tighter whenever $p > 1/4$, and Eq. (5.6) is tighter whenever $p < 1/4$. They match if $p = 1/4$. This comparison might be relevant for understanding the differences—in terms of tightness of risk certificates but also test performance—between the solutions found by these training objectives.

5.2.2 The Choice of the PAC-Bayes Prior Distribution

We experiment both with priors centered at randomly initialised weights and priors learnt by empirical risk minimisation using the surrogate loss on a subset of the data set which is independent of the subset used to compute the risk certificate. Note that all n training data are used by the learning algorithm (n_0 examples used to build the prior, n to learn the posterior and $n - n_0$ to evaluate the risk certificate). This is to avoid needing differentially private arguments to justify learning the prior [Dziugaite and Roy, 2018a]. Since the posterior is initialised to the prior, the learnt prior translates to the posterior being initialised to a large region centered at the empirical risk minimiser. Similar approaches for building data-dependent priors have been considered before in the PAC-Bayesian literature [Ambroladze et al., 2007, Parrado-Hernández et al., 2012].

For our PAC-Bayes prior over weights we experiment with Gaussian and with Laplace distributions. In each case, the PAC-Bayes posterior learnt by PBB is of the same kind (Gaussian or Laplace) as the prior. Next we give formulas for computing the KL term in our training objectives for each of these distributions.

5.2.2.1 Formulas for the KL: Laplace and Gaussian

The Laplace density with mean parameter $\mu \in \mathbb{R}$ and with variance $b > 0$ is the following:

$$p(x) = (2b)^{-1} \exp\big(-\frac{|x-\mu|}{b}\big) .$$

The KL divergence for two Laplace distributions is

$$\mathrm{KL}(\mathrm{Lap}(\mu_1, b_1) \| \mathrm{Lap}(\mu_0, b_0)) = \log(\frac{b_0}{b_1}) + \frac{|\mu_1 - \mu_0|}{b_0} + \frac{b_1}{b_0} e^{-|\mu_1-\mu_0|/b_1} - 1 . \qquad (5.7)$$

For comparison, recall that the Gaussian density with mean parameter $\mu \in \mathbb{R}$ and variance $b > 0$ has the following form:

$$p(x) = (2\pi b)^{-1/2} \exp\big(-\frac{(x-\mu)^2}{2b}\big) .$$

The KL divergence for two Gaussian distributions is

$$\mathrm{KL}(\mathrm{Gauss}(\mu_1, b_1) \| \mathrm{Gauss}(\mu_0, b_0)) = \frac{1}{2}\Big(\log(\frac{b_0}{b_1}) + \frac{(\mu_1 - \mu_0)^2}{b_0} + \frac{b_1}{b_0} - 1\Big) . \qquad (5.8)$$

The formulas (5.7) and (5.8) above are for the KL divergence between one-dimensional Laplace or Gaussian distributions. It is straightforward to extend them to multi-dimensional product distributions, corresponding to random vectors with independent components, as in this case the KL is equal to the sum of the KL divergences of the components. Note that formula (5.7) could seem to pose a challenge during gradient-based optimisation due to the presence of the absolute value. However, auto-differentiation packages solve this by calculating left or right derivatives which are defined in every case.

5.3 Computing Risk Certificates

After optimising the distribution over network weights through the previously presented training objectives, we compute a risk certificate on the error of the stochastic predictor, following the procedure of Langford and Caruana [2001]. This uses the PAC-Bayes-kl bound (Eq. (2.8)). First we describe how to invert the binary KL (defined in Eq. (2.5)) with respect to its second argument. For $x \in [0,1]$ and $b \in [0,\infty)$, we define:

$$f^\star(x,b) = \sup\{y \in [x,1] \,:\, \mathrm{kl}(x\|y) \leq b\} .$$

This is easily seen to be well-defined. Furthermore, the crucial property that we rely on is that $\mathrm{kl}(x\|y) \leq b$ holds precisely when $y \leq f^\star(x,b)$.

Note that the function $f^\star$ provides a way for computing an upper bound on $L(Q)$ based on the PAC-Bayes-kl bound (given in Eq. (2.8)): For any confidence $\delta \in (0,1)$, with probability at least $1-\delta$ over size-n random samples S we have:

$$L(Q) \leq f^\star\Big(\hat{L}_S(Q), \frac{\mathrm{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{n}\Big) .$$

At this point, as noted by Langford and Caruana [2001], the difficulty is evaluating $\hat{L}_S(Q)$. This quantity is not computable. Since $f^\star$ is a monotonically increasing function of its first argument (when fixing the second argument), it suffices to upper-bound $\hat{L}_S(Q)$.

5.3.1 Estimating the Empirical Loss via Monte Carlo Sampling

In fact, $f^\star$ is also used to estimate the empirical term $\hat{L}_S(Q)$ by random weight sampling: If $W_1,\dots,W_m \sim Q$ are i.i.d. and $\hat{Q}_m = \sum_{j=1}^m \delta_{W_j}$ is the empirical distribution, then for any $\delta' \in (0,1)$, with probability at least $1-\delta'$ we have $\mathrm{kl}(\hat{L}_S(\hat{Q}_m)\|\hat{L}_S(Q)) \leq m^{-1}\log(2/\delta')$ (see Langford and Caruana, 2001, Theorem 2.5), hence by the inversion formula:

$$\hat{L}_S(Q) \leq f^\star\Big(\hat{L}_S(\hat{Q}_m), \frac{1}{m}\log(\frac{2}{\delta'})\Big)\,.$$

This expression can be applied to upper-bound $\hat{L}_S^{01}(Q)$ or $\hat{L}_S^{\text{x-e}}(Q)$ by setting the underlying loss function to be the 01 (classification) loss or the cross-entropy loss, respectively. This estimation is valid with high probability (of at least $1-\delta'$) over random weight samples.

The latter expression also can be combined with any of the PAC-Bayes bounds presented in Chapter 2 to upper-bound the loss $L(Q_S)$ by a computable expression. Just to illustrate, combining with the classical PAC-Bayes bound we would get the following risk bound:

$$L(Q_S) \leq f^\star\Big(\hat{L}_S(\hat{Q}_m), \frac{1}{m}\log(\frac{2}{\delta'})\Big) + \sqrt{\frac{\mathrm{KL}(Q_S\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{2n}}\,,$$

which holds with probability at least $1-\delta-\delta'$ over random size-n data samples S and size-m weight samples $W_1,\dots,W_m \sim Q_S$. The parameter $\delta \in (0,1)$ quantifies the confidence over random data samples, and $\delta' \in (0,1)$ the confidence over random weight samples.

As we said before, our evaluation of risk certificates was based on the PAC-Bayes-kl bound. The next subsection fills the details.

5.3.2 Final Expression for Evaluating the Risk Certificate

In our experiments we evaluate the risk certificates (risk upper bounds) corresponding to the cross-entropy loss ($\ell^{\text{x-e}}$) and the 0-1 loss (ℓ^{01}), respectively, computed using the PAC-Bayes-kl bound and Monte Carlo weight sampling. For any $\delta,\delta' \in (0,1)$, with probability at least $1-\delta-\delta'$ over random size-n data samples S and size-m weight samples $W_1,\dots,W_m \sim Q_S$ we have:

$$L(Q) \leq f^\star\bigg(f^\star\Big(\hat{L}_S(\hat{Q}_m), \frac{1}{m}\log(\frac{2}{\delta'})\Big), \frac{\mathrm{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{n}\bigg)\,.$$

In our experiments we used a numerical implementation of the kl inversion $f^\star$ and the upper bound just shown to evaluate risk certificates for the stochastic predictors corresponding to the distributions over weights obtained by our training methods.

5.4 Experimental Results

We performed a series of experiments on MNIST and CIFAR-10 to thoroughly investigate the training objectives presented before with regards to their ability to give self-certified predictors. Specifically, we empirically evaluate the two proposed training objectives f_{quad} and f_{lambda} of Eq. (5.2) and Eq. (5.3), and compare these to f_{classic} of Eq. (5.4) and f_{bbb} of Eq. (5.1). When possible, we also compare to empirical risk minimisation (f_{erm}) with dropout. In all experiments, training objectives are compared under the same conditions, i.e. network architecture, weight initialisation, the prior distribution over weights, and optimisation algorithm (vanilla SGD with momentum). The code for our experiments is publicly available[6] in PyTorch.

5.4.1 Choice of Distribution over Weights

We studied Gaussian and Laplace distributions over the model weights. The PAC-Bayes posterior distribution Q is learned by optimising a PBB training objective, and is of the same kind as the PAC-Bayes prior (Gaussian or Laplace) in each case.

We also tested in our experiments both data-free random priors (with randomness in the initialisation of the weights) and data-dependent priors. In both cases, the center parameters μ_0 of the prior were initialised randomly from a truncated centered Gaussian distribution with standard deviation set to $1/\sqrt{n_{\text{in}}}$, where n_{in} is the dimension of the inputs to a particular layer, truncating at ± 2 standard deviations. The main difference between our data-free and data-dependent priors is that, after initialisation, the center parameters of data-dependent priors are optimised through ERM on a subset of the training data (50% if not indicated otherwise), while we simply use the initial random weights in the case of data-free priors. The prior scale parameters ρ_0 are set to the constant scale hyper-parameter. The posterior Q is always initialised at the prior (both center and scale parameters). This means that the posterior center μ is initialised at the empirical risk minimiser in the case of data-dependent priors, and to the initial random weights in the case of data-free priors. We find in our experiments that the prior can be over-fitted easily. To avoid this, we use dropout during the learning process (exclusive to learning the prior, not the posterior).

5.4.2 Experimental Setup

All risk certificates were computed using the PAC-Bayes-kl inequality, as explained in Section 5.3, with $\delta = 0.025$ and $\delta' = 0.01$ and $m = 150.000$ Monte Carlo model samples, as done by Dziugaite and Roy [2017]. The same confidence δ was used in all the PBB training objectives (f_{quad}, f_{lambda}, f_{classic}). Input data was standardised.

[6]Code available at `https://github.com/mperezortiz/PBB`

5.4.2.1 Hyperparameter selection

For all experiments we performed a grid search over all hyper-parameters and selected the run with the best risk certificate on 0-1 error[7] (evaluated as explained in Section 5.3). We elaborate more on the use of PAC-Bayes bounds for model selection in the next subsection. The hyperparameters are the following: learning rate and momentum for learning the prior, learning date and momentum for the posterior, the prior distribution scale, and the dropout rate for learning the prior.

We did a grid sweep over the prior distribution scale hyper-parameter (i.e. standard deviation σ_0) with values in $[0.1, 0.05, 0.04, 0.03, 0.02, 0.01, 0.005]$. We observed that higher variance values lead to instability during training and lower variance does not explore the weight space. For the SGD with momentum optimiser for learning the posterior we performed a grid sweep over learning rate in $[1\text{e}-3, 5\text{e}-3, 1\text{e}-2]$ and momentum in $[0.95, 0.99]$. We found that learning rates higher than $1\text{e}-2$ caused divergence in training and learning rates lower than $5\text{e}-3$ converged slowly. We also found that the best optimiser hyper-parameters for building the data-dependent prior differ from those selected for optimising the posterior. Because of this, we also performed a grid sweep over the learning rate and momentum used for learning the data-dependent prior (testing the same values as before). The dropout rate used for learning the prior was selected from $[0.0, 0.05, 0.1, 0.2, 0.3]$. All training objectives derived from PAC-Bayes bounds used the 'bounded cross-entropy' function as surrogate loss during training, for which we enforced boundedness by restricting the minimum probability (see Section 5.2). We observed that the value $p_{\text{min}} = 1\text{e}-5$ performed well. Values higher than $1\text{e}-2$ distorts the input to loss function and leads to higher training loss. The lambda value in f_{lambda} was initialised to 1.0 (as done by Thiemann et al., 2017) and optimised using alternate minimisation using SGD with momentum, using the same choice of learning rate and momentum as for the posterior optimisation. Notice that f_{bbb} requires an additional sweep over a KL trade-off coefficient, which was done with values in $[1\text{e}-5, 1\text{e}-4, \ldots, 1\text{e}-1]$, see Blundell et al. [2015].

For ERM, we used the same range for optimising the learning rate, momentum and dropout rate. However, given that in this case we do not have a risk certificate we need to set aside some data for validation and hyper-parameter tuning. We set 4% of the data as validation in MNIST (2400 examples) and 5% in the case of CIFAR-10 (2500 examples). At this time we have not done model selection with a risk bound for this ERM point estimator model, since to the best of our knowledge those bounds are notoriously vacuous and we are not aware of any empirical evidence that they could be

[7]Note that if we use a total of C hyperparameter combinations, the union bound correction would add no more than $\log(C)/30000$ to the PAC-Bayes-kl upper bound. Even with say $C = 42\text{M}$ (forty two million), the value of our risk certificates, computed via kl inversion, will not be impacted significantly. The reader can be assured that we used much less than 42M hyperparameter combinations.

used for model selection.

5.4.2.2 Predictors and metrics reported

For all methods, we compare three different prediction strategies using the final model weights: i) stochastic predictor, randomly sampling fresh model weights for each test example; ii) deterministic predictor, using exclusively the posterior mean; iii) ensemble predictor, as done by Blundell et al. [2015], in which majority voting is used with the predictions of a number of model weight samples, in our case 100. We report the test cross-entropy loss (x-e) and 0-1 error of these predictors. We also report a series of metrics at the end of training (train empirical risk using cross-entropy $\hat{L}_S^{\text{x-e}}(Q)$ and 0-1 error $\hat{L}_S^{01}(Q)$ and KL divergence between posterior and prior) and the risk certificate (obtained via PAC-Bayes-kl inversion) for the stochastic predictor ($\ell^{\text{x-e}}$ for cross-entropy loss and ℓ^{01} for 0-1 loss).

