Tighter risk certificates for (probabilistic) neural networks

Omar Rivasplata
o.rivasplata@cs.ucl.ac.uk

01 July 2020
• María Pérez-Ortiz (UCL)
• Yours truly (UCL / DeepMind)
• Csaba Szepesvári (DeepMind)
• John Shawe-Taylor (UCL)
Overview of this talk

- Motivation
  - Classic NNs: weights
    - Probabilistic NNs: random weights
  - Highlights of experiments
- Conclusions
What motivated this project
Variational Bayes: \( \min_\theta KL(q_\theta(w)||p(w|D)) \)

Objective: \( f(\theta) = \mathbb{E}_{q_\theta(w)}[\log(1/p(D|w))] + KL(q_\theta(w)||p(w)) \) (ELBO)

Algorithm: ‘Bayes by Backprop’
A Strongly Quasiconvex PAC-Bayesian Bound

Niklas Thiemann  
Department of Computer Science, University of Copenhagen  
NIKLASTHIEMANN@GMAIL.COM

Christian Igel  
Department of Computer Science, University of Copenhagen  
IGEL@DI.KU.DK

Olivier Wintenberger  
LSTA, Sorbonne Universités, UPMC Université Paris 06  
OLIVIER.WINTENBERGER@UPMC.FR

Yevgeny Seldin  
Department of Computer Science, University of Copenhagen  
SELDIN@DI.KU.DK

• PAC-Bayes-lambda:

\[
\mathbb{E}_{q_\theta(w)}[L(w)] \leq \frac{\mathbb{E}_{q_\theta(w)}[\hat{L}_n(w, D)]}{1 - \lambda/2} + \frac{KL(q_\theta(w)||p(w)) + C_n}{n\lambda(1 - \lambda/2)} \quad \lambda \in (0, 2)
\]

• Algorithm: \( f(\theta, \lambda) = \text{RHS}, \) alternated optimization over \( \theta \) and \( \lambda \)
Optimized a classic PAC-Bayes bound

Experiments on ‘binary MNIST’ ([0-4] vs. [5-9])

Demonstrated non-vacuous risk bound values
Classic Neural Nets
Use the available data to:

1. learn a weight vector $\hat{w}$
2. certify $\hat{w}$’s performance

- split the data, part for (1) and part for (2)?
- the whole of the data for (1) and (2) simultaneously?
  ▶ self-certified learning!
Learning framework

\[ \text{ALG} : \mathcal{Z}^n \rightarrow \mathcal{W} \]

\[ \mathcal{W} \rightarrow \mathcal{H} \]

- \( \mathcal{Z} = \mathcal{X} \times \mathcal{Y} \)
  - \( \mathcal{X} \) = set of inputs
  - \( \mathcal{Y} \) = set of labels

- \( \mathcal{W} \subset \mathbb{R}^p \)
  - weight space
  - \( \hat{w} = \text{ALG}(\text{data}) \)

- \( \mathcal{H} \) function class
  - predictors
  - \( h_{\hat{w}} : \mathcal{X} \rightarrow \mathcal{Y} \)

data set: \( D = (Z_1, \ldots, Z_n) \in \mathcal{Z}^n \) (e.g. training set)

a finite sequence of input-label examples \( Z_i = (X_i, Y_i) \).
Empirical risk: $$\hat{L}_n(w) = \hat{L}_n(w, D) = \frac{1}{n} \sum_{i=1}^{n} \ell(w, Z_i)$$ (in-sample error)

Tied to the choice of a loss function $$\ell(w, z)$$

- the square loss (regression)
- the 0-1 loss (classification)
- the cross-entropy loss (NN classification)
  > surrogate loss, nice properties
Training set error: \[ \hat{L}_{\text{trn}}(w) = \frac{1}{n_{\text{trn}}} \sum_{Z_i \in D_{\text{trn}}} \ell(w, Z_i) \]

ERM: \[ \hat{w} \in \arg\min_w \hat{L}_{\text{trn}}(w) \]

Penalized ERM: \[ \hat{w} \in \arg\min_w \hat{L}_{\text{trn}}(w) + \text{Reg}(w) \]
If learned weight $\hat{w}$ does well on the train set examples...

...will it still do well on unseen examples?
data set: $D = (Z_1, \ldots, Z_n) \in \mathcal{Z}^n$

a finite sequence of input-label examples $Z_i = (X_i, Y_i)$.

