A (condensed) primer on PAC-Bayesian Learning

followed by

News from the PAC-Bayes frontline

Benjamin Guedj
https://bguedj.github.io

Chargé de recherche Inria, LNE MODAL
Principal research fellow, University College London
Visiting researcher, The Alan Turing Institute
Scientific director, The Inria London Programme

MODAL Seminar
October 20, 2020
The London branch of Modal
1h20 (and 14 days of quarantine) far from here...
What to expect

I will...

- Provide an overview of what PAC-Bayes is
- Illustrate its flexibility and relevance to tackle modern machine learning tasks, and rethink generalisation
- Cover key ideas and a few results
- Briefly present a sample of recent contributions from my group

I won’t...

- Cover all of our ICML 2019 tutorial!
  See https://bguedj.github.io/icml2019/index.html
- Cover our NIPS 2017 workshop ”(Almost) 50 Shades of Bayesian Learning: PAC-Bayesian trends and insights”
  See https://bguedj.github.io/nips2017/
Take-home message

PAC-Bayes is a generic framework to efficiently rethink generalisation for numerous machine learning algorithms. It leverages the flexibility of Bayesian learning and allows to derive new learning algorithms.

PhD students, postdocs, tenured researchers, visiting positions
Through the Centre for AI at UCL,
and through the newly founded Inria London Programme
Part I

A Primer on PAC-Bayesian Learning
(short version of our ICML 2019 tutorial)

Learning is to be able to generalise

From examples, what can a system learn about the underlying phenomenon?

Memorising the already seen data is usually bad \(\rightarrow\) overfitting

Generalisation is the ability to ’perform’ well on unseen data.
Statistical Learning Theory is about high confidence

For a fixed algorithm, function class and sample size, generating random samples $\rightarrow$ distribution of test errors

- Focusing on the mean of the error distribution?
  - can be misleading: learner only has one sample

- **Statistical Learning Theory**: tail of the distribution
  - finding bounds which hold with high probability over random samples of size $m$

- Compare to a statistical test – at 99% confidence level
  - chances of the conclusion not being true are less than 1%

- **PAC**: probably approximately correct (Valiant, 1984)
  - Use a ‘confidence parameter’ $\delta$: $\Pr^m[\text{large error}] \leq \delta$
  - $\delta$ is the probability of being misled by the training set

- Hence **high confidence**: $\Pr^m[\text{approximately correct}] \geq 1 - \delta$
Mathematical formalisation

Learning algorithm $A : \mathcal{Z}^m \rightarrow \mathcal{H}$

- $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$
  - $\mathcal{X}$ = set of inputs
  - $\mathcal{Y}$ = set of outputs (e.g. labels)

- $\mathcal{H}$ = hypothesis class = set of predictors (e.g. classifiers) functions $\mathcal{X} \rightarrow \mathcal{Y}$

Training set (aka sample): $S_m = \{(X_1, Y_1), \ldots, (X_m, Y_m)\}$ a sequence of input-output examples.

- Data-generating distribution $\mathbb{P}$ over $\mathcal{Z}$
- Learner doesn’t know $\mathbb{P}$, only sees the training set
- Examples are i.i.d.: $S_m \sim \mathbb{P}^m$
What to achieve from the sample?

Use the available sample to:

1. learn a predictor
2. certify the predictor’s performance

Learning a predictor:

• algorithm driven by some learning principle
• informed by prior knowledge resulting in inductive bias

Certifying performance:

• what happens beyond the training set
• generalisation bounds

Actually these two goals interact with each other!
Risk (aka error) measures

A loss function $\ell(h(X), Y)$ is used to measure the discrepancy between a predicted output $h(X)$ and the true output $Y$.

Empirical risk: $R_{\text{in}}(h) = \frac{1}{m} \sum_{i=1}^{m} \ell(h(X_i), Y_i)$ (in-sample)

Theoretical risk: $R_{\text{out}}(h) = \mathbb{E}[\ell(h(X), Y)]$ (out-of-sample)

Examples:
- $\ell(h(X), Y) = 1[h(X) \neq Y] : 0$-1 loss (classification)
- $\ell(h(X), Y) = (Y - h(X))^2 :$ square loss (regression)
- $\ell(h(X), Y) = (1 - Yh(X))_{+} :$ hinge loss
- $\ell(h(X), 1) = -\log(h(X)) :$ log loss (density estimation)
- ...
Generalisation

If predictor $h$ does well on the in-sample $(X, Y)$ pairs...
...will it still do well on out-of-sample pairs?