5.4.2.3 Architectures

For MNIST, we tested both a fully connected neural network (FCN) with 3 layers (excluding the 'input layer') and 600 units per hidden layer, as well as a convolutional neural network (CNN) with 4 layers (two convolutional followed by two fully connected). For the latter, we learn a distribution over the convolutional kernels and the weight matrix. We trained our models using the standard MNIST data set split of 60000 training and 10000 test examples. For CIFAR-10, we tested three convolutional architectures: one with a total of 9 layers with learnable parameters and the other two with 13 and 15 layers; and we used the standard data set split of 50000 training and 10000 test examples. ReLU activations were used in each hidden layer for both data sets. Both for learning the posterior and the prior, we ran the training for 100 epochs (however we observed that methods converged around 70). We used a training batch size of 250 for all the experiments.

5.4.3 Hyper-parameter and Architecture Search through PAC-Bayes Bounds

We show now that PAC-Bayes bounds can be used not only as training objectives to guide the optimisation algorithm but also for model selection. Specifically, Figure 5.1 compares the PAC-Bayes-kl bound for cross-entropy and 0-1 losses (x-axis) to the test 0-1 error for the stochastic predictor (y-axis, top row) and deterministic predictor (y-axis, bottom row) for more than 600 runs from the hyper-parameter grid search performed for f_{quad} with a CNN architecture and a data-dependent Gaussian prior on MNIST. We do a grid search over 6 hyper-parameters: prior scale, dropout rate, and the learning rate and momentum both for learning the prior and the posterior. To depict a larger range of performance values (thus avoiding only showing the risk and performance for relatively accurate classifiers) we use here a reduced training set for these experiments (i.e. 10%

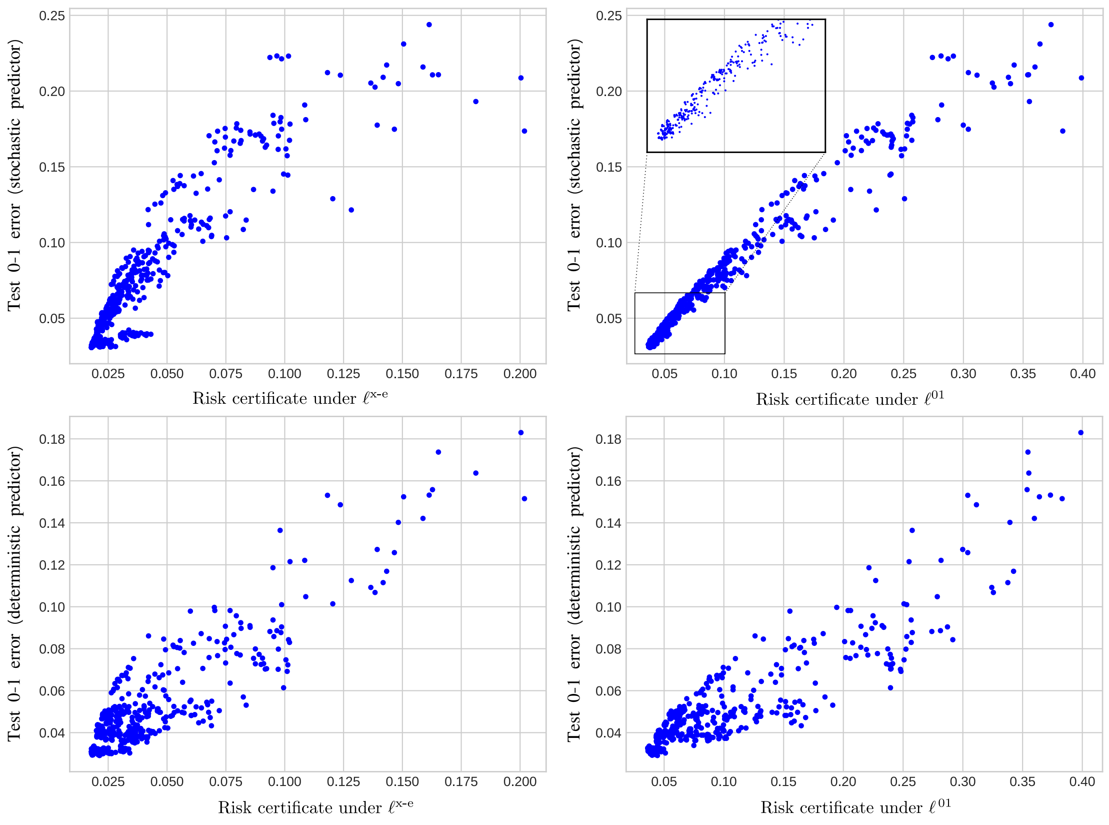

Figure 5.1: Model selection results from more than 600 runs with different hyper-parameters. We use a reduced subset of MNIST for these experiments (10% of training data). The architecture used is a CNN, with Gaussian distributions over weights and data-dependent PAC-Bayes priors. The horizontal axes in the plots show the values of the risk certificate for the stochastic predictor, computed by inversion of the PAC-Bayes-kl bound, under $\ell^{\text{x-e}}$ (left) and ℓ^{01} (right). The vertical axes show the test set error rates for the stochastic (top) and deterministic (bottom) predictors.

of training data from MNIST). The test set is maintained. The results show a clear positive correlation between the risk certificate and test set 0-1 error of the stochastic predictor, especially for the risk certificate of the 0-1 error, as expected. The results are obviously not as positive for the test 0-1 error of the deterministic predictor (since the bound is on the stochastic predictor), but there still exist a linear trend. While the plots also show heteroskedasticity (there is a noticeable increase of variability towards the right side of the x-axis) the crucial observation is that for small error values the corresponding values of the risk certificate are reasonable stable. It is worth keeping in mind, however, that bounds generally get weaker with higher error values.

Figure 5.2 shows a different experiment regarding model selection using MNIST and a fully connected architecture. In this case, we fix the hyperparameters and run several versions of the network with different number of layers and neurons per layer. All the networks are trained in the exact same way using f_{quad}. The linear trend between the risk certificate under ℓ^{01} and the test 0-1 error further validates the usefulness of the risk certificate under ℓ^{01} for model selection.

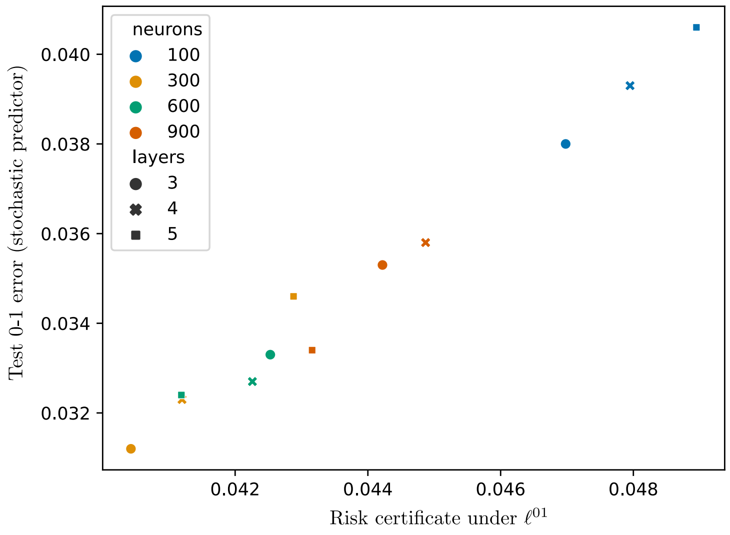

Figure 5.2: Risk certificate under ℓ^{01} vs test 0-1 error on MNIST for a set of fully connected architectures (varying the number of layers and number of neurons per layer).

Motivated by the results shown in Figure 5.1 and Figure 5.2, where it is shown that the bound could potentially be used for model selection, we use the risk certificate with ℓ^{01} (evaluated as explained in Section 5.3) for hyper-parameter tuning in all our subsequent experiments. Note that the advantage in this case is that our approach obviates the need of a held-out set of examples for hyper-parameter tuning.

5.4.4 Comparison of Different Training Objectives and Priors

We first present a comparison of the four considered training objectives on MNIST using Gaussian distributions over weights. Table 5.1 shows the results for the two architectures previously described for MNIST (FCN and CNN) and both data-free and data-dependent priors (referred to as Rand.Init. and Learnt, respectively). We also include the results obtained by standard ERM using the cross-entropy loss, for which part of the table can not be completed (e.g. risk certificates). The last column of the table shows the test set 0-1 error of the prior mean deterministic predictor (column named Prior). We also report the test set performance for the stochastic predictor (Stch. pred.), the posterior mean deterministic predictor (Det. pred.) and the ensemble predictor (Ens pred.). For all the reported results and tables, we highlight the best risk certificate and stochastic test set error in bold face and the second best is highlighted in italics.

An important note is that we used the risk certificates for model selection for all training objectives, including f_{bbb} (with the sole exception of f_{erm}, for which we used a validation set due to the reasons discussed in Section 5.4.2.1). The KL trade-off coefficient included in f_{bbb} [Blundell et al., 2015] relaxes the importance given to the prior in the optimisation, but obviously not in the computation of the risk certificate,

Setup			Risk cert. &		Train metrics			Stch. pred.		Det. pred.		Ens. pred.		Prior
Arch.	Prior	Obj.	$\ell^{\text{x-e}}$	ℓ^{01}	KL/n	$\hat{L}^{\text{x-e}}_S(Q)$	$\hat{L}^{01}_S(Q)$	x-e	01 err.	x-e	01 err.	x-e	01 err.	01 err.
FCN	Rand.Init. (Gaussian)	f_{quad}	.2033	**.3155**	.1383	.0277	.0951	.0268	.0921	.0137	.0558	.0007	.0572	.8792
		f_{lambda}	.2326	*.3275*	.1856	.0218	.0742	.0211	*.0732*	.0077	.0429	.0004	.0448	.8792
		f_{classic}	.1749	.3304	.0810	.0433	.1531	.0407	.1411	.0204	.0851	.0009	.0868	.8792
		f_{bbb}	.5163	.5516	.6857	.0066	.0235	.0088	**.0293**	.0038	.0172	.0003	.0178	.8792
	Learnt (Gaussian)	f_{quad}	.0146	**.0279**	.0010	.0092	.0204	.0084	.0202	.0032	.0186	.0002	.0189	.0202
		f_{lambda}	.0201	.0354	.0054	.0073	.0178	.0082	*.0196*	.0071	.0185	.0001	.0185	.0202
		f_{classic}	.0141	*.0284*	.0001	.0115	.0247	.0101	.0230	.0089	.0189	.0002	.0191	.0202
		f_{bbb}	.0788	.0968	.0704	.0025	.0090	.0063	**.0179**	.0066	.0153	.0001	.0153	.0202
	-	f_{erm}	-	-	-	.0004	.0007	-	-	.0101	.0152	-	-	-
CNN	Rand.Init. (Gaussian)	f_{quad}	.1453	**.2165**	.1039	.0157	.0535	.0143	.0513	.0062	.0257	.0003	.0261	.9478
		f_{lambda}	.1583	*.2202*	.1256	.0126	.0430	.0109	*.0397*	.0056	.0207	.0003	.0211	.9478
		f_{classic}	.1260	.2277	.0622	.0273	.0932	.0253	.0869	.0111	.0425	.0006	.0421	.9478
		f_{bbb}	.3400	.3645	.3948	.0034	.0120	.0039	**.0154**	.0016	.0088	.0001	.0092	.9478
	Learnt (Gaussian)	f_{quad}	.0078	**.0155**	.0001	.0058	.0127	.0045	**.0104**	.0003	.0105	.0001	.0104	.0104
		f_{lambda}	.0095	.0186	.0010	.0051	.0123	.0044	*.0106*	.0047	.0098	.0000	.0100	.0104
		f_{classic}	.0083	*.0166*	.0000	.0064	.0139	.0049	.0123	.0048	.0103	.0001	.0103	.0104
		f_{bbb}	.0447	.0538	.0398	.0012	.0042	.0040	**.0104**	.0043	.0082	.0002	.0082	.0104
	-	f_{erm}	-	-	-	.0003	.0004	-	-	.0081	.0092	-	-	-

Table 5.1: Training and test set metrics on MNIST using Gaussian distributions over weights. The table includes two architectures (FCN and CNN), two kinds of PAC-Bayes priors (a data-free prior centered at the randomly initialised weights, and a data-dependent prior learnt on a subset of the data set) and four training objectives. For the stochastic predictor, the best risk certificate and test set error are highlighted in bold face, and second best are highlighted in italics.

which in practice means that larger KL attenuating coefficients will lead to worse risk certificates. Because of this, in all cases, the model selection strategy chose the lowest value (namely, 0.1) for the KL attenuating coefficient for f_{bbb}, meaning there are cases in which f_{bbb} obtained better test set performance than the ones we report in this table, but much looser risk certificates. We present more experiments on this in the next subsection where we experiment with the KL attenuating trick.