Assumptions:

- A data-generating distribution $P \in M_1(\mathcal{Z})$.
- $P$ is unknown, only the training set is given.
- The input-label examples are i.i.d. $\sim P$.

Population risk:

$$L(w) = \mathbb{E}[\ell(w, Z)] = \int_{\mathcal{Z}} \ell(w, z) dP(z)$$

(out-of-sample)
Certifying performance: test set error

Test set error: $\hat{L}_{tst}(\hat{w}) = \frac{1}{n_{tst}} \sum_{Z_i \in D_{tst}} \ell(\hat{w}, Z_i)$

- $\hat{w}$ obtained from the training set
- test set not used for training
- $\hat{L}_{tst}(\hat{w})$ serves as estimate of $L(\hat{w})$
- Note: $L(\hat{w})$ remains unknown!
Risk upper bound: For any given \(\delta \in (0, 1)\), with probability at least 1 \(-\delta\) over random datasets of size \(n\), simultaneous for all \(w\):

\[
L(w) \leq \hat{L}_n(w) + \epsilon(n, \delta)
\]

For \(\hat{w} = \text{ALG(train_set)}\) this gives:

\[
L(\hat{w}) \leq \hat{L}_{tst}(\hat{w}) + \epsilon(n_{tst}, \delta)
\]

Recommendable practice:

- report confidence bound together with your test set error estimate
Risk upper bound: For any given $\delta \in (0, 1)$, with probability at least $1 - \delta$ over random datasets of size $n$, simultaneous for all $w$:

$$L(w) \leq \hat{L}_n(w) + \epsilon(n, \delta)$$

Alternative practice: Find $\hat{w}$ by minimizing the risk bound

- A form of regularized ERM
- the learned $\hat{w}$ comes with its own risk certificate
- best if the risk bound is non-vacuous, ideally tight!
- may avoid the need of data-splitting
- may lead to self-certified learning!
Randomized weights

Based on data $D$, learn a distribution over weights:

$$Q_D \in M_1(\mathcal{W}), \quad Q_D = \text{ALG(train\_set)}.$$ 

Predictions:
- draw $w \sim Q_D$ and predict with the chosen $w$.
- each prediction with a fresh random draw.

The risk measures $L(w)$ and $\hat{L}_n(w)$ are extended to $Q$ by averaging:

$$Q[L] \equiv \int_{\mathcal{W}} L(w) \, dQ(w) = \mathbb{E}_{w \sim Q}[L(w)]$$
$$Q[\hat{L}_n] \equiv \int_{\mathcal{W}} \hat{L}_n(w) \, dQ(w) = \mathbb{E}_{w \sim Q}[\hat{L}_n(w)]$$
Fix a distribution $Q_0$.

For any sample size $n$, for any confidence parameter $\delta \in (0, 1)$, with probability at least $1 - \delta$ over random samples (of size $n$) simultaneously for all distributions $Q$.

$$Q[L] \leq Q[\hat{L}_n] + \sqrt{\frac{KL(Q\|Q_0) + \log\left(\frac{2\sqrt{n/\delta}}{\delta}\right)}{2n}}$$  \hfill (PB-classic)

$$\text{kl}(Q[\hat{L}_n]\|Q[L]) \leq \frac{KL(Q\|Q_0) + \log\left(\frac{2\sqrt{n}}{\delta}\right)}{n}$$  \hfill (PB-kl)
Fix a distribution $Q_0$. For any size $n$, for any confidence $\delta \in (0, 1)$, with probability at least $1 - \delta$ over random samples (of size $n$)

**PB-quad:** simultaneously for all distributions $Q$

$$Q[L] \leq \left( \sqrt{Q[\hat{L}_n]} + \frac{KL(Q\|Q^0) + \log\left(\frac{2\sqrt{n}}{\delta}\right)}{2n} \right)^2 + \sqrt{KL(Q\|Q^0) + \log\left(\frac{2\sqrt{n}}{\delta}\right)^2}$$

**PB-lambda:** simultaneously for all distributions $Q$ and $\lambda \in (0, 2)$

$$Q[L] \leq \frac{Q[\hat{L}_n]}{1 - \lambda/2} + \frac{KL(Q\|Q^0) + \log\left(\frac{2\sqrt{n}}{\delta}\right)}{n\lambda(1 - \lambda/2)}$$
Donsker & Varadhan (1975), Csiszár (1975)

\[ KL(Q\|Q_0) = \sup_{f: \mathcal{W} \rightarrow \mathbb{R}} \left\{ Q[f] - \log Q_0[e^f] \right\} \]

- Let \( f : \mathbb{Z}^n \times \mathcal{W} \rightarrow \mathbb{R} \). For a given \( Q_0 \):
  \[ Q[f(D, w)] \leq KL(Q\|Q_0) + \log Q_0[e^{f(D,w)}]. \]

- Apply Markov’s inequality to \( Q_0[e^{f(D,w)}] \).