Generalisation gap: $\Delta(h) = R_{\text{out}}(h) - R_{\text{in}}(h)$

Upper bounds: w.h.p. $\Delta(h) \leq \epsilon(m, \delta)$

Lower bounds: w.h.p. $\Delta(h) \geq \tilde{\epsilon}(m, \delta)$

Flavours:
- distribution-free
- algorithm-free
- distribution-dependent
- algorithm-dependent
Why you should care about generalisation bounds

Generalisation bounds are a safety check: give a theoretical guarantee on the performance of a learning algorithm on any unseen data.

\[ R_{\text{out}}(h) \leq R_{\text{in}}(h) + \epsilon(m, \delta) \]

Generalisation bounds:

- may be computed with the training sample only, do not depend on any test sample
- provide a computable control on the error on any unseen data with prespecified confidence
- explain why specific learning algorithms actually work
- and even lead to designing new algorithm which scale to more complex settings
Before PAC-Bayes

- Single hypothesis $h$ (building block):
  
  with probability $\geq 1 - \delta$,  
  $R_{\text{out}}(h) \leq R_{\text{in}}(h) + \sqrt{\frac{1}{2m} \log \left( \frac{1}{\delta} \right)}$.

- Finite function class $\mathcal{H}$ (worst-case approach):
  
  w.p. $\geq 1 - \delta$, $\forall h \in \mathcal{H}$,  
  $R_{\text{out}}(h) \leq R_{\text{in}}(h) + \sqrt{\frac{1}{2m} \log \left( \frac{|\mathcal{H}|}{\delta} \right)}$

- Structural risk minimisation: data-dependent hypotheses $h_i$ associated with prior weight $p_i$
  
  w.p. $\geq 1 - \delta$, $\forall h_i \in \mathcal{H}$,  
  $R_{\text{out}}(h_i) \leq R_{\text{in}}(h_i) + \sqrt{\frac{1}{2m} \log \left( \frac{1}{p_i \delta} \right)}$

- Uncountably infinite function class: VC dimension, Rademacher complexity...

These approaches are suited to analyse the performance of individual functions, and take some account of correlations.

→ Extension: PAC-Bayes allows to consider distributions over hypotheses.
The PAC-Bayes framework

- Before data, fix a distribution $P \in M_1(\mathcal{H}) \triangleright \text{‘prior’}$
- Based on data, learn a distribution $Q \in M_1(\mathcal{H}) \triangleright \text{‘posterior’}$
- Predictions:
  - draw $h \sim Q$ and predict with the chosen $h$.
  - each prediction with a fresh random draw.

The risk measures $R_{\text{in}}(h)$ and $R_{\text{out}}(h)$ are extended by averaging:

$$R_{\text{in}}(Q) \equiv \int_{\mathcal{H}} R_{\text{in}}(h) \, dQ(h) \quad R_{\text{out}}(Q) \equiv \int_{\mathcal{H}} R_{\text{out}}(h) \, dQ(h)$$

$$\text{KL}(Q \| P) = \mathbb{E}_{h \sim Q} \ln \frac{Q(h)}{P(h)}$$ is the Kullback-Leibler divergence.
"Prior": exploration mechanism of $\mathcal{H}$
"Posterior" is the twisted prior after confronting with data
PAC-Bayes bounds vs. Bayesian learning

- **Prior**
  - **PAC-Bayes**: bounds hold for any distribution
  - **Bayes**: prior choice impacts inference

- **Posterior**
  - **PAC-Bayes**: bounds hold for any distribution
  - **Bayes**: posterior uniquely defined by prior and statistical model

- **Data distribution**
  - **PAC-Bayes**: bounds hold for any distribution
  - **Bayes**: randomness lies in the noise model generating the output
A classical PAC-Bayesian bound

**Pre-history:** PAC analysis of Bayesian estimators
*Shawe-Taylor and Williamson (1997); Shawe-Taylor et al. (1998)*

**Birth:** PAC-Bayesian bound
*McAllester (1998, 1999)*

**McAllester Bound**

For any prior $P$, any $\delta \in (0, 1]$, we have

$$
\mathbb{P}^m \left( \forall Q \text{ on } \mathcal{H}: R_{\text{out}}(Q) \leq R_{\text{in}}(Q) + \sqrt{\frac{\text{KL}(Q\|P) + \ln \frac{2\sqrt{m}}{\delta}}{2m}} \right) \geq 1 - \delta,
$$
A flexible framework

Since 1997, PAC-Bayes has been successfully used in many machine learning settings (this list is by no means exhaustive).