The findings from our experiments on MNIST, reported in Table 5.1 and Figure 5.3, are as follows: i) f_{quad} achieves consistently the best risk certificates for 0-1 error (see ℓ^{01}) in all experiments, providing as well better test performance than f_{classic}, as observed when comparing the 0-1 loss of the stochastic predictors. ii) Based on the results of the stochastic predictor, f_{lambda} is the best PAC-Bayes inspired objective in terms of test performance, although the risk certificates are generally less tight. iii) In most cases, the stochastic predictor does not worsen the performance of the prior mean predictor, improving it very significantly for random data-free priors (i.e. Rand.Init). iv) The mean of the weight distribution is also improved, as shown by comparing the results of the deterministic predictor (Det. pred.), corresponding to the posterior mean, with the prior mean predictor. The ensemble predictor also generally improves on the prior. v) The improvements brought by data-dependent priors (labelled

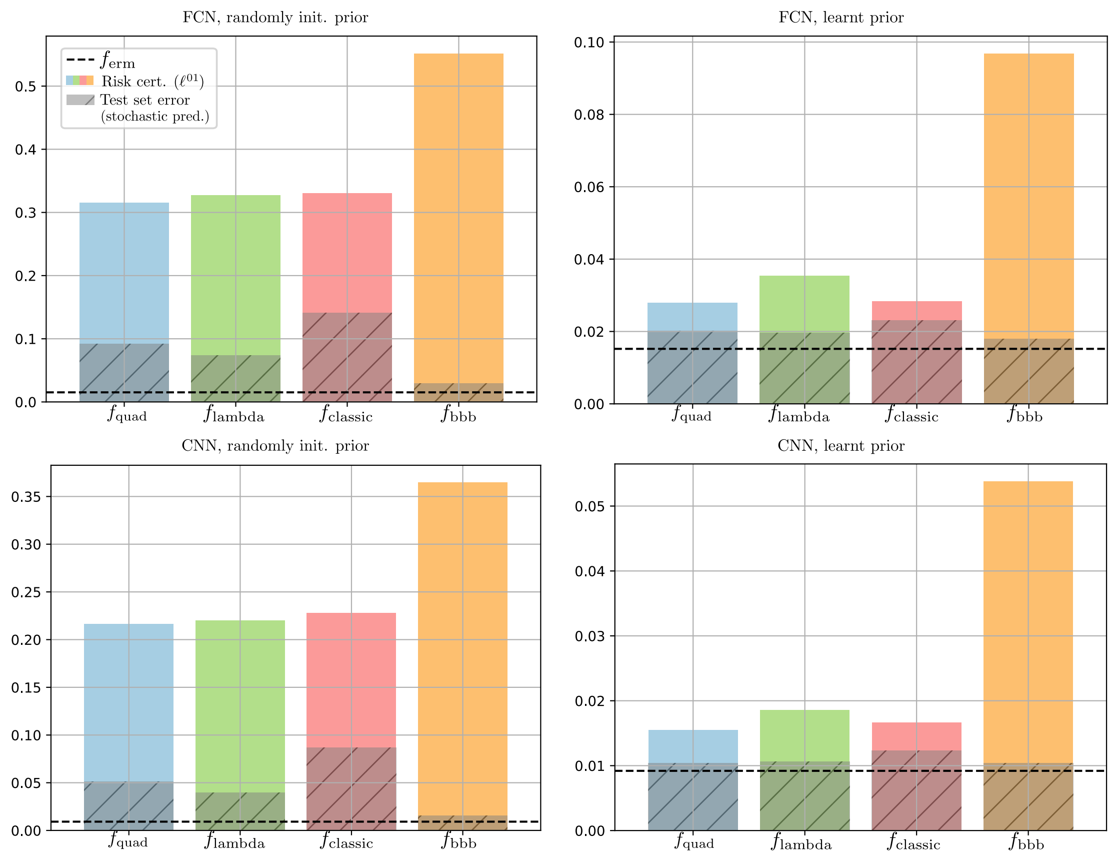

Figure 5.3: Tightness of the risk certificates for MNIST across different architectures, priors and training objectives. The bottom shaded areas correspond to the test set 0-1 error of the stochastic classifier. The coloured areas on top indicate the tightness of the risk certificate (smaller is better). The horizontal dashed line corresponds to the test set 0-1 error of f_{erm}, i.e. the deterministic classifier learnt by empirical risk minimisation of the surrogate loss on the whole training set (shown for comparison purposes).

as "Learnt" in the table) are consistent across the two architectures, showing better test performance and risk certificates (although the use of data-free priors still produced non-vacuous risk certificates). vi) The application of PBB is successful not only for learning fully connected layers but also for learning convolutional ones. The improvements in performance and risk certificates that the use of a CNN brings are also noteworthy. vii) The proposed PAC-Bayes inspired learning strategies show competitive performance (specially when using data-dependent priors) when compared to the Bayesian inspired f_{bbb} and the widely-used f_{erm}. Besides this comparable test set performance, our training methods also provide risk certificates with tight values.

We now compare our results to those reported by Dziugaite and Roy [2018a] for MNIST. Note that in this case there are differences regarding optimiser, prior chosen and weight initialisation (however, the neural network architecture used is the same, FCN as described in this chapter). Dziugaite and Roy [2018a] evaluated the bound of their Theorem 4.2 and the bound of Lever et al. [2013] for comparison. We compare the results reported by them with the results of training with our two training objectives f_{quad} and f_{lambda}, and with f_{classic} (optimised as per our f_{quad} and f_{lambda}). These results are presented in Table 5.2.

	Training method	Stch. Pred. 01 Err	Risk cert. ℓ^{01}	Bound used
D&R 2018	SGLD ($\tau = 3e+3$)	0.1200	0.2100 0.2600	D&R18 Thm. 4.2 Lever et al. 2013
D&R 2018	SGLD ($\tau = 1e+5$)	0.0600	0.6500 1.0000	D&R18 Thm. 4.2 Lever et al. 2013
This work	SGD + f_{quad}	*0.0202*	**0.0279**	PAC-Bayes-kl
	SGD + f_{lambda}	**0.0196**	0.0354	PAC-Bayes-kl
	SGD + f_{classic}	0.0230	*0.0284*	PAC-Bayes-kl

Table 5.2: Comparison of test set error rate (0-1 loss) for the stochastic predictor and its risk certificate for the standard MNIST data set. We compare here our results for the FCN with data-dependent priors to the previous work of Dziugaite and Roy [2018a]. All methods reported in this table use data-dependent priors (albeit different ones) and exactly the same architecture of dimensions $784 \times 600 \times 600 \times 10$ (with 2 hidden layers of 600 units per layer).

The hyperparameter τ in both Dziugaite and Roy [2018a] and Lever et al. [2013] controls the temperature of a Gibbs distribution with unnormalised density $e^{-\tau \hat{L}_S(w)}$ with respect to some fixed measure on weight space. In the table we display only the two values of their τ parameter which achieve best test set error and risk certificate. We note that the best values reported by Dziugaite and Roy [2018a] correspond to test accuracy of 94% or 93% while in those cases their risk certificates (0.650 or 0.350, respectively), although non-vacuous, were far from being tight. On the other hand, the tightest value of their risk bound (0.21) only gives an 88% accuracy. In contrast, our PBB methods achieve close to 98% test accuracy (or 0.0202 test error). At the same time, as noted above, our risk certificate (0.0279) is much tighter than theirs (0.210), meaning that our training scheme (not only training objectives but also prior) are a significant improvement with respect to theirs (an order of magnitude tighter). Even more accurate predictors and tighter bounds are achieved by the CNN architecture, as shown in Table 5.1.

5.4.5 KL Attenuating Trick

As many works have pointed out before (and we have observed in our experiments), the problem with all the four presented training objectives is that the KL term tends to dominate and most of the work in training is targeted at reducing it, which effectively means often the posterior cannot move far from the prior. To address this issue, distribution-dependent [Lever et al., 2013] or data-dependent [Dziugaite and Roy,

Setup		Risk cert. &		Train metrics			Stch. pred.		Det. pred.		Ens. pred.		Prior
Arch. & Prior	Obj.	$\ell^{\text{x-e}}$	ℓ^{01}	KL/n	$\hat{L}^{\text{x-e}}_S(Q)$	$\hat{L}^{01}_S(Q)$	x-e	01 err.	x-e	01 err.	x-e	01 err.	01 err.
CNN	f_{quad}	.2292	**.2824**	.2174	.0097	.0330	.0084	.0305	.0042	.0193	.0002	.0201	.9478
Rand.Init	f_{lambda}	.2840	.3241	.3004	.0066	.0225	.0058	*.0222*	.0039	.0144	.0002	.0148	.9478
(KL	f_{classic}	.2297	*.2846*	.2167	.0101	.0344	.0096	.0343	.0047	.0208	.0002	.0216	.9478
attenuating)	f_{bbb}	.4815	.4974	.6402	.0024	.0082	.0035	**.0107**	.0024	.0082	.0000	.0079	.9478
CNN	f_{quad}	.0191	**.0296**	.0104	.0030	.0087	.0033	.0101	.0000	.0095	.0000	.0096	.0104
Learnt	f_{lambda}	.0245	*.0354*	.0162	.0025	.0076	.0031	**.0092**	.0040	.0092	.0000	.0095	.0104
(KL	f_{classic}	.0187	**.0296**	.0100	.0031	.0089	.0037	.0106	.0043	.0095	.0001	.0095	.0104
attenuating)	f_{bbb}	.0470	.0557	.0421	.0012	.0041	.0034	*.0096*	.0025	.0085	.0001	.0083	.0104

Table 5.3: Training and test set results on MNIST using Gaussian distributions over weights and a penalty of $\eta = 0.001$ on the KL term for all the training objectives shown. Only a CNN architecture is considered.

2018a] priors have been used in the literature. Another approach to address this is to add a coefficient that controls the influence of the KL in the training objective [Blundell et al., 2015]. This means that in the case of f_{bbb} we could see marginal decrease in the KL divergence during the course of training (specially given small KL attenuating coefficients) and the solution it returns is expected to be similar to that returned simply using ERM with cross-entropy. However, this also has its effects on the risk certificate. To show these effects, we run all four training objectives with a KL penalty of 0.0001 during training and report the results in Table 5.3. For simplicity, only a CNN architecture is considered in this experiment. What we can see comparing these results to the ones reported in Table 5.1 is that while the 0-1 error for the stochastic classifier decreases, the KL term increases and so does the final risk certificate. Practitioners may want to consider this trade-off between test set performance and tightness of the risk certificates.

5.4.6 Laplace Weight Distributions

We experimented with both Laplace and Gaussian distributions over weights. The results are presented in Table 5.4. Comparing these to the results with Gaussian weight distributions from Table 5.1, we did not observe significant and consistent differences in terms of risk certificates and test set error between the two kinds of prior/posterior distributions. The distribution to use could be problem-dependent, but we found that both Gaussian and Laplace distributions achieve good risk certificates and test set performance.

Figure 5.4 shows a summary of all the results obtained for MNIST (i.e. results reported in Table 5.1 and Table 5.4). This shows clearly the differences between the three training objectives: f_{lambda} tends to lead generally to the lowest test set error, but worse risk certificates than f_{quad}, and f_{classic} leads to the worse test set performance and looser bounds. Thus, f_{quad} gives a reasonable trade-off between test set performance and tight risk certificates. The general trend of the relationship shows a slight curvature,

Setup			Risk cert. &		Train metrics			Stch. pred.		Det. pred.		Ens. pred.		Prior
Arch.	prior	Obj.	$\ell^{\text{x-e}}$	ℓ^{01}	KL/n	$\hat{L}_S^{\text{x-e}}(Q)$	$\hat{L}_S^{01}(Q)$	x-e	01 err.	x-e	01 err.	x-e	01 err.	01 err.
CNN	Rand.Init. (Laplace)	f_{quad}	.1548	**.2425**	.1024	.0207	.0709	.0190	.0677	.0113	.0429	.0004	.0436	.9478
		f_{lambda}	.1844	.2540	.1489	.0147	.0496	.0131	*.0461*	.0096	.0310	.0003	.0312	.9478
		f_{classic}	.1334	*.2489*	.0610	.0322	.1101	.0296	.1014	.0208	.0719	.0007	.0695	.9478
		f_{bbb}	.4280	.4487	.5385	.0031	.0107	.0038	**.0139**	.0006	.0096	.0001	.0090	.9478
	Learnt (Laplace)	f_{quad}	.0085	*.0167*	.0004	.0056	.0126	.0043	*.0098*	.0011	.0103	.0001	.0103	.0104
		f_{lambda}	.0119	.0216	.0025	.0049	.0118	.0041	.0106	.0052	.0103	.0003	.0100	.0104
		f_{classic}	.0076	**.0155**	.0000	.0060	.0131	.0046	.0107	.0015	.0105	.0001	.0106	.0104
		f_{bbb}	.0737	.0866	.0673	.0019	.0062	.0031	**.0092**	.0013	.0093	.0001	.0091	.0104

Table 5.4: Training and test set results on MNIST using Laplace distributions over weights. For simplicity, only a CNN architecture is considered here.

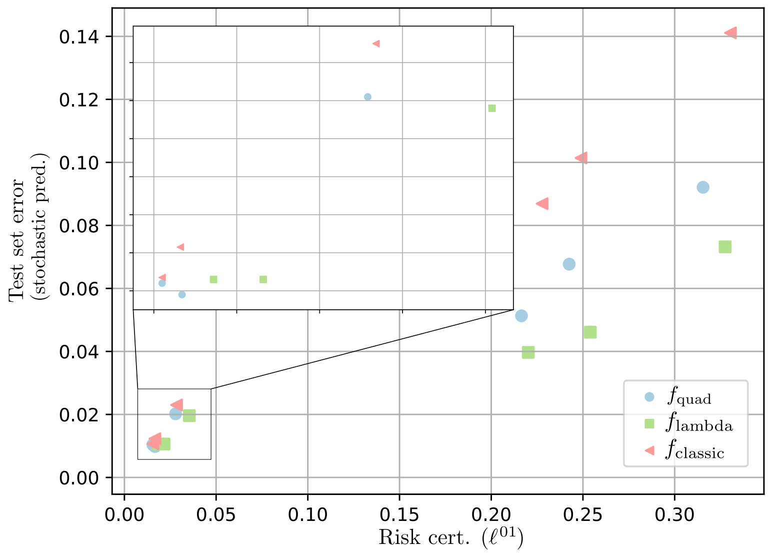

Figure 5.4: Scatter plot of the results obtained for MNIST using different training objectives. The x-axis shows values of the risk certificate (under ℓ^{01} loss), and the y-axis shows the test set error rates, achieved by the stochastic classifier.

as also seen in Figure 5.1.