- w.p. \( \geq 1 - \delta \) over the random draw of \( D \sim P^n \), simultaneously for all distributions \( Q \):
  \[ Q[f(D, w)] \leq KL(Q\|Q_0) + \log P^n[Q_0[e^{f(D,w)}]] + \log(1/\delta). \]

- Use with suitable \( f \), upper-bound the exponential moment \( P^n[Q_0[e^{f(D,w)}]]. \)
Using a PAC-Bayes bound

- Use your favourite $\text{ALG}$ to find $Q_D = \text{ALG}(\text{train\_set})$, and plug $Q_D$ into the PAC-Bayes bound to certify its risk:

$$Q_D[L] \leq Q_D[\hat{L}_n] + \sqrt{\frac{KL(Q_D\|Q_0) + \log\left(\frac{2\sqrt{n}}{\delta}\right)}{2n}}$$

- Use the PAC-Bayes bound itself as a training objective:

$$Q_D \in \arg\min_Q Q[\hat{L}_n] + \sqrt{\frac{KL(Q\|Q_0) + \log\left(\frac{2\sqrt{n}}{\delta}\right)}{2n}}$$

Note: both uses illustrated here with PB-classic, but the same can be done with PB-quad or PB-lambda (or any other)
Training objectives

\[ f_{\text{classic}}(Q) = Q[\hat{L}_n^{\text{ce}}] + \sqrt{\frac{\text{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{2n}} \]

\[ f_{\text{quad}}(Q) = \left( \sqrt{Q[\hat{L}_n^{\text{ce}}] + \frac{\text{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{2n}} + \sqrt{\frac{\text{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{2n}} \right)^2 \]

\[ f_{\lambda}(Q, \lambda) = \frac{Q[\hat{L}_n^{\text{ce}}]}{1 - \lambda/2} + \frac{\text{KL}(Q\|Q^0) + \log(\frac{2\sqrt{n}}{\delta})}{n\lambda(1 - \lambda/2)} \]
Experiments
Algorithm 1 PAC-Bayes with Backprop (PBB)

Input:

\begin{align*}
\mu_0 & \quad \triangleright \text{Prior center parameters (random init.)} \\
\rho_0 & \quad \triangleright \text{Prior scale hyper-parameter} \\
Z_{1:n} & \quad \triangleright \text{Training examples (inputs + labels)} \\
\delta \in (0,1) & \quad \triangleright \text{Confidence parameter} \\
\alpha \in (0,1), T & \quad \triangleright \text{Learning rate; # of iterations}
\end{align*}

Output: Optimal $\mu, \rho$  
\triangleright Centers, scales

1: \textbf{procedure} PBB\_QUAD\_GAUSS
2: \quad $\mu \leftarrow \mu_0$  \triangleright \text{Set posterior centers to init. of prior}
3: \quad $\rho \leftarrow \rho_0$  \triangleright \text{Set posterior scale to $\rho_0$ hyperparam.}
4: \quad \textbf{for} $t \leftarrow 1 : T$ \textbf{do}  \triangleright \text{Run SGD for T iterations.}
5: \quad \text{Sample } V \sim \mathcal{N}(0, I)
6: \quad W = \mu + \log(1 + \exp(\rho)) \odot V
7: \quad f(\mu, \rho) = f_{\text{quad}}(Z_{1:n}, W, \mu, \rho, \mu_0, \rho_0, \delta)
8: \quad \text{SGD gradient step using} \begin{bmatrix} \nabla_{\mu} f \\ \nabla_{\rho} f \end{bmatrix}
9: \quad \textbf{return} \mu, \rho
Prior mean at the random initialization

- **(PAC-Bayes) prior** $Q_0 = \text{Gauss}(w_0, \Sigma_0)$
  - $\Sigma_0 = \lambda_0 I$ \hspace{0.5cm} ($\lambda_0$ is hyperparameter)
  - $w_0$ = randomly initialized weights

- **(PAC-Bayes) posterior** $Q_D = \text{Gauss}(w, \Sigma)$
  - $w, \Sigma$ learned by PAC-Bayes with Backprop