**Statistical learning theory**  Shawe-Taylor and Williamson (1997); McAllester (1998, 1999, 2003a,b); Seeger (2002, 2003); Maurer (2004); Catoni (2004, 2007); Audibert and Bousquet (2007); Thiemann et al. (2017); Guedj (2019); Mhammedi et al. (2019, 2020); Guedj and Pujol (2019); Haddouche et al. (2020)

**SVMs & linear classifiers**  Langford and Shawe-Taylor (2002); McAllester (2003a); Germain et al. (2009a)

**Supervised learning algorithms** reinterpreted as bound minimizers  
Ambroladze et al. (2007); Shawe-Taylor and Hardoon (2009); Germain et al. (2009b)

**High-dimensional regression**  Alquier and Lounici (2011); Alquier and Biau (2013); Guedj and Alquier (2013); Li et al. (2013); Guedj and Robbiano (2018)

**Classification**  Langford and Shawe-Taylor (2002); Catoni (2004, 2007); Lacasse et al. (2007); Parrado-Hernández et al. (2012)
A flexible framework

Transductive learning, domain adaptation  Derbeko et al. (2004); Bégin et al. (2014); Germain et al. (2016b); Nozawa et al. (2020)

Non-iid or heavy-tailed data  Lever et al. (2010); Seldin et al. (2011, 2012); Alquier and Guedj (2018); Holland (2019)

Density estimation  Seldin and Tishby (2010); Higgs and Shawe-Taylor (2010)

Reinforcement learning  Fard and Pineau (2010); Fard et al. (2011); Seldin et al. (2011, 2012); Ghavamzadeh et al. (2015)

Sequential learning  Gerchinovitz (2011); Li et al. (2018)

Algorithmic stability, differential privacy  London et al. (2014); London (2017); Dziugaite and Roy (2018a, b); Rivasplata et al. (2018)

Deep neural networks  Dziugaite and Roy (2017); Neyshabur et al. (2017); Zhou et al. (2019); Letarte et al. (2019); Biggs and Guedj (2020)

...
PAC-Bayes-inspired learning algorithms

With an arbitrarily high probability and for any posterior distribution $Q$, ...

\[ \text{Error on unseen data} \leq \text{Error on sample} + \text{complexity term} \]
\[ R_{\text{out}}(Q) \leq R_{\text{in}}(Q) + F(Q, \cdot) \]

This defines a principled strategy to obtain new learning algorithms:

\[ h \sim Q^* \]
\[ Q^* \in \arg \inf_{Q \ll P} \left\{ R_{\text{in}}(Q) + F(Q, \cdot) \right\} \]

(optimisation problem which can be solved or approximated by [stochastic] gradient descent-flavoured methods, Monte Carlo Markov Chain, (generalized) variational inference...) SVMs, KL-regularized Adaboost, exponential weights are all minimisers of PAC-Bayes bounds.
Variational definition of KL-divergence (Csiszár, 1975; Donsker and Varadhan, 1975; Catoni, 2004).

Let \((A, \mathcal{A})\) be a measurable space.

(i) For any probability \(P\) on \((A, \mathcal{A})\) and any measurable function \(\phi : A \rightarrow \mathbb{R}\) such that \(\int (\exp \circ \phi) \, dP < \infty\),

\[
\log \int (\exp \circ \phi) \, dP = \sup_{Q \ll P} \left\{ \int \phi \, dQ - KL(Q, P) \right\}.
\]

(ii) If \(\phi\) is upper-bounded on the support of \(P\), the supremum is reached for the Gibbs distribution \(G\) given by

\[
\frac{dG}{dP}(a) = \frac{\exp \circ \phi(a)}{\int (\exp \circ \phi) \, dP}, \quad a \in A.
\]
\[
\log \int (\exp \circ \phi) dP = \sup_{Q \ll P} \left\{ \int \phi dQ - \KL(Q, P) \right\}, \quad \frac{dG}{dP} = \frac{\exp \circ \phi}{\int (\exp \circ \phi) dP}.
\]

Proof: let \( Q \ll P \) and \( P \ll Q \).