5.4.7 CIFAR-10 with Larger Architectures

We evaluate now our training objectives on CIFAR-10 using deep CNN architectures. Note that this is a much larger scale experiment than the ones presented before (15 layers with learnable parameters vs 4). As far as we know, we are the first to evaluate PAC-Bayes inspired training objectives in such deep architectures. The results are presented in Table 5.5 and Figure 5.5 for three architectures (with 9, 13 and 15 layers, with around 6M, 10M and 13M parameters, respectively). Note, however, that the number of parameters is doubled for our probabilistic neural networks. We also experiment with using different amount of data for learning the prior: 50% and 70%, leaving respectively 25.000 and 15.000 examples to evaluate the bound. The conclusions are as follows: i) In this case, the improvements brought by learning the posterior through PBB with

Setup			Risk cert. &	Train metrics				Stch. pred.		Det. pred.		Ens. pred.		Prior
Arch.	Prior	Obj.	$\ell^{\text{x-e}}$	ℓ^{01}	KL/n	$\hat{L}_S^{\text{x-e}}(Q)$	$\hat{L}_S^{01}(Q)$	x-e	01 err.	x-e	01 err.	x-e	01 err.	01 err.
CNN (9 layers)	Learnt (50% data)	f_{quad}	.1296	*.3034*	.0089	.0868	.2428	.0903	.2452	.0726	.2439	.0024	.2413	.2518
		f_{lambda}	.1742	.3730	.0611	.0571	.2108	.0689	*.2307*	.0609	.2225	.0018	.2133	.2518
		f_{classic}	.1173	**.2901**	.0035	.0903	.2511	.0931	.2537	.0952	.2437	.0025	.2332	.2518
		f_{bbb}	.8096	.8633	1.5107	.0239	.0926	.0715	**.2198**	.0735	.2160	.0017	.2130	.2518
	Learnt (70% data)	f_{quad}	.1017	*.2502*	.0026	.0796	.2179	.0816	*.2137*	.0928	.2137	.0023	.2100	.2169
		f_{lambda}	.1414	.3128	.0307	.0630	.2022	.0708	**.2081**	.0767	.2061	.0021	.2049	.2169
		f_{classic}	.0957	**.2377**	.0004	.0851	.2223	.0862	.2161	.0827	.2167	.0021	.2135	.2169
		f_{bbb}	.6142	.6965	.8397	.0212	.0822	.0708	.1979	.0562	.1992	.0019	.1944	.2169
	-	f_{erm}	-	-	-	.0355	.0552	-	-	.1400	.1946	-	-	-
CNN (13 layers)	Learnt (50% data)	f_{quad}	.0821	*.2256*	.0042	.0577	.1874	.0585	.1809	.0519	.1788	.0011	.1783	.1914
		f_{lambda}	.1163	.2737	.0272	.0491	.1741	.0516	*.1740*	.0466	.1726	.0015	.1690	.1914
		f_{classic}	.0757	**.2127**	.0009	.0635	.1936	.0622	.1880	.0592	.1810	.0017	.1816	.1914
		f_{bbb}	.6787	.7566	.9999	.0250	.0924	.0505	**.1676**	.0422	.1646	.0011	.1614	.1914
	Learnt (70% data)	f_{quad}	.0659	*.1832*	.0015	.0519	.1608	.0517	.1568	.0421	.1553	.0010	.1546	.1587
		f_{lambda}	.0896	.2177	.0145	.0449	.1499	.0479	*.1541*	.0604	.1522	.0011	.1507	.1587
		f_{classic}	.0619	**.1758**	.0002	.0548	.1644	.0541	.1588	.0605	.1578	.0013	.1557	.1587
		f_{bbb}	.4961	.5858	.5826	.0213	.0772	.0487	**.1508**	.0532	.1495	.0016	.1461	.1587
	-	f_{erm}	-	-	-	.0576	.0810	-	-	.0930	.1566	-	-	-
CNN (15 layers)	Learnt (50% data)	f_{quad}	.0867	*.2174*	.0053	.0587	.1753	.0584	.1668	.0538	.1662	.0014	.1653	.1688
		f_{lambda}	.1217	.2707	.0304	.0494	.1661	.0506	*.1618*	.0417	.1639	.0015	.1622	.1688
		f_{classic}	.0782	**.1954**	.0007	.0667	.1783	.0652	.1686	.0594	.1692	.0013	.1674	.1688
		f_{bbb}	.6069	.7066	.7908	.0287	.1073	.0468	**.1553**	.0412	.1530	.0012	.1517	.1688
	Learnt (70% data)	f_{quad}	.0756	*.1806*	.0028	.0559	.1513	.0559	.1463	.0391	.1469	.0016	.1449	.1490
		f_{lambda}	.0922	.2121	.0133	.0486	.1477	.0500	*.1437*	.0507	.1449	.0012	.1438	.1490
		f_{classic}	.0703	**.1667**	.0003	.0622	.1548	.0615	.1475	.0551	.1480	.0010	.1476	.1490
		f_{bbb}	.4481	.5572	.4795	.0259	.0947	.0455	**.1413**	.0395	.1405	.0008	.1409	.1490
	-	f_{erm}	-	-	-	.0208	.0339	-	-	.0957	.1413	-	-	-

Table 5.5: Training and test set results on CIFAR-10 using Gaussian distributions over weights. The table includes results for three deep CNN architectures (with 9, 13, and 15 layers, respectively) and data-dependent PAC-Bayes priors which are obtained via empirical risk minimisation for learning the prior mean using two percentages of the data (50% and 70%, corresponding to 25.000 and 35.000 examples respectively). For the stochastic predictor, the best risk certificate and test set error are highlighted in bold face, and second best are highlighted in italics.

respect to the prior are much better and generally consistent across all experiments (e.g. 2 points in test 0-1 error for f_{lambda} when using 50% of the data for learning the prior). ii) Risk certificates are also non-vacuous and tight (although less than for MNIST). iii) We validate again that f_{lambda} shows better test performance but less tight risk certificates. iv) In this case, however, f_{classic} and f_{quad} seem much closer in terms of performance and tightness. In some cases, f_{classic} provides slightly tighter bounds, but also often worse test performance. The tighter bounds can be explained by our findings with the Pinsker inequality, which makes f_{classic} tighter when true loss is more than 0.25. This observation can be seen clearly in Figure 5.6. v) Obtained results with 15 layers are competitive, achieving similar performance than those reported for VGG-16 [Simonyan and Zisserman, 2015] (deep network proposed for CIFAR-10 with comparable architecture to the one tested with only fully connected and convolutional

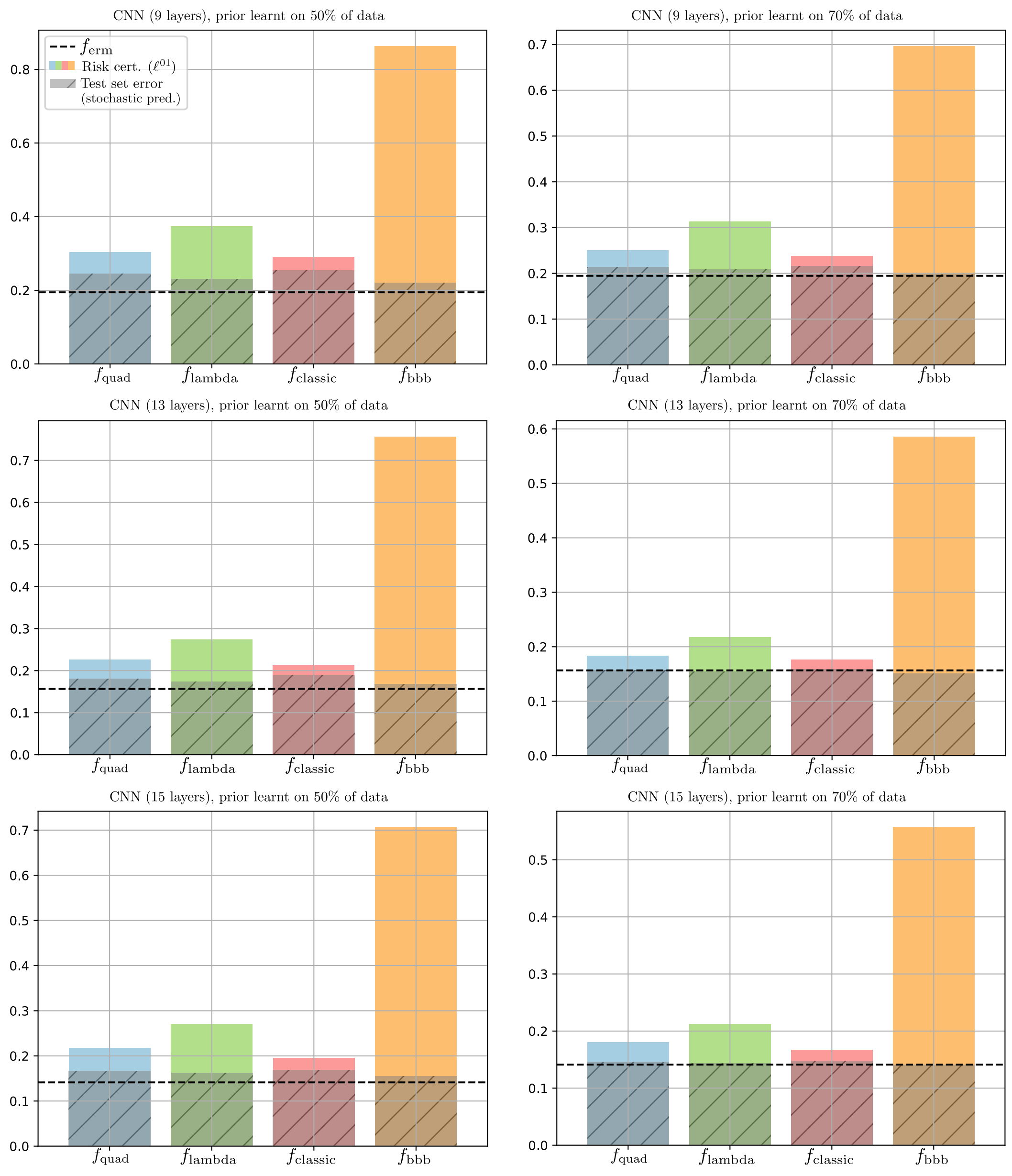

Figure 5.5: Tightness of the risk certificates on CIFAR-10 for 3 different network architectures and two data-dependent priors (learnt using 50% and 70% of the data).

layers). vi) The results indicate that 50% of the training data is not enough in this data set to build a competitive prior and this influences the test performance and the risk certificates. The results with 70% of the data are, however, very close to those achieved by ERM across all three architectures. vii) Similarly than with the rest of the experiments, a major difference can be seen when comparing the risk certificate achieved by f_{bbb} with the risk certificate achieved by PAC-Bayes inspired training objectives. viii) Finally, it is noteworthy how the KL gets generally smaller as we move to deeper architectures (specially from 9 to 13 layers), which is an interesting observation, as there are many more parameters used in the computation of the KL. This

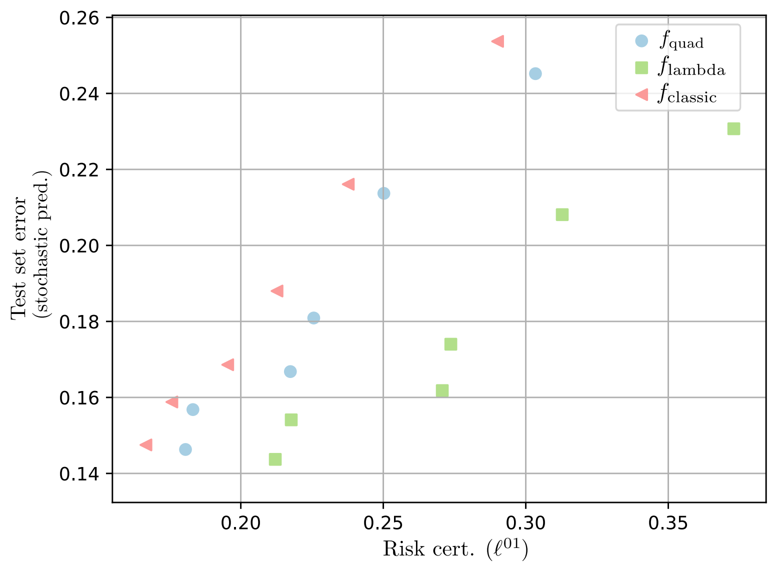

Figure 5.6: Scatter plot of the results obtained for CIFAR-10 using three different training objectives. The x-axis shows values of the risk certificate (under ℓ^{01} loss), and the y-axis shows the test set error rates, achieved by the stochastic classifier.

indicates that the posterior in deeper architectures stays much closer to the prior. We believe this may be because in a higher-dimensional weight space, the weight updates have a smaller euclidean norms, hence the smaller KL.

Note that more competitive and deeper neural baselines exist for CIFAR-10 nowadays. However, those deeper architectures often require of more advanced training strategies such as batch norm, data augmentation, cyclical learning rates, weight decay, etc. In our experiments, we decided to keep the training strategy as simple as possible, in order to focus on the ability of our training objectives alone to give good predictors and, more importantly, risk certificates with tight values. It is noteworthy that our training objectives are able to achieve this with a simple training strategy, and we leave the exploration of all the available training choices as future work.

5.4.8 Additional Miscellaneous Experiments

In this section we discuss four interesting observations from our experiments, which we believe mark promising future research directions.