Experiments (ours) on MNIST

\begin{align*}
  f_{\text{quad}} & \quad f_{\text{classic}} \hspace{0.5cm} \text{(cf. D & R (2017))} \\
  \text{Test acc.} &= 86.36 \quad \text{Test acc.} = 84.22 \\
  \text{Test error} &= 0.1364 \quad \text{Test error} = 0.1578 \\
  \text{RUB value} &= 0.24107 \quad \text{RUB value} = 0.24375
\end{align*}
Prior mean learned from data

Motivation
Classic weights
Random weights
Experiments
Conclusions

• (PAC-Bayes) prior $Q_0 = \text{Gauss}(w_0, \Sigma_0)$
  $\Sigma_0 = \lambda_0 I \quad (\lambda_0 \text{ is hyperparameter})$
  $w_0 = \text{ERM on a split of the data}$

• (PAC-Bayes) posterior $Q_D = \text{Gauss}(w, \Sigma)$
  $w, \Sigma \text{ learned by PAC-Bayes with Backprop}$

Experiments (ours) on MNIST

$f_{\text{quad}}$
- Test acc. = 97.89
- Test error = 0.0211
- RUB value = 0.04588

$f_{\text{classic}}$ (cf. D & R (2018))
- Test acc. = 97.21
- Test error = 0.0279
- RUB value = 0.06029
Closing remarks
Bayesian Learning

**Posterior** $Q_D$, density $q_D(w)$  
**Prior** $Q_0$, density $q_0(w)$

$$q_D(w) = \mathcal{L}(D|w) \cdot q_0(w) / C$$

- Bayes rule update on prior to form posterior
  - likelihood factor $\mathcal{L}(D|w)$

- Principled approach, e.g. MAP learning

- Derive learning algorithms
  - balance ‘fit to data’ and ‘fit to prior’
A bit more general:
“temperature” $\lambda > 0$

$$q_D(w) = \mathcal{L}(D|w)^\lambda q_0(w) / C$$

Even more general:
data-dependent factor $\mathcal{F}$

$$q_D(w) = \mathcal{F}(D, w) q_0(w)$$

  A general framework for updating belief distributions
PAC-Bayes

Motivation
- Classic weights
- Random weights
- Experiments
- Conclusions

\[ q_D(w) \quad \text{[no update factor]} \quad q_0(w) \]

- more general than generalized Bayes
- increased flexibility in choice of distributions
- balance \( q_D[\hat{L}_n] \) and \( KL(q_D||q_0) \)
  - ‘fit to data’ versus ‘fit to prior’
Future

- choice of distributions
- understand properties
- scaling to larger problems?
- architecture vs. PAC-Bayes bounds?
- problem-specific PAC-Bayes bounds?
Thank you!
Wait...
some PAC-Bayes history

  A PAC analysis of a Bayesian estimator

- D.A. McAllester (1998)
  Some PAC-Bayesian Theorems

- D.A. McAllester (1999)
  PAC-Bayesian Model Averaging

- J. Langford & M. Seeger (2001)
  Bounds for Averaging Classifiers

  (Not) Bounding the True Error

- M. Seeger (2002)
  PAC-Bayesian generalization bounds for gaussian processes
some more PAC-Bayes history

  PAC-Bayes & Margins

- D.A. McAllester (2003)
  Simplified PAC-Bayesian Margin Bounds

  A note on the PAC Bayesian theorem

  A better variance control for PAC-Bayesian classification

  PAC-Bayesian supervised classification:
  The thermodynamics of statistical learning

  PAC-Bayesian learning of linear classifiers
some recent PAC-Bayes

- J. Keshet, D.A. McAllester, T. Hazan (2011) PAC-Bayesian approach for minimization of phoneme error rate
- A. Noy & K. Crammer (2014) Robust forward algorithms via PAC-Bayes and Laplace distributions
- N. Thiemann, C. Igel, O. Wintenberger, Y. Seldin (2017) A Strongly Quasiconvex PAC-Bayesian Bound
more recent PAC-Bayes

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

Simpler PAC-Bayesian bounds for hostile data

- S.S. Lorenzen, C. Igel, Y. Seldin (2019)
On PAC-Bayesian Bounds for Random Forests

Dichotomize and Generalize: PAC-Bayesian Binary Activated Deep Neural Networks

PAC-Bayes with Backprop (in arXiv)
Thank you again!