\[
-\KL(Q, G) = -\int \log \left( \frac{dQ}{dP} \frac{dP}{dG} \right) dQ
= -\int \log \left( \frac{dQ}{dP} \right) dQ + \int \log \left( \frac{dG}{dP} \right) dQ
= -\KL(Q, P) + \int \phi dQ - \log \int (\exp \circ \phi) dP.
\]

\( \KL(\cdot, \cdot) \) is non-negative, \( Q \mapsto -\KL(Q, G) \) reaches its max. in \( Q = G \):

\[
0 = \sup_{Q \ll P} \left\{ \int \phi dQ - \KL(Q, P) \right\} - \log \int (\exp \circ \phi) dP.
\]

Let \( \lambda > 0 \) and take \( \phi = -\lambda R_{\text{in}} \),

\[
Q_{\lambda} \propto \exp (-\lambda R_{\text{in}}) P = \arg \inf_{Q \ll P} \left\{ R_{\text{in}}(Q) + \frac{\KL(Q, P)}{\lambda} \right\}.
\]
Recap

What we’ve seen so far

- Statistical learning theory is about **high confidence control of generalisation**
- PAC-Bayes is a **generic, powerful tool** to derive generalisation bounds...
- ... and invent **new learning algorithms with a Bayesian flavour**
- PAC-Bayes mixes tools from **statistics, probability theory, optimisation**, and is now quickly re-emerging as a key theory and practical framework in **machine learning**

What is coming next

- What we’ve been up to with PAC-Bayes recently!
Part II

News from the PAC-Bayes frontline

Learning with non-iid or heavy-tailed data

We drop the iid and bounded loss assumptions. For any integer $q$,

$$
M_q := \int \mathbb{E} (|R_{\text{in}}(h) - R_{\text{out}}(h)|^q) \, dP(h).
$$

Csiszár $f$-divergence: let $f$ be a convex function with $f(1) = 0$,

$$
D_f(Q, P) = \int f \left( \frac{dQ}{dP} \right) \, dP
$$

when $Q \ll P$ and $D_f(Q, P) = +\infty$ otherwise.

The KL is given by the special case $\text{KL}(Q\|P) = D_{x \log(x)}(Q, P)$.

Power function: $\phi_p : x \mapsto x^p$. 
PAC-Bayes with $f$-divergences

Fix $p > 1$, $q = \frac{p}{p-1}$ and $\delta \in (0, 1)$. With probability at least $1 - \delta$ we have for any distribution $Q$

$$|R_{\text{out}}(Q) - R_{\text{in}}(Q)| \leq \left( \frac{\mathcal{M}_q}{\delta} \right)^{\frac{1}{q}} \left( D_{\Phi_{p-1}}(Q, P) + 1 \right)^{\frac{1}{p}}.$$ 

The bound decouples

- the moment $\mathcal{M}_q$ (which depends on the distribution of the data)
- and the divergence $D_{\Phi_{p-1}}(Q, P)$ (measure of complexity).

Corollary: with probability at least $1 - \delta$, for any $Q$,

$$R_{\text{out}}(Q) \leq R_{\text{in}}(Q) + \left( \frac{\mathcal{M}_q}{\delta} \right)^{\frac{1}{q}} \left( D_{\Phi_{p-1}}(Q, P) + 1 \right)^{\frac{1}{p}}.$$ 

Again, strong incitement to define the "optimal" posterior as the minimizer of the right-hand side!

For $p = q = 2$, w.p. $\geq 1 - \delta$, $R_{\text{out}}(Q) \leq R_{\text{in}}(Q) + \sqrt{\frac{\nu}{m\delta} \int \left( \frac{dQ}{dP} \right)^2 dP}$. 
Proof

Let $\Delta(h) := |R_{in}(h) - R_{out}(h)|$.

\[
\left| \int R_{out} dQ - \int R_{in} dQ \right| \\
\leq \int \Delta dQ \\
= \int \Delta \frac{dQ}{dP} dP \\
\leq \left( \int \Delta^q dP \right)^{\frac{1}{q}} \left( \int \left( \frac{dQ}{dP} \right)^p dP \right)^{\frac{1}{p}} \\
\leq \left( \frac{\mathbb{E} \int \Delta^q dP}{\delta} \right)^{\frac{1}{q}} \left( \int \left( \frac{dQ}{dP} \right)^p dP \right)^{\frac{1}{p}} \\
= \left( \frac{M_q}{\delta} \right)^{\frac{1}{q}} \left( D_{\Phi_{p-1}}(Q, P) + 1 \right)^{\frac{1}{p}}.
\]
Previous attempts to circumvent the bounded range assumption on the loss in PAC-Bayes:

- Assume sub-gaussian or sub-exponential tails of the loss (Alquier et al., 2016; Germain et al., 2016a) - requires knowledge of additional parameters.