First, we present a plot of the performance obtained when using different training epochs to learn the prior and posterior. Figure 5.7 and 5.8 show a contour plot of the loss and risk certificate when training the prior and the posterior for different epochs (e.g. to check the effect of training the posterior with an under-fitted prior). These plots have been generated using the FCN architecture on MNIST with Gaussian distributions over weights. Similar results are obtained for the CNN architecture. Note that for the sake of visualisation in Figure 5.7 we are plotting much less epochs that those used to generate the final results (in this case up to 20, whereas the rest of reported results were with 100

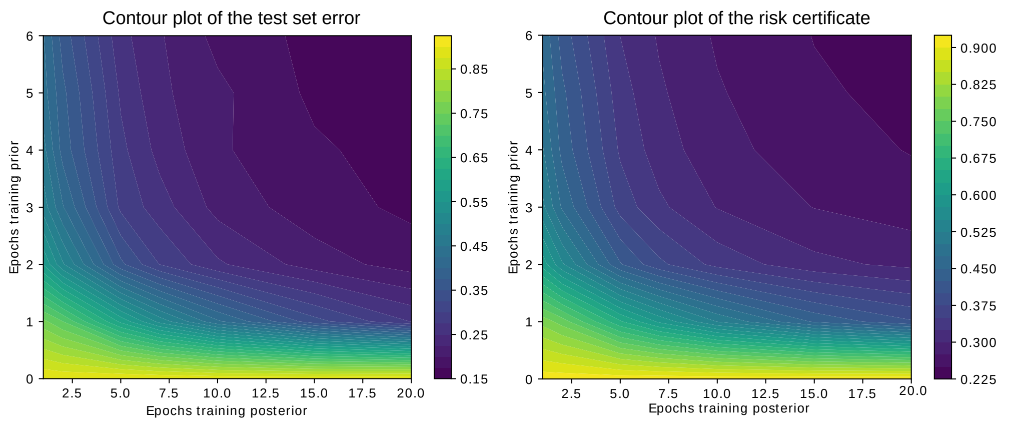

Figure 5.7: Contour plots of the test set error and risk certificate (under ℓ_{01}) after different training epochs learning the prior and posterior and initial scale hyper-parameters value $\sigma_0 = 0.1$ for the prior.

epochs) so the reported test set errors and risk certificates in this plot differ from those previously reported. Figure 5.7 shows that both training the prior and the posterior are crucial to improve the final loss and risk certificates, as the best loss and risk certificate values are found in the top right corner of the plot. The plot also shows that if the prior is under-fitted (e.g. if trained for only one or two epochs), then the final predictor can still be much improved with more training epochs for the posterior. However, a more adequate prior means that less epochs are needed to reach a reasonable posterior. Nonetheless, this is less apparent if the prior is not learnt (represented here as a training of 0 epochs, i.e. a random prior), in which case learning the posterior for longer does not seem to reach such competitive posteriors, which demonstrates the usefulness of data-dependent priors for obtaining tight risk certificates. In this experiment depicted in Figure 5.7 and 5.8, we also noted that only a few epochs of training the prior are enough to reach competitive posteriors and that learning the posterior for much longer (e.g. 1000 epochs) does not lead to overfitting, which reinforces the finding of Blundell et al. [2015] that the KL term act as a regulariser. Specifically, this can be seen in Figure 5.8, which shows that training the posterior for a large number of epochs does not worsen the test set error and the risk certificate. There are still small scale differences (of up to 1%) in risk certificate and test set error for the dark blue colour region, but these can not be visually seen because of the scale of the colour legend. However, the important observation is that the differences are small across the dark blue region (if there were significant differences within this region, then that would be an evidence of overfitting). This is, however, opposed to what we observe when training the prior through empirical risk minimisation, since the prior overfits easily in that case, which is why we had to learn the priors using dropout in all our experiments.

Next, we compare the test set performance of the different predictors considered in this work (stochastic, deterministic and ensemble). The results for MNIST and

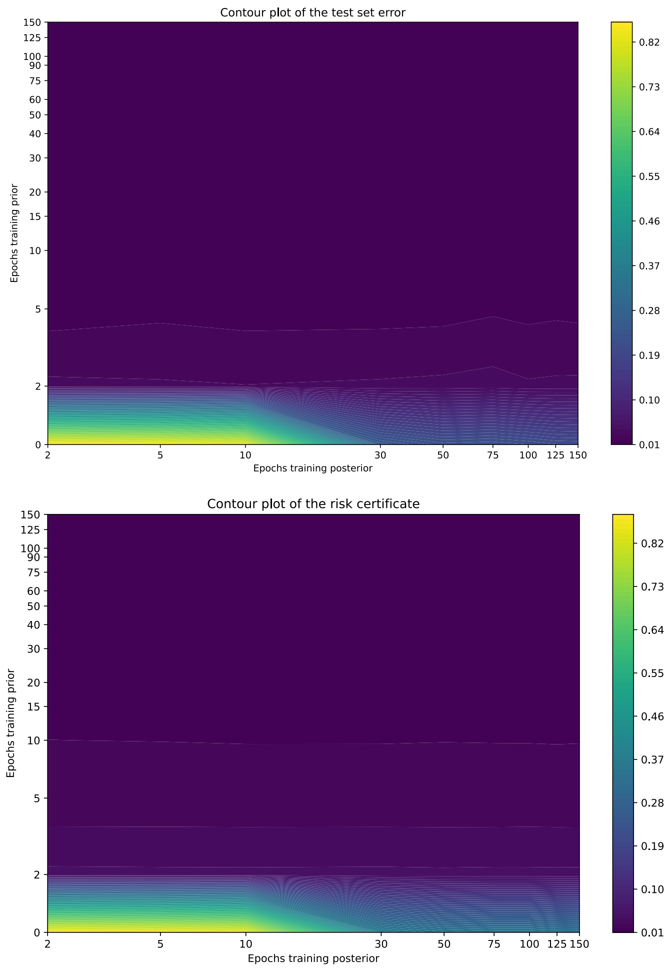

Figure 5.8: Contour plots of the test set error and risk certificate (under ℓ_{01}) after different training epochs learning the prior and posterior. Dropout is used when learning the prior. Note that training the posterior for a large number of epochs does not worsen the test set error or the risk certificate.

CIFAR-10 are depicted in Figure 5.9. One can appreciate a very clear linear relationship between predictors. In the case of CIFAR-10 the results are similar across all predictors, whereas for MNIST the stochastic predictor obtains significantly worse results (see differences in scales of x and y axes). In the case of CIFAR-10 this may hint that our training strategy finds a solution within a large region of comparably good solutions, so that weight randomisation does not affect significantly the test performance of the classifier. We plan to explore this interesting phenomenon in future work.

Thirdly, in Figure 5.10 we show a histogram of the final scale parameters $\hat{\sigma}$ (i.e. standard deviation) for the Gaussian posterior distribution (both weights and biases). The plot shows that the optimisation changes the scale of different weights and biases, reducing specially those associated to the input and output layer. We think it is worth to experiment with different scale initialisations per layer in future work, as well as different covariance structures for the weight and bias distributions.

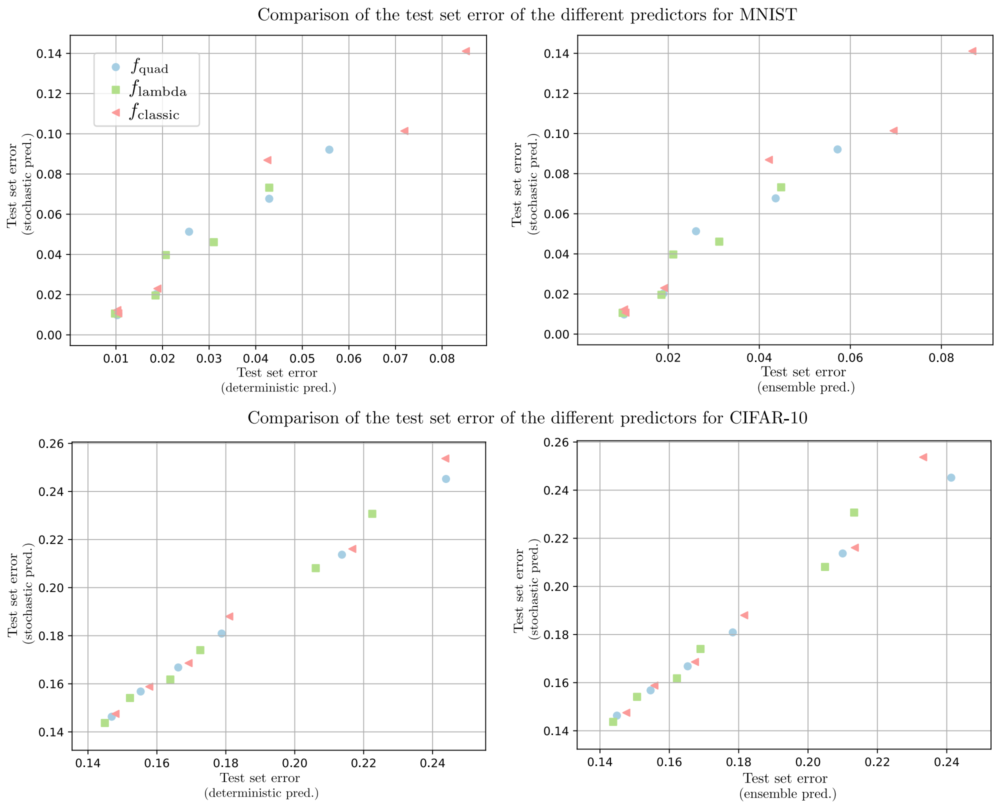

Figure 5.9: Representation of the results achieved by the different predictors that were studied (stochastic, deterministic, and ensemble).

Finally, we aim to validate the use of the learnt posterior for uncertainty quantification. To do so, we use the ensemble predictor (100 members) using the CNN architecture in MNIST. Each member of the ensemble is a sample from the posterior. We define uncertainty as the number of members of the ensemble that disagree in the prediction.[8] Figure 5.11 shows the test set digits for which the ensemble is most certain (top row) and uncertain (bottom row). It can be seen that the most uncertain digits indeed look unusual and could even confuse a human, whereas the most certain digits are easily identifiable as 4, 6 and 9. We believe that this simple visual experiment may indicate that there is promise in probabilistic neural networks trained by PBB objectives being of use for uncertainty quantification. However, more experiments in this direction are needed.

5.4.9 Further Discussion

We now discuss further the probabilistic neural network models studied in this chapter, with a focus on their practical usefulness. We have demonstrated that the randomised predictors learnt by PBB come with a tight performance guarantee that is valid at

[8]Similar measures of disagreement have been used in the literature on majority vote classifiers, see e.g. Lacasse et al. [2006], Germain et al. [2015], Masegosa et al. [2020]; and in some related literature on domain adaptation, e.g. Germain et al. [2013, 2016b, 2020].

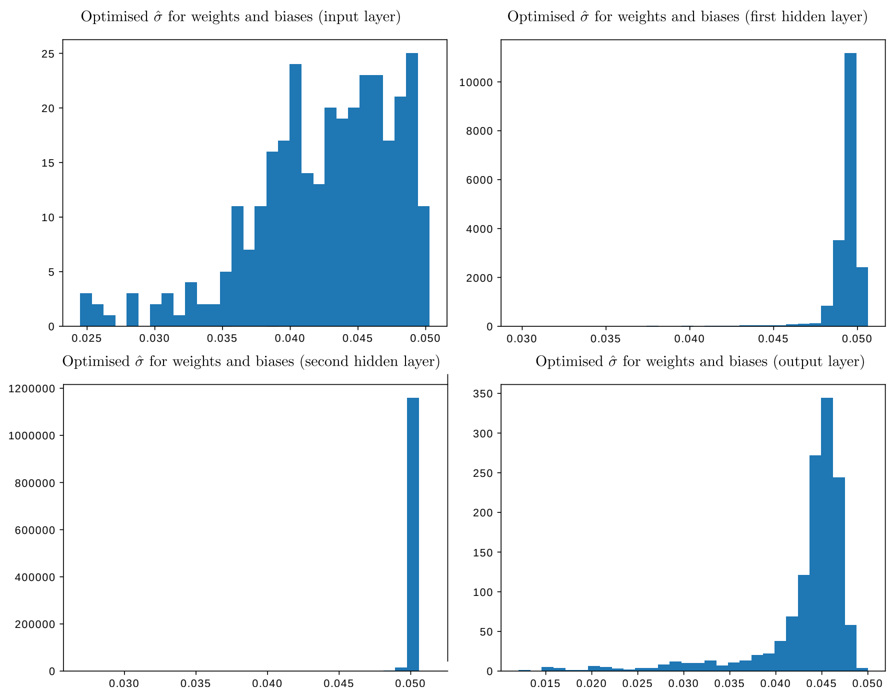

Figure 5.10: Histograms of the scale parameters for the Gaussian distribution at the end of the optimisation for the different layers of the CNN architecture on MNIST. All scale parameters were initialised to 0.05, i.e. $\sigma_0 = 0.05$ for all coordinates, and $\hat{\sigma}$ is the scale parameter value of the final output of training.

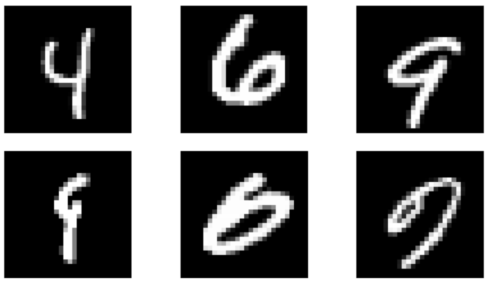

Figure 5.11: Representation of the test set digits for three classes (4, 6 and 9) in which the ensemble predictor is most certain/uncertain. The top row shows the digits with minimum uncertainty (all 100 members of the ensemble agree in the prediction). The bottom row shows the digits with highest uncertainty.

population level, and is evaluated on a subset of the data used to train the PAC-Bayes posterior, i.e. evaluation of our certificates does not require a held out test set. We have observed that our methods show promise for self-certified learning, which is a data-efficient principle, and also shown that the same bound used for post-training evaluation of the risk certificate is useful for model selection. Practitioners may want to consider all these favourable properties.

