Uncertainty Quantification for Predictive Models

Eric Nalisnick
University of Amsterdam
\[ f(x) = \hat{y} \]

classifier

feature vector

predicted label
$f(x) = \hat{y}$
diagnosis

\[ f(x) = \hat{y} \]

health measurements
Wait a minute…How certain is the model about its prediction?
Wait a minute…How *certain* is the model about its prediction?

\[ f(x) = \hat{y} + ??? \]
Wait a minute...How certain is the model about its prediction?

\[ f(x) = \hat{y} + P(y = \hat{y} | x) \]

example: 80% confident that the patient has pneumonia
Wait a minute... How certain is the model about its prediction?

$$f(x) = \hat{y} + \mathbb{P}(y^* \in C(x)) \geq 95\%$$

set of labels
Wait a minute…How certain is the model about its prediction?

\[ f(x) = \hat{y} + P\left(y^* \in C(x)\right) \geq 95\% \]

example: 95% confident that the patient has pneumonia, tuberculosis, or asthma.
This talk is about how to know what your model doesn’t know.
I. What is Uncertainty?
Two Types of Uncertainty

⊗ **Aleatoric**
  • fundamental, related to Bayes error rate
  • possibly reduced by collecting more features

⊗ **Epistemic**
  • due to lack of experience / observations
  • always reduced by collecting more data points
Two Types of Uncertainty

Aleatoric
- fundamental, related to Bayes error rate
- possibly reduced by collecting more features

Epistemic
- due to lack of experience / observations
- always reduced by collecting more data points
Examples from Autonomous Driving

train in urban environment
Examples from Autonomous Driving

train in urban environment

high aleatoric:
avoiding a head-on collision
Two Types of Uncertainty

Aleatoric
- fundamental, related to Bayes error rate
- possibly reduced by collecting more features

Epistemic
- due to lack of experience / observations
- always reduced by collecting more data points
Two Types of Uncertainty

⊗ Aleatoric
  • fundamental, related to Bayes error rate
  • possibly reduced by collecting more features

⊗ Epistemic
  • due to lack of experience / observations
  • always reduced by collecting more data points
Examples from Autonomous Driving

train in urban environment

high aleatoric: avoiding a head-on collision
Examples from Autonomous Driving

train in urban environment

high aleatoric: avoiding a head-on collision

high epistemic: driving in the mountains
In practice, distinguishing aleatoric vs epistemic uncertainty is incredibly difficult.
In practice, distinguishing aleatoric vs epistemic uncertainty is incredibly difficult.

For the rest of this talk, I’ll ignore the distinction: uncertainty is ‘high’ when either types are ‘high.’
II. Modeling Paradigms
Assumptions

fixed, unknown distribution generates the data:

\[ y \sim \mathbb{P} \left( y \mid x \right) \]

we only see samples, i.e. the training set:

\[ \mathfrak{D} = \left\{ (x_n, y_n) \right\}_{n=1}^{N} \]

fit model to recover the ground-truth distribution:

\[ p(y \mid x) \approx \mathbb{P} \left( y \mid x \right) \]
Assumptions

fixed, unknown distribution generates the data:

\[ y \sim \mathbb{P} \left( y \mid x \right) \]

we only see samples, i.e. the training set:

\[ \mathcal{D} = \left\{ (x_n, y_n) \right\}_{n=1}^{N} \]

fit model to recover the ground-truth distribution:

\[ p(y \mid x) \approx \mathbb{P} \left( y \mid x \right) \]
Assumptions

fixed, unknown distribution generates the data:

\[ y \sim \mathbb{P} \left( y \mid x \right) \]

we only see samples, i.e. the training set:

\[ \mathcal{D} = \left\{ (x_n, y_n) \right\}_{n=1}^{N} \]

fit model to recover the ground-truth distribution:

\[ p(y \mid x) \approx \mathbb{P} \left( y \mid x \right) \]
Assumptions

fixed, unknown distribution generates the data:

\[ y \sim P(y \mid x) \]

we only see samples, i.e. the training set:

\[ \mathcal{D} = \left\{ (x_n, y_n) \right\}_{n=1}^{N} \]

fit model to recover the ground-truth distribution:

\[ p(y \mid x) \approx P(y \mid x) \]
Two Modeling Paradigms

- **Frequentism**: randomness from data’s sampling distribution

- **Bayesianism**: randomness from prior distribution over model parameters
Two Modeling Paradigms