- Analysis for heavy-tailed losses, e.g. Alquier and Guedj (2018) proposed a polynomial moment-dependent bound with $f$-divergences, while Holland (2019) devised an exponential bound which assumes that the second (uncentered) moment of the loss is bounded by a constant (with a truncated risk estimator).

- Kuzborskij and Szepesvári (2019) do not assume boundedness of the loss, but rather control higher-order moments of the generalization gap through the Efron-Stein variance proxy.

We investigate a different route.

We introduce the **HYPothesis-dependent range** condition (HYPE) which means the loss is upper bounded by a hypothesis-only-dependent term. Designed to be user-friendly!
Novelty lies in the proof technique: we adapt the notion of **self-bounding function**, introduced by Boucheron et al. (2000) and further developed in Boucheron et al. (2004, 2009).

**Definition**

A loss function $\ell : \mathcal{H} \times \mathcal{Z} \to \mathbb{R}^+$ is said to satisfy the **hypothesis-dependent range** $(\text{HYPE})$ condition if there exists a function $K : \mathcal{H} \to \mathbb{R}^+\setminus\{0\}$ such that

$$\sup_{z \in \mathcal{Z}} \ell(h, z) \leq K(h)$$

for any predictor $h$. We then say that $\ell$ is $\text{HYPE}(K)$ compliant.
Theorem 2.1 from Germain et al., 2009a

For any $P$, for any convex function $D : \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}$, for any $\alpha \in \mathbb{R}$ and for any $\delta \in [0 : 1]$, we have with probability at least $1 - \delta$, for any $Q$ such that $Q \ll P$ and $P \ll Q$:

$$D \left( R_{\text{in}}(Q), R_{\text{out}}(Q) \right) \leq \frac{1}{m^\alpha} \left( \text{KL}(Q \| P) + \log \left( \frac{1}{\delta} \mathbb{E}_{h \sim P} \mathbb{E} e^{m^\alpha D(R_m(h), R(h))} \right) \right).$$

Goal is to control $\mathbb{E} \left[ e^{m^\alpha \Delta(h)} \right]$ for a fixed $h$. The technique we use is based on the notion of $(a, b)$-self-bounding functions defined in Boucheron et al. (2009, Definition 2).
A function \( f : \mathcal{X}^m \to \mathbb{R} \) is said to be \((a, b)\)-self-bounding with \((a, b) \in (\mathbb{R}^+)^2 \setminus \{(0, 0)\}\), if there exists \( f_i : \mathcal{X}^{m-1} \to \mathbb{R} \) for every \( i \in \{1..m\} \) such that \( \forall i \in \{1..m\} \) and \( x \in \mathcal{X} \):

\[
0 \leq f(x) - f_i(x^{(i)}) \leq 1
\]

and

\[
\sum_{i=1}^{m} f(x) - f_i(x^{(i)}) \leq af(x) + b
\]

where for all \( 1 \leq i \leq m \), the removal of the \( i \)th entry is \( x^{(i)} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_m) \). We denote by \( SB(a, b) \) the class of \((a, b)\)-self-bounding functions.
Let $Z = g(X_1, \ldots, X_m)$ where $X_1, \ldots, X_m$ are independent (not necessarily identically distributed) $X$-valued random variables. Assume that $\mathbb{E}[Z] < +\infty$. If $g \in \mathcal{SB}(a, b)$, then defining $c = (3a - 1)/6$, for any $s \in [0; c_{+}^{-1})$ we have:

$$\log \left( \mathbb{E} \left[ e^{s(Z - \mathbb{E}[Z])} \right] \right) \leq \frac{(a \mathbb{E}[Z] + b) s^2}{2(1 - c_{+} s)}.$$
Theorem

Let \( h \in \mathcal{H} \) be a fixed predictor and \( \alpha \in \mathbb{R} \). If the loss function \( \ell \) is HYPE\((K)\) compliant, then for \( \Delta(h) = R_{\text{out}}(h) - R_{\text{in}}(h) \) we have:

\[
\mathbb{E} \left[ e^{m\alpha \Delta(h)} \right] \leq \exp \left( \frac{K(h)^2}{2m^{1-2\alpha}} \right).
\]