However, probabilistic neural networks have additional advantages over their standard point estimator counterparts. The results of Blundell et al. [2015] have shown that probabilistic neural networks enable an intuitive and principled implementation of uncertainty quantification and classification reject options (e.g. allow the model to say "I don't know" when the classification uncertainty for a new example is higher than a certain threshold). Similarly, we have also shown the use of our models for uncertainty quantification in a very simple experiment with the ensemble predictor. This is just one example of the advantages of probabilistic neural network models (distributions over weights) compared to using point estimator models (fixed weights), but these models have shown promise towards many other goals, such as model pruning/distillation. Blundell et al. [2015] also showed that learning a weight distribution by minimising the empirical loss while constraining its KL divergence to a prior gives similar results to implicit regularisation schemes (such as dropout). Similarly, in the experiments with our training objectives we have seen that overfitting was only an issue while learning the prior through ERM, but not during the posterior learning phase (as demonstrated by Figure 5.8).

Even though we have not experimented exhaustively with all of the cases described above, we hypothesise that all these advantages extend to probabilistic neural networks learnt by PAC-Bayes inspired objectives. This may make the use of stochastic classifiers with PAC-Bayes bounds more desirable than point estimator models with a PAC bound. This is also notwithstanding the tightness of the former, in contrast with the latter, which are known to be notoriously vacuous for the kinds of models studied in our experiments. All of these hypotheses should be validated thoroughly in future work.

5.5 Conclusion and Future Work

This chapter is based on my paper Pérez-Ortiz et al. [2021b]. In this work we explored 'PAC-Bayes with Backprop' (PBB) methods to train probabilistic neural networks with different weight distributions, priors and network architectures. The take-home message is that the training methods presented in this chapter are derived from sound theoretical principles and provide a simple strategy that comes with a performance guarantee that is valid at population level, i.e. valid for any unseen data from the same distribution as the training data. This is an improvement over methods derived heuristically rather than from theoretically justified arguments, and over methods that do not include a risk certificate valid on unseen examples. Additionally, we empirically demonstrate the usefulness of data-dependent priors for achieving competitive test set performance and, importantly, for computing risk certificates with tight values.

The results of our experiments on MNIST and CIFAR-10 have showed that these PBB objectives give predictors with competitive test set performance and with non-vacuous risk certificates that significantly improve previous results and can be used not only for guiding the learning algorithm and certifying the risk but also for model selection. This shows that PBB methods are promising examples of self-certified learning, since the values of the risk certificates output by these training methods are tight, i.e. close to the values of the test set error estimates. In particular, our results in MNIST with a small convolutional neural network (2 hidden layers) achieve 1% test set error and a risk certificate of 1.5%. We also evaluated our training objectives on large convolutional neural networks (up to 15 layers and around 13M parameters) with CIFAR-10. These results also showed risk certificates with tight values (18% of risk certificate for a stochastic predictor that achieves 14.6% of test set error). Note that to claim that self-certified learning is achieved would require testing a given training method across a wide range or data sets and architectures (so as to experimentally validate the claim), or theoretically characterising the problems on which a given learning method is guaranteed to produce tight risk certificates.

In future work we plan to test different covariance structures for the weight distribution and validate a more extensive list of choices for the weight distributions across a larger list of data sets. We also plan to experiment how to approach the well-known dominance of the KL term in the optimisation of these objectives. Data-dependent priors seem like a promising avenue to do so. We also plan to explore deeper architectures. Finally, we plan to study risk certificates for the ensemble predictor.

Chapter 6

Epilogue

This thesis concerned learning and certification strategies for randomised classification algorithms which are defined by a probability distribution over the weight space corresponding to a parametric hypothesis class. Both strategies (for learning and for certification) were based on PAC-Bayes bounds. In particular, the certification strategy was based on upper-bounding the risk via the PAC-Bayes-kl bound, which is very tight, and the learning strategies consisted of converting various PAC-Bayes bounds into optimisation objectives which are then optmised in a data-driven way. This justifies the name PAC-Bayesian Computation (PBC) which is the title of this thesis.

The opening Chapter 1 outlined (verbally) various kinds of generalisation bounds; the content of this chapter re-used discussions that appeared in the introduction sections of my papers Rivasplata et al. [2020] and Pérez-Ortiz et al. [2021b]. The background Chapter 2 gave a concise account of statistical learning and PAC-Bayes bounds while also setting the terminology and notation for the thesis; the content of this chapter was based on knowledge that I have accumulated over the years (references were given) and largely re-used material that has appeared in corresponding sections of my papers Rivasplata et al. [2020] and Pérez-Ortiz et al. [2021b]. Then the remaining chapters of the thesis were Chapter 3, Chapter 4, and Chapter 5, each of which is based on a paper of mine (Rivasplata et al. [2020], Rivasplata et al. [2018], and Pérez-Ortiz et al. [2021b], respectively) published at a top-tier machine learning venue.

The presentation of PAC-Bayes bounds given in Chapter 2 was centered around a general PAC-Bayes theorem, namely Theorem 2.1, and how this theorem is used to derive specific PAC-Bayes bounds. As discussed there, essentially two steps are involved in deriving a PAC-Bayes bound from Theorem 2.1: (i) choose a convex function F (and a prior Q^0) to use in Eq. (2.7), and (ii) obtain an upper bound on the exponential moment ξ defined in Eq. (2.6). For illustration, it was discussed how these steps were done for deriving the well-known PAC-Bayes bounds that have appeared in the literature (namely the classic one of McAllester, the PAC-Bayes-kl bound of Langford & Seeger, and the bound of Catoni), as well as other more recent ones (the PAC-Bayes-λ bound of Thiemann et al., and the PAC-Bayes-quadratic bound). The prominent role of the exponential moment term ξ defined in Eq. (2.6) was discussed, highlighting in particular that the usual assumptions of data-free PAC-Bayes priors, bounded loss functions and i.i.d. data, which are emphasised in the previous literature on PAC-Bayes bounds, only came into play when specific techniques were used for upper-bounding ξ, while the general Theorem 2.1 is valid without these assumptions. I would like to highlight that Theorem 2.1 is a slight modification of the general PAC-Bayes theorem given by Germain et al. [2009], and indeed the whole content of Chapter 2 concerns previous knowledge that existed before my work (references were given throughout the chapter). An important feature of the PAC-Bayes bounds derived from Theorem 2.1 is that they give a high-probability inequality that holds uniformly for all distributions over weights, which makes these bounds useful for optimisation: one can re-use the bound successively during optimisation without changing the form of the bound! Naturally, these bounds can be used for risk certification by applying the bound to the particular 'posterior' distribution found by some learning method. The presentation was kept at a high level of generality in this chapter, in particular without going over instantiations of the bounds for specific learning problems. However, a discussion of the related literature was provided, to the extent that it was known to me. Some of the more recent literature is mentioned below.

The main topic of Chapter 3 was an extension of the PAC-Bayes analysis to stochastic kernels, which are a convenient way to formalise data-dependent distributions over a weight space. The content of this chapter is from my paper Rivasplata et al. [2020]. The main results of this chapter were a theorem giving two high-probability inequalities for stochastic kernels (Theorem 3.1), and general PAC-Bayes theorem for stochastic kernels (Theorem 3.2), from which one may derive PAC-Bayes style bounds of similar forms to the usual PAC-Bayes bounds in the literature (namely the one of McAllester, that of Langford & Seeger, and that of Catoni) and novel bounds, with derivation arguments that follow the same two essential steps (i) and (ii) as discussed in the previous paragraph. The novelty of the new results is that they enable PAC-Bayes priors that are data-dependent by default, whereas the usual bounds in the literature required such priors to be data-free. As discussed in Chapter 3, the trade-off in this extension is that the new results give high-probability inequalities that hold for a given 'posterior' distribution, whereas the usual bounds in the previous literature hold uniformly for all distributions over the weight space. This chapter, as my paper Rivasplata et al. [2020], also discussed the restrictions of (a) data-free priors, (b) bounded losses, and (c) i.i.d. data; which have been emphasised throughout the previous literature on PAC-Bayes bounds. The take-home message regarding these restrictions is that they only play a role in step (ii) concerning the exponential moment, while the general theorem holds without these restrictions, which is an insight that may lead to discovering novel bounds for unbounded losses and for non-i.i.d. data.

The main topic of Chapter 4 was risk certification strategies for randomised SVM classifiers. The content of this chapter is from my paper Rivasplata et al. [2018]. This work combined algorithmic stability with the PAC-Bayes-kl bound in order to develop a risk bound for the randomised SVM classifier defined by a Gaussian distribution over the weight space, centered at the weight vector output by the SVM optimisation process and with a suitably defined covariance. The stability notion used here was applied to the weight vector found by the SVM optimisation, and it was defined in terms of the sensitivity of the weight vector to a small change in the composition of the data set. Then, the stability analysis led to a concentration of the weight around its expectation, and in turn the latter led to a PAC-Bayes bound for the risk of the randomised SVM classifier, which was obtained by using a Gaussian distribution centered at the expected weight vector as 'prior' distribution in the PAC-Bayes-kl bound. In the numerical experiments this novel stability-based PAC-Bayes bound was compared to other generalisation bounds, in particular to a previous stability-based (but not PAC-Bayes) bound and a previous PAC-Bayes (but not stability-based) bound, with respect to tightness of the numerical values of the risk certificates, as well as the test set error rates of the classifiers. The take-home message of this work is that the proposed novel bound was the first stability-based PAC-Bayes bound, and its performance on the numerical experiments showed its ability to produce tight bound values on various benchmark problems.

The topic of Chapter 5 was learning and certification strategies for neural network classifiers. The content of this chapter is from my paper Pérez-Ortiz et al. [2021b]. An early version of this work was presented at a workshop [Pérez-Ortiz et al., 2020]. This work explored various learning and certification strategies for randomised neural network classifiers, where the randomisation was defined by a suitably chosen Gaussian distribution over the connection weights of a given neural network architecture. The learning strategy consisted of learning the Gaussian mean and variance parameters, which was done by optimising a PAC-Bayes bound via SGD. Several objectives inspired by corresponding PAC-Bayes bounds were investigated. The 'prior' distributions used for the learning strategy were Gaussians centered at the randomly initialised weights, and Gaussians centered at the weights learned by SGD optimisation using a subset of the training set. The certification strategy used the PAC-Bayes-kl bound which was evaluated on the part of the training set that was not used for training the prior. Importantly, however, the Gaussian posterior parameters were optimised using the whole training set, i.e. including the data that was used to learn the prior. This work investigated several training objectives, including two that were used here for the first time for learning a distribution over neural network weights, as well as a previously used training objective inspired by the classical PAC-Bayes bound, and the variational Bayesian learning objective. The experiments compared all these objectives, and also compared to the output of plain ERM which is the most widely used neural network training method. The comparisons addressed the values of the risk certificates for the randomised (aka stochastic) classifiers (for all methods except ERM), and also the values of the test error rates evaluated on held out data (all methods, including ERM) for various choices of neural network architectures and prior distributions. The results of the experiments showed that the randomised classifiers defined by distributions learned by optimising PAC-Bayes bounds (more precisely, the training objectives derived from them) not only achieve competitive test set error rates as evaluated on the held out test set, but also they give remarkably tight risk certificates, in many cases with values of the same order of magnitude as the corresponding test set error rates (e.g. a risk certificate of $\sim 1.5\%$ for a classifier with a test set error rate of $\sim 1\%$ on MNIST). This work therefore made a significant contribution to certification for deep learning methods.

More works on PAC-Bayes bounds and their uses have appeared since the time of writing (during the summer of 2021) and submission (14 September 2021) of this thesis, and indeed since the publication dates of the papers on which this thesis is based. Subsequent works of mine [Pérez-Ortiz et al., 2021a] and others [Boll et al., 2022] have continued to explore the idea of self-certifiable learning methods. The recent study of Foong et al. [2021] explored the question of tightness of PAC-Bayes bounds in the small data regime. The recent work of Farid and Majumdar [2021] combined algorithmic stability at the "base level" and PAC-Bayes bounds at the "meta level" in order to derive learning guarantees for meta-learning. The recent work of Wu et al. [2021] presented a novel PAC-Bayes bound which the authors called the PAC-Bayes-Bennett inequality, and they used it to bound the expected risk of the weighted majority vote. The recent work of Zantedeschi et al. [2021] studied stochastic majority vote rules learned by minimising a PAC-Bayes bound. The recent work of Viallard et al. [2021] combined PAC-Bayes analysis and perturbations for majority votes to study adversarial robustness. The recent work of Clerico et al. [2021a,b] proposed strategies to train a Gaussian distribution over neural network weights by optimising the PAC-Bayes-kl bound. The recent work of Chérief-Abdellatif et al. [2022] proposed to leverage PAC-Bayes bounds to obtain reconstruction guarantees for variational autoencoders (VAEs). Another recent work I would like to highlight is Alquier [2021]'s tutorial which gives a very informative account of PAC-Bayes bounds and comprehensive coverage the related literature. There is a line of work connecting PAC-Bayes bounds with information-theoretic quantities, although I cannot in all honesty say that I am familiar with the latter, but see e.g. Grunwald et al. [2021], Steinke and Zakynthinou [2020], Hellström and Durisi [2020], Negrea et al. [2019]. There is a line of work on algorithmic learning theory that talks about "hedged predictions" in the sense that they "incorporate a valid indication of their own accuracy and reliability" [Vovk et al., 2005]; this line of work is not recent but rather I have recently become acquainted with it, and its ideas are then similar in some way to the ideas of self-certified learning.