- **Frequentism**: randomness from data’s sampling distribution

- **Bayesianism**: randomness from prior distribution over model parameters
Frequentist Learning: Maximum Likelihood

maximize the log-likelihood of parameters:

\[ \hat{\theta} = \arg\max_{\theta \in \Theta} \sum_{n=1}^{N} \log p \left( y_n \mid x_n, \theta \right) \]

(for classification, equivalent to cross-entropy loss)
Frequentist Learning: Ideal UQ

under (near) perfect learning, can quantify uncertainty simply by...

model probability reflects confidence:

\[ p \left( \hat{y} \mid x; \hat{\theta} \right) \approx \mathbb{P} \left( y = \hat{y} \mid x \right) \]
Frequentist Learning: Ideal UQ

under (near) perfect learning, can quantify uncertainty simply by...

model probability reflects confidence:
Frequentist Learning: Ideal UQ

under (near) perfect learning, can quantify uncertainty simply by…

certainty set is top ranked outputs:

- softmax output
  - classes
  - probabilities
    - .30
    - .15
    - .05
    - .50
Frequentist Learning: Ideal UQ

under (near) perfect learning, can quantify uncertainty simply by...

confidence set is top ranked outputs:

softmax output

.30 .15 .05 .50

classes
Frequentist Learning: Limitations

well-motivated with large data sets, but models are big and data is often scarce!

[Guo et al., ICML 2017]
Two Modeling Paradigms

- **Frequentism**: randomness from data’s sampling distribution

- **Bayesianism**: randomness from prior distribution over model parameters
Two Modeling Paradigms

⊗ Frequentism: randomness from data’s sampling distribution

⊗ Bayesianism: randomness from prior distribution over model parameters
Bayesian Learning: Posterior Distribution

\[ p(\theta | \mathcal{D}) = \frac{p(\theta) \prod_{n=1}^{N} p(y_n | x_n, \theta)}{p(\mathcal{D})} \]

posterior distribution
Bayesian Learning: Posterior Distribution

\[ p(\theta | \mathcal{D}) = \frac{p(\theta) \prod_{n=1}^{N} p(y_n | x_n, \theta)}{p(\mathcal{D})} \]

posterior distribution
Bayesian Learning: Posterior Distribution

\[ p(\theta | \mathcal{D}) = \frac{p(\theta) \prod_{n=1}^{N} p(y_n | x_n, \theta)}{p(\mathcal{D})} \]

posterior distribution
Bayesian Learning: Posterior Distribution

$$p(\theta | \mathcal{D}) = \frac{p(\theta) \prod_{n=1}^{N} p(y_n | x_n, \theta)}{p(\mathcal{D})}$$

posterior distribution
Bayesian Learning: Posterior Distribution

\[ p(\theta | \mathcal{D}) = \frac{p(\theta) \prod_{n=1}^{N} p(y_n | x_n, \theta)}{p(\mathcal{D})} \]

\[ \int_{\theta} p(\theta) \prod_{n=1}^{N} p(y_n | x_n, \theta) \, d\theta \]

marginal likelihood
Bayesian Learning: Ideal UQ

for new data point $\tilde{x}$

integrate out posterior distribution:

$$p \left( \tilde{y} | \tilde{x}, \mathcal{D} \right) = \int_{\theta} p \left( \tilde{y} | \tilde{x}, \theta \right) p \left( \theta | \mathcal{D} \right) \, d\theta$$

posterior predictive distribution
Bayesian Learning: Ideal UQ

under (near) perfect learning, use posterior predictive distribution for ground-truth probabilities

\[
p(\tilde{y} | \tilde{x}, \mathcal{D}) \approx P(y = \tilde{y} | x)
\]
Bayesian Learning: Ideal UQ

under (near) perfect learning, use posterior predictive distribution for ground-truth probabilities
Bayesian Learning: Limitations

integrating over the parameters is difficult to even approximate for neural networks.