Illustrates the strength of our approach: we traded on the right-hand side of the bound the large exponent \( m^\alpha K(h)^2 \) (naive bound) for \( \frac{K(h)^2}{m^{1-2\alpha}} \), the latter being much smaller when \( \alpha \leq 1 \).
Theorem

Let the loss $\ell$ be $\text{HYPE}(K)$ compliant. For any $P$, for any $\alpha \in \mathbb{R}$ and for any $\delta \in [0 : 1]$, we have with probability at least $1 - \delta$, for any $Q$ such that $Q \ll P$ and $P \ll Q$:

$$R_{\text{out}}(Q) \leq R_{\text{in}}(Q) + \frac{\text{KL}(Q \| P) + \log \left( \frac{1}{\delta} \right)}{m^\alpha} \left( + \frac{1}{m^\alpha} \log \left( \mathbb{E}_{h \sim P} \left[ \exp \left( \frac{K(h)^2}{2m^{1-2\alpha}} \right) \right] \right) \right).$$
**Standard Neural Networks**

**Classification setting:**

- \( x \in \mathbb{R}^{d_0} \)
- \( y \in \{-1, 1\} \)

**Architecture:**

- \( L \) fully connected layers
- \( d_k \) denotes the number of neurons of the \( k^{th} \) layer
- \( \sigma : \mathbb{R} \to \mathbb{R} \) is the *activation function*

**Parameters:**

- \( W_k \in \mathbb{R}^{d_k \times d_{k-1}} \) denotes the weight matrices, \( D = \sum_{k=1}^{L} d_{k-1} d_k \).
- \( \theta = \text{vec} \left( \{ W_k \}_{k=1}^{L} \right) \in \mathbb{R}^{D} \)

**Prediction**

\[
\hat{f}_\theta(x) = \sigma \left( w_L \sigma \left( w_{L-1} \sigma \left( \ldots \sigma \left( w_1 x \right) \right) \right) \right). 
\]
PAC-Bayesian bounds for Stochastic NN

**Langford and Caruana (2001)**
- Shallow networks ($L = 2$)
- Sigmoid activation functions

**Dziugaite and Roy (2017)**
- Deep networks ($L > 2$)
- ReLU activation functions

**Idea:** Bound the expected loss of the network under a Gaussian perturbation of the weights

Empirical loss: $\mathbb{E}_{\theta' \sim \mathcal{N}(\theta, \Sigma)} R_{in}(f_{\theta'}) \rightarrow$ estimated by sampling

Complexity term: $KL(\mathcal{N}(\theta, \Sigma)\|\mathcal{N}(\theta_0, \Sigma_0)) \rightarrow$ closed form
**Binary Activated Neural Networks**

- \( \mathbf{x} \in \mathbb{R}^{d_0} \)
- \( \mathbf{y} \in \{-1, 1\} \)

**Architecture:**

- \( L \) fully connected layers
- \( d_k \) denotes the number of neurons of the \( k^{th} \) layer
- \( \text{sgn}(a) = 1 \) if \( a > 0 \) and \( \text{sgn}(a) = -1 \) otherwise

**Parameters:**

- \( \mathbf{W}_k \in \mathbb{R}^{d_k \times d_{k-1}} \) denotes the weight matrices.
- \( \theta = \text{vec}(\{\mathbf{W}_k\}_{k=1}^L) \in \mathbb{R}^D \)

**Prediction**

\[
f_\theta(\mathbf{x}) = \text{sgn}(\mathbf{w}_L \text{sgn}(\mathbf{W}_{L-1} \text{sgn}(\ldots \text{sgn}(\mathbf{W}_1 \mathbf{x}))))\]

---

**Diagram:**

A diagram illustrating a binary activated neural network with \( L \) layers, where each layer uses the \( \text{sgn} \) function to activate neurons.
One Layer (linear predictor)

Germain et al. (2009a)

\[ f_w(x) \overset{\text{def}}{=} \text{sgn}(w \cdot x), \text{ with } w \in \mathbb{R}^{d_0}. \]
One Layer (linear predictor)

Germain et al. (2009a)

\[ f_w(x) \overset{\text{def}}{=} \text{sgn}(w \cdot x), \text{ with } w \in \mathbb{R}^d. \]

PAC-Bayes analysis:

- Space of all linear classifiers \( \mathcal{F}_d \overset{\text{def}}{=} \{ f_v \mid v \in \mathbb{R}^d \} \)
- Gaussian posterior \( Q_w \overset{\text{def}}{=} \mathcal{N}(w, I_d) \) over \( \mathcal{F}_d \)
- Gaussian prior \( P_{w_0} \overset{\text{def}}{=} \mathcal{N}(w_0, I_d) \) over \( \mathcal{F}_d \)
- Predictor \( F_w(x) \overset{\text{def}}{=} E_{v \sim Q_w} f_v(x) = \text{erf} \left( \frac{w \cdot x}{\sqrt{d} \|x\|} \right) \)

Bound minimisation — under the linear loss \( \ell(y, y') := \frac{1}{2} (1 - yy') \)

\[ CmR_{\text{in}}(F_w) + KL(Q_w \parallel P_{w_0}) = C \frac{1}{2} \sum_{i=1}^{m} \text{erf} \left( -y_i \frac{w \cdot x_i}{\sqrt{d} \|x_i\|} \right) + \frac{1}{2} \|w - w_0\|^2. \]
Two Layers (shallow network)
Two Layers (shallow network)

Posterior $Q_\theta = \mathcal{N}(\theta, I_D)$, over the family of all networks $\mathcal{F}_D = \{f_\tilde{\theta} | \tilde{\theta} \in \mathbb{R}^D\}$, where

$$f_\theta(x) = \text{sgn}(w_2 \cdot \text{sgn}(W_1 x)).$$

$$F_\theta(x) = \mathbb{E}_{\tilde{\theta} \sim Q_\theta} f_{\tilde{\theta}}(x)$$

$$= \int_{\mathbb{R}^{d_1} \times d_0} Q_1(V_1) \int_{\mathbb{R}^{d_1}} Q_2(v_2) \text{sgn}(v_2 \cdot \text{sgn}(V_1 x)) dv_2 dV_1$$

$$= \int_{\mathbb{R}^{d_1} \times d_0} Q_1(V_1) \text{erf} \left( \frac{w_2 \cdot \text{sgn}(V_1 x)}{\sqrt{2} \| \text{sgn}(V_1 x) \|} \right) dV_1$$

$$= \sum_{s \in \{-1, 1\}^{d_1}} \text{erf} \left( \frac{w_2 \cdot s}{\sqrt{2d_1}} \right) \int_{\mathbb{R}^{d_1} \times d_0} \mathbb{1}[s = \text{sgn}(V_1 x)] Q_1(V_1) dV_1$$

$$= \sum_{s \in \{-1, 1\}^{d_1}} \text{erf} \left( \frac{w_2 \cdot s}{\sqrt{2d_1}} \right) \prod_{i=1}^{d_1} \left[ \frac{1}{2} + \frac{S_i}{2} \text{erf} \left( \frac{w_i^1 \cdot x}{\sqrt{2} \| x \|} \right) \right].$$
Stochastic Approximation

\[ F_\theta(x) = \sum_{s \in \{-1, 1\}^{d_1}} F_{w_2}(s) \Pr(s|x, W_1) \]

Monte Carlo sampling

We generate \( T \) random binary vectors \( \{s^t\}_{t=1}^T \) according to \( \Pr(s|x, W_1) \)

Prediction.

\[ F_\theta(x) \approx \frac{1}{T} \sum_{t=1}^T F_{w_2}(s^t). \]

Derivatives.

\[ \frac{\partial}{\partial w_{1k}} F_\theta(x) \approx \frac{x}{2^{3/2} \|x\|} \text{erf}' \left( \frac{w_{1k}^T \cdot x}{\sqrt{2} \|x\|} \right) \frac{1}{T} \sum_{t=1}^T \frac{s^t_k}{\Pr(s^t_k|x, w_{1k})} F_{w_2}(s^t). \]
More Layers (deep)

\[ F_1^{(j)}(x) = \text{erf} \left( \frac{w^j_1 \cdot x}{\sqrt{2\|x\|}} \right), \quad F_{k+1}^{(j)}(x) = \sum_{s \in \{-1, 1\}^{d_k}} \text{erf} \left( \frac{w^j_{k+1} \cdot s}{\sqrt{2d_k}} \right) \prod_{i=1}^{d_k} \left( \frac{1}{2} + \frac{1}{2} s_i \times F_i^{(j)}(x) \right) \]
Generalisation bound