Regarding future research directions I would like to mention the following. The work of Rivasplata et al. [2020] reported in Chapter 3 made available a general result for deriving PAC-Bayes style bounds with data-dependent priors; while a particular example was given, this work begs the question of what kinds of new PAC-Bayes style bounds with data-dependent priors may be derived in future work. The work of Rivasplata et al. [2018] reported in Chapter 4 considered exclusively SVM models without a bias (offset) term; while this choice seemed sensible at the time and was supported by some arguments in the literature (as discussed in Chapter 4), future work may ask how to reconcile this choice with the results of Hanneke and Kontorovich [2019]. The work of Pérez-Ortiz et al. [2021b] reported in Chapter 5 started a line of thought around the idea of 'self-certifiable learning' which continues to be explored further in subsequent works of mine with collaborators (and also by others); natural questions for future research include how to sensibly choose architectures for a given learning task, sensible choices of distributions for a given architecture, in particular whether there is a covariance structure that is the 'most suitable' in some sense, ways to reason about the uncertainty in the predictions of our randomised neural network classifiers (one of which was explored here in terms the the agreement/disagreement between members of an ensemble of predictors, but there are other ways to explore), whether and how it is possible to modify our strategies to construct conformal predictors (promising result were shown by Stutz et al. [2022]), among others.

London, March 14, 2022

Bibliography

Karim Abou-Moustafa and Csaba Szepesvári. An a priori exponential tail bound for k-folds cross-validation. arXiv:1706.05801, 2017.

Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *Journal of Machine Learning Research*, 19(50):1–34, 2018.

Charu C. Aggarwal. *Neural Networks and Deep Learning*. Springer, 2018.

Pierre Alquier. PAC-Bayesian bounds for randomized empirical risk minimizers. *Mathematical Methods of Statistics*, 17(4):279–304, 2008.

Pierre Alquier. User-friendly introduction to PAC-Bayes bounds. arXiv:2110.11216, 2021.

Pierre Alquier and Benjamin Guedj. Simpler PAC-Bayesian bounds for hostile data. *Machine Learning*, 107(5):887–902, May 2018.

Pierre Alquier and Olivier Wintenberger. Model selection for weakly dependent time series forecasting. *Bernoulli*, 18(3):883–913, 2012.

Pierre Alquier, James Ridgway, and Nicolas Chopin. On the properties of variational approximations of Gibbs posteriors. *Journal of Machine Learning Research*, 17(236): 1–41, 2016.

Amiran Ambroladze, Emilio Parrado-Hernández, and John Shawe-Taylor. Tighter PAC-Bayes bounds. *Advances in Neural Information Processing Systems [NIPS]*, pages 9–16, 2007.

Jean-Yves Audibert. A better variance control for PAC-Bayesian classification. Preprint, 2004.

Pranjal Awasthi, Satyen Kale, Stefani Karp, and Mehryar Mohri. PAC-Bayes learning bounds for sample-dependent priors. In *Advances in Neural Information Processing Systems [NeurIPS]*, 2020.

David Barber and Christopher M. Bishop. Ensemble learning for multi-layer networks. In *Advances in Neural Information Processing Systems [NIPS]*, pages 395–401, 1997.

Luc Bégin, Pascal Germain, François Laviolette, and Jean-Francis Roy. PAC-Bayesian theory for transductive learning. In *International Conference on Artificial Intelligence and Statistics [AISTATS]*, pages 105–113. PMLR, 2014.

Luc Bégin, Pascal Germain, François Laviolette, and Jean-Francis Roy. PAC-Bayesian bounds based on the Rényi divergence. In *International Conference on Artificial Intelligence and Statistics [AISTATS]*, pages 435–444. PMLR, 2016.

Pier Giovanni Bissiri, Chris C. Holmes, and Stephen G. Walker. A general framework for updating belief distributions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(5):1103–1130, 2016.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *International Conference on Machine Learning [ICML]*, pages 1613–1622. PMLR, 2015.

Vladimir I. Bogachev. *Gaussian Measures*. American Mathematical Society, 1998.

Bastian Boll, Alexander Zeilmann, Stefania Petra, and Christoph Schnörr. Self-Certifying Classification by Linearized Deep Assignment. arXiv:2201.11162, 2022.

Stéphane Boucheron, Olivier Bousquet, and Gábor Lugosi. Theory of classification: A survey of some recent advances. *ESAIM: Probability and Statistics*, 9:323–375, 2005.

Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press, 2013.

Olivier Bousquet and André Elisseeff. Stability and generalisation. *Journal of Machine Learning Research*, 2:499–526, 2002.

Olivier Bousquet, Yegor Klochkov, and Nikita Zhivotovskiy. Sharper bounds for uniformly stable algorithms. In *Conference on Learning Theory [COLT]*, pages 610–626. PMLR, 2020.

Wray L. Buntine and Andreas S. Weigend. Bayesian back-propagation. *Complex Systems*, 5(6):603–643, 1991.

Olivier Catoni. *Statistical Learning Theory and Stochastic Optimization: Ecole d'Eté de Probabilités de Saint-Flour XXXI-2001*, volume 1851 of *Lecture Notes in Mathematics*. Springer, 2004.

Olivier Catoni. *PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning*, volume 56 of *IMS Lecture Notes-Monograph Series*. Institute of Mathematical Statistics, 2007.

Alain Celisse and Benjamin Guedj. Stability revisited: New generalisation bounds for the leave-one-out. arXiv:1608.06412, 2016.

Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. In *International Workshop on Artificial Intelligence and Statistics [AISTATS]*, pages 57–64. PMLR, 2005.

Badr-Eddine Chérief-Abdellatif, Yuyang Shi, Arnaud Doucet, and Benjamin Guedj. On PAC-Bayesian reconstruction guarantees for VAEs. arXiv:2202.11455, 2022.

Eugenio Clerico, George Deligiannidis, and Arnaud Doucet. Conditional Gaussian PAC-Bayes. arXiv:2110.11886, 2021a.

Eugenio Clerico, George Deligiannidis, and Arnaud Doucet. Wide stochastic networks: Gaussian limit and PAC-Bayesian training. arXiv:2106.09798, 2021b.

Imre Csiszár. I-divergence geometry of probability distributions and minimization problems. *The Annals of Probability*, 3(1):146–158, 1975.

Arnak Dalalyan and Alexandre B. Tsybakov. Aggregation by exponential weighting and sharp oracle inequalities. In *Conference on Learning Theory [COLT]*, pages 97–111. Springer, 2007.

Arnak Dalalyan and Alexandre B. Tsybakov. Aggregation by exponential weighting, sharp PAC-Bayesian bounds and sparsity. *Machine Learning*, 72(1-2):39–61, 2008.

Luc Devroye, László Györfi, and Gabor Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1997.

Monroe D. Donsker and S.R. Srinivasa Varadhan. Asymptotic evaluation of certain Markov process expectations for large time, I. *Communications on Pure and Applied Mathematics*, 28(1):1–47, 1975.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toni Pitassi, Omer Reingold, and Aaron Roth. Generalization in adaptive data analysis and holdout reuse. In *Advances in Neural Information Processing Systems [NIPS]*, pages 2350–2358, 2015a.

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. Preserving statistical validity in adaptive data analysis. In *Symposium on Theory of Computing [STOC]*, pages 117–126. ACM, 2015b.

Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In *Conference on Uncertainty in Artificial Intelligence [UAI]*. AUAI Press, 2017.

Gintare Karolina Dziugaite and Daniel M. Roy. Data-dependent PAC-Bayes priors via differential privacy. In *Advances in Neural Information Processing Systems [NeurIPS]*, pages 8440–8450, 2018a.

Gintare Karolina Dziugaite and Daniel M. Roy. Entropy-SGD optimizes the prior of a PAC-Bayes bound: Generalization properties of Entropy-SGD and data-dependent priors. In *International Conference on Machine Learning [ICML]*, pages 1376–1385. PMLR, 2018b.

Gintare Karolina Dziugaite, Kyle Hsu, Waseem Gharbieh, Gabriel Arpino, and Daniel M. Roy. On the role of data in PAC-Bayes bounds. In *International Conference on Artificial Intelligence and Statistics [AISTATS]*, pages 604–612. PMLR, 2021.

Stewart N. Ethier and Thomas G. Kurtz. *Markov processes: characterization and convergence*. Wiley, 1986.

Alec Farid and Anirudha Majumdar. Generalization Bounds for Meta-Learning via PAC-Bayes and Uniform Stability. In *Advances in Neural Information Processing Systems [NeurIPS]*, 2021.

Andrew Y.K. Foong, Wessel P. Bruinsma, David R. Burt, and Richard E. Turner. How Tight Can PAC-Bayes be in the Small Data Regime? In *Advances in Neural Information Processing Systems [NeurIPS]*, 2021.

Dylan J. Foster, Spencer Greenberg, Satyen Kale, Haipeng Luo, Mehryar Mohri, and Karthik Sridharan. Hypothesis set stability and generalization. In *Advances in Neural Information Processing Systems [NeurIPS]*, pages 6729–6739, 2019.

Charles W. Fox and Stephen J. Roberts. A tutorial on variational Bayesian inference. *Artificial Intelligence Review*, 38(2):85–95, 2012.

Yoav Freund. Self bounding learning algorithms. In *Computational Learning Theory [COLT]*, pages 247–258. ACM, 1998.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning [ICML]*, pages 1050–1059. PMLR, 2016.

Pascal Germain, Alexandre Lacasse, François Laviolette, and Mario Marchand. PAC-Bayesian learning of linear classifiers. In *International Conference on Machine Learning [ICML]*, pages 353–360. ACM, 2009.

Pascal Germain, Amaury Habrard, François Laviolette, and Emilie Morvant. A PAC-Bayesian approach for domain adaptation with specialization to linear classifiers. In *International Conference on Machine Learning [ICML]*, pages 738–746. PMLR, 2013.

Pascal Germain, Alexandre Lacasse, Francois Laviolette, Mario Marchand, and Jean-Francis Roy. Risk Bounds for the majority vote: From a PAC-Bayesian analysis to a learning algorithm. *Journal of Machine Learning Research*, 16(26):787–860, 2015.

Pascal Germain, Francis Bach, Alexandre Lacoste, and Simon Lacoste-Julien. PAC-Bayesian theory meets Bayesian inference. In *Advances in Neural Information Processing Systems [NIPS]*, pages 1884–1892, 2016a.

Pascal Germain, Amaury Habrard, François Laviolette, and Emilie Morvant. A new PAC-Bayesian perspective on domain adaptation. In *International Conference on Machine Learning [ICML]*, pages 859–868. PMLR, 2016b.

Pascal Germain, Amaury Habrard, François Laviolette, and Emilie Morvant. PAC-Bayes and domain adaptation. *Neurocomputing*, 379:379–397, 2020.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT press, 2016.

Peter Grunwald, Thomas Steinke, and Lydia Zakynthinou. PAC-Bayes, MAC-Bayes and Conditional Mutual Information: Fast rate bounds that handle general VC classes. In *Conference on Learning Theory [COLT]*, pages 2217–2247. PMLR, 2021.

Peter D. Grünwald and Nishant A. Mehta. A tight excess risk bound via a unified PAC-Bayesian-Rademacher-Shtarkov-MDL complexity. In *International Conference on Algorithmic Learning Theory [ALT]*, volume 98, pages 433–465. PMLR, 2019.

Benjamin Guedj. A primer on PAC-Bayesian learning. In Emmanuel Breuillard, editor, *Congrès de la Société Mathématique de France, Collection SMF*, volume 33, 2019.

Steve Hanneke and Aryeh Kontorovich. Optimality of SVM: Novel proofs and tighter bounds. *Theoretical Computer Science*, 796:99–113, 2019.

Fredrik Hellström and Giuseppe Durisi. Generalization bounds via information density and conditional information density. *IEEE Journal on Selected Areas in Information Theory*, 1(3):824–839, 2020.

José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International Conference on Machine Learning [ICML]*, pages 1861–1869. PMLR, 2015.

Matthew Holland. PAC-Bayes under potentially heavy tails. In *Advances in Neural Information Processing Systems [NeurIPS]*, pages 2715–2724, 2019.

Martin Jankowiak and Fritz Obermeyer. Pathwise derivatives beyond the reparameterization trick. In *International Conference on Machine Learning [ICML]*, pages 2240–2249. PMLR, 2018.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. In *Learning in Graphical Models*, volume 89 of *NATO ASI Series*, pages 105–161. Springer, 1998.

Olav Kallenberg. *Random Measures, Theory and Applications*. Springer, 2017.

Joseph Keshet, David McAllester, and Tamir Hazan. PAC-Bayesian approach for minimization of phoneme error rate. In *IEEE International Conference on Acoustics, Speech and Signal Processing [ICASSP]*, pages 2224–2227. IEEE, 2011.

Joseph Keshet, Subhransu Maji, Tamir Hazan, and Tommi Jaakkola. Perturbation models and PAC-Bayesian generalization bounds. In *Perturbations, Optimization, and Statistics*, pages 289–309. MIT Press, 2017.

Mohammad Emtiyaz Khan and Wu Lin. Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. In *International Conference on Artificial Intelligence and Statistics [AISTATS]*, pages 878–887. PMLR, 2017.

Vladimir Koltchinskii and Dmitriy Panchenko. Rademacher processes and bounding the risk of function learning. In *High dimensional probability II*, pages 443–457. Springer, 2000.

Aryeh Kontorovich. Concentration in unbounded metric spaces and algorithmic stability. In *ICML*, pages 28–36, 2014.