- marginal likelihood
- posterior predictive distribution
Bayesianism vs Frequentism: Summary

❌ Frequentism
  • data-driven, easy computation
  • misled by sampling noise if dataset is not large

❌ Bayesianism
  • prior distribution ‘jump starts’ learning
  • computation is very, very costly
Bayesianism vs Frequentism: Summary

☑ Frequentism
  • data-driven, easy computation
  • misled by sampling noise if dataset is not large

☑ Bayesianism
  • prior distribution ‘jump starts’ learning
  • computation is very, very costly
Bayesianism vs Frequentism: Summary

✗ Frequentism
• data-driven, easy computation
• misled by sampling noise if dataset is not large

✗ Bayesianism
• prior distribution ‘jump starts’ learning
• computation is very, very costly
III. Uncertainty Quantification in Practice
Practical Methods for UQ

Frequentism
• bootstrap aggregation (bagging)
• conformal prediction

Bayesianism
• sample-then-optimize ensembling
• variational inference w/ Laplace approximation
Practical Methods for UQ

- **Frequentism**
  - bootstrap aggregation (bagging)
  - conformal prediction

- **Bayesianism**
  - sample-then-optimize ensembling
  - variational inference w/ Laplace approximation
Bootstrap Aggregation

recall that frequentism assumes randomness comes from the data sampling process.

if we could see additional data sets, we could know more about the sampling noise.

\[ \times \quad \{ \mathcal{D}_k \}_{k=1}^K \sim \mathbb{P} \left( y \mid x \right) \]
Bootstrap Aggregation

Bootstrapping synthesizes additional data sets by resampling from the training set.

$$\{\mathcal{D}_k\}_{k=1}^K \sim \frac{1}{N} \sum_{n=1}^{N} \delta \left[(x_n, y_n)\right]$$

sampling with replacement
Bootstrap Aggregation

Original dataset → resample → \( \mathcal{D}_1 \) → max likelihood → \( \hat{\theta}_1 \) → resample → \( \mathcal{D}_K \) → max likelihood → \( \hat{\theta}_K \) → \( \frac{1}{K} \sum_{k=1}^{K} p(y \mid x, \hat{\theta}_k) \) → bagging ensemble

\[ p(y \mid x, \hat{\theta}) = \frac{1}{K} \sum_{k=1}^{K} p(y \mid x, \hat{\theta}_k) \]
Bootstrap Aggregation

\[ \frac{1}{K} \sum_{k=1}^{K} p(y \mid x, \hat{\theta}_k) \]

Original dataset \( \mathcal{D}_1 \) \( \cdots \) \( \mathcal{D}_K \)

Resample

Max likelihood

Ensemble
Bootstrap Aggregation

Original dataset $\Sigma_1$ → resample $\Sigma_1$ → max likelihood $\hat{\theta}_1$

$\vdots$

$\Sigma_K$ → resample $\Sigma_K$ → max likelihood $\hat{\theta}_K$

$\frac{1}{K} \sum_{k=1}^{K} p(y \mid x, \hat{\theta}_k)$ → bagging ensemble
Bootstrap Aggregation

Original dataset → \( \mathcal{D}_1 \) → max likelihood → \( \hat{\theta}_1 \) → resample → \( \mathcal{D}_2 \) → max likelihood → \( \hat{\theta}_2 \) → resample → \( \mathcal{D}_K \) → max likelihood → \( \hat{\theta}_K \) → \( \frac{1}{K} \sum_{k=1}^{K} p(y \mid x, \hat{\theta}_k) \) → bagging ensemble
Further Reading