Let $G_\theta$ denote the predictor with posterior mean as parameters. With probability at least $1 - \delta$, for any $\theta \in \mathbb{R}^D$

\[
R_{\text{out}}(G_\theta) \leq \inf_{C > 0} \left\{ \frac{1}{1 - e^{-C}} \left( 1 - \exp \left( -CR_{\text{in}}(G_\theta) - \frac{\text{KL}(\theta, \theta_0) + \log \frac{2\sqrt{m}}{\delta}}{m} \right) \right) \right\}. 
\]
### Numerical results

<table>
<thead>
<tr>
<th>Model name</th>
<th>Cost function</th>
<th>Train split</th>
<th>Valid split</th>
<th>Model selection</th>
<th>Prior</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLP–tanh</td>
<td>linear loss, L2 regularized</td>
<td>80%</td>
<td>20%</td>
<td>valid linear loss</td>
<td>-</td>
</tr>
<tr>
<td>PBGNet$\ell$</td>
<td>linear loss, L2 regularized</td>
<td>80%</td>
<td>20%</td>
<td>valid linear loss</td>
<td>random init</td>
</tr>
<tr>
<td>PBGNet</td>
<td>PAC-Bayes bound</td>
<td>100%</td>
<td>-</td>
<td>PAC-Bayes bound</td>
<td>random init</td>
</tr>
<tr>
<td>PBGNet$\text{pre}$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>– pretrain</td>
<td>linear loss (20 epochs)</td>
<td>50%</td>
<td>-</td>
<td>-</td>
<td>random init</td>
</tr>
<tr>
<td>– final</td>
<td>PAC-Bayes bound</td>
<td>50%</td>
<td>-</td>
<td>PAC-Bayes bound</td>
<td>pretrain</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dataset</th>
<th>MLP–tanh</th>
<th>PBGNet$\ell$</th>
<th>PBGNet</th>
<th>PBGNet$\text{pre}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$E_S$</td>
<td>$E_T$</td>
<td>$E_S$</td>
<td>$E_T$</td>
</tr>
<tr>
<td>ads</td>
<td>0.021</td>
<td>0.037</td>
<td>0.018</td>
<td><strong>0.032</strong></td>
</tr>
<tr>
<td>adult</td>
<td>0.128</td>
<td>0.149</td>
<td>0.136</td>
<td><strong>0.148</strong></td>
</tr>
<tr>
<td>mnist17</td>
<td>0.003</td>
<td><strong>0.004</strong></td>
<td>0.008</td>
<td>0.005</td>
</tr>
<tr>
<td>mnist49</td>
<td>0.002</td>
<td><strong>0.013</strong></td>
<td>0.003</td>
<td>0.018</td>
</tr>
<tr>
<td>mnist56</td>
<td>0.002</td>
<td>0.009</td>
<td>0.002</td>
<td>0.009</td>
</tr>
<tr>
<td>mnistLH</td>
<td>0.004</td>
<td><strong>0.017</strong></td>
<td>0.005</td>
<td>0.019</td>
</tr>
</tbody>
</table>
Thanks!

What this talk could have been about...

- Tighter PAC-Bayes bounds (Mhammedi et al., 2019)
- PAC-Bayes for conditional value at risk (Mhammedi et al., 2020)
- PAC-Bayes-driven deep neural networks (Biggs and Guedj, 2020)
- PAC-Bayes and robust learning (Guedj and Pujol, 2019)
- PAC-Bayesian online clustering (Li et al., 2018)
- PAC-Bayesian bipartite ranking (Guedj and Robbiano, 2018)
- Online $k$-means clustering (Cohen-Addad et al., 2019)
- Sequential learning of principal curves (Guedj and Li, 2018)
- Stability and generalisation (Celisse and Guedj, 2016)
- Contrastive unsupervised learning (Nozawa et al., 2020)
- Image denoising (Guedj and Rengot, 2020)
- Matrix factorisation (Alquier and Guedj, 2017; Chrétien and Guedj, 2020)
- Preventing model overfitting (Zhang et al., 2019)
- Decentralised learning with aggregation (Klein et al., 2019)
- Ensemble learning (nonlinear aggregation) in Python (Guedj and Srinivasa Desikan, 2018, 2020)
- Identifying subcommunities in social networks (Vendeville et al., 2020b,a)
- Prediction with multi-task Gaussian processes (Leroy et al., 2020)
- + a few others in the pipe, hopefully soon on arXiv!

This talk:
https://bguedj.github.io/talks/2020-10-20-seminar-modal
References I


References II


References III


References IV