Ilja Kuzborskij and Csaba Szepesvári. Efron-Stein PAC-Bayesian inequalities. arXiv:1909.01931, 2019.

Ilja Kuzborskij, Nicolò Cesa-Bianchi, and Csaba Szepesvári. Distribution-dependent analysis of Gibbs-ERM principle. In *Conference on Learning Theory [COLT]*, volume 99, pages 2028–2054. PMLR, 2019.

Alexandre Lacasse, François Laviolette, Mario Marchand, Pascal Germain, and Nicolas Usunier. PAC-Bayes bounds for the risk of the majority vote and the variance of the Gibbs classifier. In *Advances in Neural Information Processing Systems [NIPS]*, pages 769–776, 2006.

Xinjie Lan, Xin Guo, and Kenneth E. Barner. PAC-Bayesian generalization bounds for multilayer perceptrons. arXiv:2006.08888, 2020.

John Langford. Tutorial on practical prediction theory for classification. *Journal of Machine Learning Research*, 6(Mar):273–306, 2005.

John Langford and Avrim Blum. Microchoice bounds and self bounding learning algorithms. *Machine Learning*, 51(2):165–179, 2003.

John Langford and Rich Caruana. (Not) bounding the true error. In *Advances in Neural Information Processing Systems [NIPS]*, pages 809–816, 2001.

John Langford and Matthias Seeger. Bounds for averaging classifiers. Technical Report CMU-CS-01-102, Carnegie Mellon University, 2001.

Gaël Letarte, Pascal Germain, Benjamin Guedj, and François Laviolette. Dichotomize and generalize: PAC-Bayesian binary activated deep neural networks. In *Advances in Neural Information Processing Systems [NeurIPS]*, pages 6872–6882, 2019.

Guy Lever, François Laviolette, and John Shawe-Taylor. Distribution-dependent PAC-Bayes priors. In *International Conference on Algorithmic Learning Theory [ALT]*, pages 119–133. Springer, 2010.

Guy Lever, François Laviolette, and John Shawe-Taylor. Tighter PAC-Bayes bounds through distribution-dependent priors. *Theoretical Computer Science*, 473:4–28, 2013.

Tongliang Liu, Gábor Lugosi, Gergely Neu, and Dacheng Tao. Algorithmic stability and hypothesis complexity. In *ICML*, pages 2159–2167, 2017.

Ben London. A PAC-Bayesian analysis of randomized learning with application to stochastic gradient descent. In *Advances in Neural Information Processing Systems [NIPS]*, pages 2931–2940, 2017.

Ben London, Bert Huang, Ben Taskar, and Lise Getoor. Collective stability in structured prediction: Generalization from one example. In *Proc. of the 30th International Conference on Machine Learning*, pages 828–836, 2013.

Stephan S. Lorenzen, Christian Igel, and Yevgeny Seldin. On PAC-Bayesian bounds for random forests. *Machine Learning*, 108(8-9):1503–1522, 2019.

Christos Louizos and Max Welling. Structured and efficient variational deep learning with matrix gaussian posteriors. In *International Conference on Machine Learning [ICML]*, pages 1708–1716. PMLR, 2016.

Gábor Lugosi and Miroslaw Pawlak. On the posterior-probability estimate of the error rate of nonparametric classification rules. *IEEE Transactions on Information Theory*, 40(2):475–481, 1994.

David J.C. MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

Wesley J. Maddox, Pavel Izmailov, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. A simple baseline for Bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems [NeurIPS]*, pages 13132–13143, 2019.

James Martens and Roger Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *International Conference on Machine Learning [ICML]*, pages 2408–2417. PMLR, 2015.

Andrés R. Masegosa, Stephan S. Lorenzen, Christian Igel, and Yevgeny Seldin. Second order PAC-Bayesian bounds for the weighted majority vote. In *Advances in Neural Information Processing Systems [NeurIPS]*, pages 5263–5273, 2020.

Andreas Maurer. A note on the PAC Bayesian theorem. arXiv:cs/0411099, 2004.

David A. McAllester. Some PAC-Bayesian theorems. In *Computational Learning Theory [COLT]*, pages 230–234. ACM, 1998. Also one year later in *Machine Learning* 37(3), pages 355–363, 1999.

David A. McAllester. PAC-Bayesian model averaging. In *Computational Learning Theory [COLT]*, pages 164–170. ACM, 1999.

David A. McAllester. A PAC-Bayesian tutorial with a dropout bound. arXiv:1307.2118, 2013.

Colin McDiarmid. On the method of bounded differences. *Surveys in Combinatorics*, 141(1):148–188, 1989.

Sean P. Meyn and Richard L. Tweedie. *Markov Chains and Stochastic Stability*. Cambridge University Press, 2nd. edition, 2009.

Zakaria Mhammedi, Peter D. Grunwald, and Benjamin Guedj. PAC-Bayes un-expected Bernstein inequality. In *Advances in Neural Information Processing Systems [NeurIPS]*, pages 12202–12213, 2019.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte Carlo gradient estimation in machine learning. *Journal of Machine Learning Research*, 21 (132):1–62, 2020.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. MIT press, 2018.

Grégoire Montavon, Genevieve B. Orr, and Klaus-Robert Müller, editors. *Neural Networks: Tricks of the Trade*, volume 7700 of *Lecture Notes in Computer Science*. Springer, 2012. (2nd edition).

Radford M. Neal. Bayesian training of backpropagation networks by the hybrid Monte Carlo method. Technical Report CRG-TR-92-1, University of Toronto, 1992.

Jeffrey Negrea, Mahdi Haghifam, Gintare Karolina Dziugaite, Ashish Khisti, and Daniel M Roy. Information-theoretic generalization bounds for SGLD via data-dependent estimates. In *Advances in Neural Information Processing Systems [NeurIPS]*, 2019.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems [NIPS]*, pages 5947–5956, 2017.

Behnam Neyshabur, Srinadh Bhojanapalli, and Nathan Srebro. A PAC-Bayesian approach to spectrally-normalized margin bounds for neural networks. In *International Conference on Learning Representations [ICLR]*, 2018.

Asaf Noy and Koby Crammer. Robust forward algorithms via PAC-Bayes and Laplace distributions. In *International Conference on Artificial Intelligence and Statistics [AISTATS]*, pages 678–686. PMLR, 2014.

Kazuki Osawa, Siddharth Swaroop, Mohammad Emtiyaz Khan, Anirudh Jain, Runa Eschenhagen, Richard E. Turner, and Rio Yokota. Practical deep learning with Bayesian principles. In *Advances in Neural Information Processing Systems [NeurIPS]*, pages 4289–4301, 2019.

Emilio Parrado-Hernández, Amiran Ambroladze, John Shawe-Taylor, and Shiliang Sun. PAC-Bayes bounds with data dependent priors. *Journal of Machine Learning Research*, 13(112):3507–3531, 2012.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12(85):2825–2830, 2011.

Anastasia Pentina and Christoph H. Lampert. A PAC-Bayesian bound for lifelong learning. In *International Conference on Machine Learning [ICML]*, pages 991–999, 2014.

María Pérez-Ortiz, Omar Rivasplata, John Shawe-Taylor, and Csaba Szepesvári. Towards self-certified learning: Probabilistic neural networks trained by PAC-Bayes with Backprop. NeurIPS 2020 Workshop - Beyond Backpropagation, 2020.

Maria Pérez-Ortiz, Omar Rivasplata, Emilio Parrado-Hernández, Benjamin Guedj, and John Shawe-Taylor. Progress in Self-Certified Neural Networks. arXiv:2111.07737, NeurIPS 2021 Workshop on Bayesian Deep Learning, 2021a.

María Pérez-Ortiz, Omar Rivasplata, John Shawe-Taylor, and Csaba Szepesvári. Tighter risk certificates for neural networks. *Journal of Machine Learning Research*, 22(227): 1–40, 2021b.

Konstantinos Pitas. Dissecting non-vacuous generalization bounds based on the mean-field approximation. In *International Conference on Machine Learning [ICML]*, pages 7739–7749. PMLR, 2020.

John C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, pages 185–208. MIT Press, Cambridge MA, 1999.

Robert Price. A useful theorem for nonlinear devices having Gaussian inputs. *IRE Transactions on Information Theory*, 4(2):69–72, 1958.

Maxim Raginsky, Alexander Rakhlin, and Matus Telgarsky. Non-convex Learning via Stochastic Gradient Langevin Dynamics: A Nonasymptotic Analysis. In *Conference on Learning Theory [COLT]*, 2017.

Emmanuel Rio. Inégalités de Hoeffding pour les fonctions lipschitziennes de suites dépendantes. *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, 330(10):905–908, 2000.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable Laplace approximation for neural networks. In *International Conference on Learning Representations [ICLR]*, 2018.

Omar Rivasplata, Emilio Parrado-Hernández, John Shawe-Taylor, Shiliang Sun, and Csaba Szepesvári. PAC-Bayes bounds for stable algorithms with instance-dependent priors. In *Advances in Neural Information Processing Systems [NeurIPS]*, pages 9214–9224, 2018.

Omar Rivasplata, Vikram M. Tankasali, and Csaba Szepesvári. PAC-Bayes with Backprop. arXiv:1908.07380, 2019.

Omar Rivasplata, Ilja Kuzborskij, Csaba Szepesvári, and John Shawe-Taylor. PAC-Bayes analysis beyond the usual bounds. In *Advances in Neural Information Processing Systems [NeurIPS]*, pages 16833–16845, 2020.

Francisco R. Ruiz, Michalis Titsias, and David Blei. The generalized reparameterization gradient. In *Advances in Neural Information Processing Systems [NIPS]*, pages 460–468, 2016.

Matthias Seeger. PAC-Bayesian generalization error bounds for Gaussian process classification. *Journal of Machine Learning Research*, 3:233–269, 2002.

Yevgeny Seldin and Naftali Tishby. PAC-Bayesian analysis of co-clustering and beyond. *Journal of Machine Learning Research*, 11:3595–3646, 2010.

Vera Shalaeva, Alireza Fakhrizadeh Esfahani, Pascal Germain, and Mihaly Petreczky. Improved PAC-Bayesian bounds for linear regression. In *AAAI Conference on Artificial Intelligence [AAAI]*, 2020.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms.* Cambridge University Press, Cambridge, 2014.

John Shawe-Taylor and Robert C. Williamson. A PAC analysis of a Bayesian estimator. In *Computational Learning Theory [COLT]*, pages 2–9. ACM, 1997.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations [ICLR]*, 2015.

Thomas Steinke and Lydia Zakynthinou. Reasoning about generalization via conditional mutual information. In *Conference on Learning Theory [COLT]*, pages 3437–3452. PMLR, 2020.

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

David Stutz, Ali Taylan Cemgil, Arnaud Doucet, et al. Learning optimal conformal classifiers. In *International Conference on Learning Representations [ICLR]*, 2022. Accessible in arXiv:2110.09192.

Sanjay Thakur, Herke Van Hoof, Gunshi Gupta, and David Meger. Unifying variational inference and PAC-Bayes for supervised learning that scales. arXiv:1910.10367, 2019.

Niklas Thiemann, Christian Igel, Olivier Wintenberger, and Yevgeny Seldin. A strongly quasiconvex PAC-Bayesian bound. In *International Conference on Algorithmic Learning Theory [ALT]*, pages 466–492. PMLR, 2017.

Ilya O. Tolstikhin and Yevgeny Seldin. PAC-Bayes-Empirical-Bernstein inequality. In *Advances in Neural Information Processing Systems [NIPS]*, pages 109–117, 2013.

Tim van Erven. PAC-Bayes mini-tutorial: A continuous union bound. arXiv:1405.1580, 2014.

Vladimir N. Vapnik. Principles of risk minimization for learning theory. In *Advances in Neural Information Processing Systems [NIPS]*, pages 831–838. Morgan-Kaufmann, 1992.

Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

Vladimir N. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

Vladimir N. Vapnik and Alexey Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264, 1971.

Paul Viallard, Rémi Emonet, Pascal Germain, Amaury Habrard, and Emilie Morvant. Interpreting neural networks as majority votes through the PAC-Bayesian theory. NeurIPS 2019 Workshop on Machine Learning with Guarantees, 2019.

Paul Viallard, Eric Guillaume VIDOT, Amaury Habrard, and Emilie Morvant. A PAC-Bayes Analysis of Adversarial Robustness. In *Advances in Neural Information Processing Systems [NeurIPS]*, 2021.

Vladimir Vovk, Alexander Gammerman, and Glenn Shafer. *Algorithmic Learning in a Random World*. Springer, 2005.

Max Welling and Yee Whye Teh. Bayesian learning via Stochastic Gradient Langevin dynamics. In *International Conference on Machine Learning [ICML]*, pages 681–688. PMLR, 2011.

Yi-Shan Wu, Andres Masegosa, Stephan Lorenzen, Christian Igel, and Yevgeny Seldin. Chebyshev-Cantelli PAC-Bayes-Bennett Inequality for the Weighted Majority Vote. In *Advances in Neural Information Processing Systems [NeurIPS]*, 2021.

Aolin Xu and Maxim Raginsky. Information-theoretic analysis of generalization capability of learning algorithms. In *Advances in Neural Information Processing Systems [NIPS]*, pages 2524–2533, 2017.

Valentina Zantedeschi, Paul Viallard, Emilie Morvant, Rémi Emonet, Amaury Habrard, Pascal Germain, and Benjamin Guedj. Learning Stochastic Majority Votes by Minimizing a PAC-Bayes Generalization Bound. In *Advances in Neural Information Processing Systems [NeurIPS]*, 2021.

Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P. Adams, and Peter Orbanz. Non-vacuous generalization bounds at the ImageNet scale: A PAC-Bayesian compression approach. In *International Conference on Learning Representations [ICLR]*, 2019.