An Introduction to the Bootstrap

Bradley Efron
Robert J. Tibshirani

CHAPMAN & HALL/CRC
Practical Methods for UQ

- Frequentism
  - bootstrap aggregation (bagging)
  - conformal prediction

- Bayesianism
  - sample-then-optimize ensembling
  - variational inference w/ Laplace approximation
Practical Methods for UQ

☐ Frequentism
  • bootstrap aggregation (bagging)
  • conformal prediction

☐ Bayesianism
  • sample-then-optimize ensembling
  • variational inference w/ Laplace approximation
Sample-then-Optimize Ensemble

recall that Bayesianism assumes randomness comes from the prior.

we perform a bagging-like procedure, but using samples from the prior to initialize training.

\[
\{ \bar{\theta}_k \}_{k=1}^{K} \sim p(\theta)
\]
Sample-then-Optimize Ensemble

$$\bar{\theta}_k \xrightarrow{\text{max likelihood}} \hat{\theta}_k \xrightarrow{\text{max likelihood}} \frac{1}{K} \sum_{k=1}^{K} p \left( y \mid x, \hat{\theta}_k \right)$$

$p(\theta)$

Bayesian prior

Sample:

$$\bar{\theta}_k$$

$$\hat{\theta}_k$$

$$\sum_{k=1}^{K}$$

$$\frac{1}{K}$$

$$\approx$$

approximate predictive distribution
Sample-then-Optimize Ensemble

$p(\theta)$

Bayesian prior

Sample

$\bar{\theta}_1$

max likelihood

$\hat{\theta}_1$

Sample

$\bar{\theta}_K$

max likelihood

$\hat{\theta}_K$

$\frac{1}{K} \sum_{k=1}^{K} p(y | x, \hat{\theta}_k)$

approximate predictive distribution
Sample-then-Optimize Ensemble

\[ p(\theta) \rightarrow \bar{\theta}_1 \xrightarrow{\text{max likelihood}} \hat{\theta}_1 \]

\[ \vdots \]

\[ \bar{\theta}_K \xrightarrow{\text{max likelihood}} \hat{\theta}_K \]

\[
\frac{1}{K} \sum_{k=1}^{K} p(y | x, \hat{\theta}_k)
\]

Bayesian prior
Sample-then-Optimize Ensemble

\[ p(\theta) \quad \rightarrow \quad \bar{\theta}_1 \quad \xrightarrow{\text{max likelihood}} \quad \hat{\theta}_1 \quad \xrightarrow{\text{max likelihood}} \quad \bar{\theta}_K \quad \xrightarrow{\text{max likelihood}} \quad \hat{\theta}_K \]

\[ \frac{1}{K} \sum_{k=1}^{K} p \left( y | x, \hat{\theta}_k \right) \]

Bayesian prior
[Matthews et al., 2017] show the procedure (very nearly) recovers the posterior in linear models.
Sample-then-Optimize Ensemble

<table>
<thead>
<tr>
<th>Metric</th>
<th>HMC (reference)</th>
<th>SGD</th>
<th>Deep Ens</th>
<th>MFVI</th>
<th>SGLD</th>
<th>SGHMC</th>
<th>SGHMC CLR</th>
<th>SGHMC CLR-PREC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CIFAR-10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Accuracy</td>
<td>89.64 ±0.25</td>
<td>83.44 ±1.14</td>
<td>88.49 ±0.10</td>
<td>86.45 ±0.27</td>
<td>89.32 ±0.23</td>
<td>89.38 ±0.32</td>
<td>89.63 ±0.37</td>
<td>87.46 ±0.21</td>
</tr>
<tr>
<td>Agreement</td>
<td>94.01 ±0.25</td>
<td>85.48 ±1.00</td>
<td>91.52 ±0.06</td>
<td>88.75 ±0.24</td>
<td>91.54 ±0.15</td>
<td>91.98 ±0.35</td>
<td>92.67 ±0.52</td>
<td>90.96 ±0.24</td>
</tr>
</tbody>
</table>

surprisingly comparable to high-fidelity Bayesian inference (performed on 512 TPUs).

[Izmailov et al., ICML 2021]
Bayesian vs Frequentist Ensembles

\[ \frac{1}{K} \sum_{k=1}^{K} p(y \mid x, \hat{\theta}_k) \]

max likelihood

\[ \hat{\theta}_1 \]

\[ \ldots \]

\[ \ldots \]

\[ \hat{\theta}_k \]

sample

sample

sample
Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles

Balaji Lakshminarayanan Alexander Pritzel Charles Blundell
DeepMind
{balajil,pritzel,cblundell}@google.com

Abstract

Deep neural networks (NNs) are powerful black box predictors that have recently achieved impressive performance on a wide spectrum of tasks. Quantifying predictive uncertainty in NNs is a challenging and yet unsolved problem. Bayesian NNs, which learn a distribution over weights, are currently the state-of-the-art for estimating predictive uncertainty; however these require significant modifications to the training procedure and are computationally expensive compared to standard (non-Bayesian) NNs. We propose an alternative to Bayesian NNs that is simple to implement, readily parallelizable, requires very little hyperparameter tuning, and yields high quality predictive uncertainty estimates. Through a series of experiments on classification and regression benchmarks, we demonstrate that our method produces well-calibrated uncertainty estimates which are as good or better than approximate Bayesian NNs. To assess robustness to dataset shift, we evaluate the predictive uncertainty on test examples from known and unknown distributions, and show that our method is able to express higher uncertainty on out-of-distribution examples. We demonstrate the scalability of our method by evaluating predictive uncertainty estimates on ImageNet.

Further Reading

What Are Bayesian Neural Network Posteriors Really Like?

Pavel Izmalkov 1  Sharad Vikram 2  Matthew D. Hoffman 2  Andrew Gordon Wilson 3

Abstract

The posterior over Bayesian neural network (BNN) parameters is extremely high-dimensional and non-convex. For computational reasons, researchers approximate this posterior using in-expensive mini-batch methods such as mean-field variational inference or stochastic gradient Markov chain Monte Carlo (SGMCMC). To investigate foundational questions in Bayesian deep learning, we instead use full-batch Hamiltonian Monte Carlo (HMC) on modern architectures. We show that (1) BNNs can achieve significant performance gains over standard training and deep ensembles; (2) a single long HMC chain can provide a comparable representation of the posterior to multiple shorter chains; (3) in contrast to recent studies, we find posterior tempering is not needed for near-optimal performance, with little evidence for a "cold posterior effect", which we show is largely an artifact of data augmentation; (4) BMA performance is robust to the choice of prior scale, and relatively similar for diagonal Gaussian, mixture of Gaussian, and logistic priors; (5) Bayesian neural networks show surprisingly poor generalization under domain shift; (6) while cheaper alternatives such as deep ensembles and SGMCMC can provide good generalization, their predictive distributions are distinct from HMC. Notably, deep ensemble predictive distributions are similar close to HMC as standard SGLD, and closer than standard variational inference.

for neural networks promises improved predictions, reliable uncertainty estimates, and principled model comparison, naturally supporting active learning, continual learning, and decision-making under uncertainty. The Bayesian deep learning community has designed multiple successful practical methods inspired by the Bayesian approach (Blundell et al., 2015; Gal & Ghahramani, 2016; Welling & Teh, 2011; Kingma et al., 2017; Maddox et al., 2019; Izmalkov et al., 2019; Danacker et al., 2020), with applications ranging from astrophysics (Crammer et al., 2021) to automatic diagnosis of Diabetic Retinopathy (Pino et al., 2019), click-through rate prediction in advertising (Liu et al., 2017) and fluid dynamics (Gorova & Zahraei, 2020).

However, inference with modern BNNs is distinctly challenging. We wish to compute a Bayesian model average corresponding to an integral over a multi-dimensional multi-modal posterior, with unusual topological properties like mode-connectivity (Gorova et al., 2018; Doshi et al., 2019), under severe computational constraints. There are therefore many unresolved questions about Bayesian deep learning practice. Variational procedures typically provide unimodal Gaussian approximations to the multimodal posterior. Practically successful methods such as deep ensembles (Lakshminarayanan et al., 2017; Fort et al., 2019) have a natural Bayesian interpretation (Wilson & Izmalkov, 2020), but only represent modes of the posterior. While Stochastic MCMC (Welling & Teh, 2011; Chen et al., 2016; Zhang et al., 2020) is computationally convenient, it could be providing heavily biased estimates of posterior expectations. Moreover, Wenzel et al. (2020) question the quality of standard Bayes posteriors, citing results when "cold posteriors", raised to a power 1/7 with
Practical Methods for UQ

Frequentism
- bootstrap aggregation (bagging)
- conformal prediction

Bayesianism
- sample-then-optimize ensembling
- variational inference w/ Laplace approximation
Practical Methods for UQ

🗑 Frequentism
• bootstrap aggregation (bagging)
• conformal prediction

🗑 Bayesianism
• sample-then-optimize ensembling
• variational inference w/ Laplace approximation
Conformal Prediction

The aim of conformal prediction is to construct uncertainty sets with guaranteed validity.

\[
P\left(y^*_{N+1} \in C(x_{N+1}) \right) \geq 1 - \alpha
\]

[Angelopoulos & Bates, 2022]
The aim of conformal prediction is to construct uncertainty sets with guaranteed validity.
Conformal Prediction

instead of using raw outputs as the confidence level, use held-out data to adapt the threshold

theoretical guarantee stems from the assumption of exchangeability:

\[ P(y_1, y_2, y_3) = P(y_{\pi(1)}, y_{\pi(2)}, y_{\pi(3)}) \]

for any permutation \( \pi \)
Conformal Prediction: Train-Time

i) for every point in the held-out set, sort the model probabilities in decreasing order.
Conformal Prediction: Train-Time

\[ p(y = 2 | x) > p(y = 1 | x) > p(y = 3 | x) \]

i) for every point in the held-out set, sort the model probabilities in decreasing order.
Conformal Prediction: Train-Time

\[ p(y = 2 \mid x) > p(y = 1 \mid x) > p(y = 3 \mid x) \]

ii) sum the probabilities (decreasing order) until the true-class is included.
Conformal Prediction: Train-Time

ii) sum the probabilities (decreasing order) until the true-class is included.
ii) sum the probabilities (decreasing order) until the true-class is included.
Conformal Prediction: Train-Time

iii) compute the \((1-\alpha)\)-quantile of the scores across the held-out set.

\[ s(x) = p(y = 2 \mid x) + p(y = 1 \mid x) \]
iv) compute the \((1-\alpha)\)-quantile of the scores.

\[ \hat{q}_{1-\alpha} \]
Conformal Prediction: Test-Time

1) rank classes by model probabilities.

\[
p(y = 1 \mid x) \quad \quad p(y = 2 \mid x) \quad \quad p(y = 3 \mid x)
\]
Conformal Prediction: Test-Time

$p(y = 3 \mid x) > p(y = 2 \mid x) > p(y = 1 \mid x)$

i) rank classes by model probabilities.
Conformal Prediction: Test-Time

\[ p(y = 3 \mid x) > p(y = 2 \mid x) > p(y = 1 \mid x) \]

\[
\text{C}(x) = \left\{ \begin{array}{c}
\text{checking if} \sum_{y \in \text{C}(x)} p(y \mid x) \geq \hat{q}_{1-\alpha} \\
\end{array} \right\}
\]

ii)
Conformal Prediction: Test-Time

\[ p(y = 3 \mid x) > p(y = 2 \mid x) > p(y = 1 \mid x) \]

\[ C(x) = \begin{cases} \text{fox} & \text{checking if} \sum_{y \in C(x)} p(y \mid x) \geq \hat{q}_{1-\alpha} \end{cases} \]
Conformal Prediction: Test-Time

\[
p(y = 3 | x) > p(y = 2 | x) > p(y = 1 | x)
\]

ii) \( C(x) = \left\{ \right\} \) checking if \( \sum_{y \in C(x)} p(y | x) \geq \hat{q}_{1-\alpha} \)
Conformal Prediction: Test-Time

\[ p(y = 3 \mid x) > p(y = 2 \mid x) > p(y = 1 \mid x) \]

\[ C(x) = \left\{ \text{fox, beaver} \right\} \]

Checking if \[ \sum_{y \in C(x)} p(y \mid x) \geq \hat{q}_{1-\alpha} \]
Conformal Prediction: Test-Time

\[ p(y = 3 \mid x) > p(y = 2 \mid x) > p(y = 1 \mid x) \]

\[ \sum_{y \in C(x)} p(y \mid x) \geq \hat{q}_{1-\alpha} \]

ii) \[ C(x) = \left\{ \text{fox, beaver} \right\} \]

checking if
Conformal Prediction: Test-Time

\[ C(x) = \begin{cases} 
  y = 3 & \text{true label is guaranteed to be in this set (1-\(\alpha\))% of the time, on average over the test set.} \\
  y = 2 
\end{cases} \]
A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification

Anastasios N. Angelopoulos and Stephen Bates
December 8, 2022

Abstract

Black-box machine learning models are now routinely used in high-risk settings, like medical diagnostics, which demand uncertainty quantification to avoid consequential model failures. Conformal prediction (a.k.a. conformal inference) is a user-friendly paradigm for creating statistically rigorous uncertainty sets/Intervals for the predictions of such models. Critically, the sets are valid in a distribution-free sense: they possess explicit, non-asymptotic guarantees even without distributional assumptions or model assumptions. One can use conformal prediction with any pre-trained model, such as a neural network, to produce sets that are guaranteed to contain the ground truth with a user-specified probability, such as 95%. It is easy-to-understand, easy-to-use, and general, applying naturally to problems arising in the fields of computer vision, natural language processing, deep reinforcement learning, and so on.

This hands-on introduction is aimed to provide the reader a working understanding of conformal prediction and related distribution-free uncertainty quantification techniques with one self-contained document. We lead the reader through practical theory for and examples of conformal prediction and describe its extensions to complex machine learning tasks involving structured outputs, distribution shift, time-series, outliers, models that abstain, and more. Throughout, there are many explanatory illustrations, examples, and code samples in Python. With each code sample comes a Jupyter notebook implementing the method on a real-data example; the notebooks can be accessed and easily run by clicking on the following icons: 🛠️.
Practical Methods for UQ

- **Frequentism**
  - bootstrap aggregation (bagging)
  - conformal prediction

- **Bayesianism**
  - sample-then-optimize ensembling
  - variational inference w/ Laplace approximation
Practical Methods for UQ

⊗ Frequentism
  • bootstrap aggregation (bagging)
  • conformal prediction

⊗ Bayesianism
  • sample-then-optimize ensembling
  • variational inference w/ Laplace approximation
Variational Inference

construct a tractable approximation to the Bayesian posterior distribution.

\[ q(\theta) \approx p(\theta | \mathcal{D}) \]
Laplace Approximation

\[-\log p(y | x, \theta) - \log p(\theta)\]
Laplace Approximation

\[- \log p(y|x, \theta) - \log p(\theta)\]
Laplace Approximation

\[-\log p(y|x, \theta) - \log p(\theta)\]
Laplace Approximation

\[ p(\theta | \mathcal{D}) \approx N\left( \hat{\theta}_{\text{MAP}}, H^{-1}(\hat{\theta}_{\text{MAP}}) \right) \]
Laplace Approximation

\[ p(\theta | \mathcal{D}) \approx N \left( \hat{\theta}_{\text{MAP}}, H^{-1}(\hat{\theta}_{\text{MAP}}) \right) \]

\[ H(\theta) = - \sum_{n=1}^{N} \frac{\partial^2 \left\{ \log p(y_n | x_n, \theta) + \log p(\theta) \right\}}{\partial \theta^2} \]
Laplace Approximation

\[ -\log p(y|x, \theta) - \log p(\theta) \]
Laplace Approximation

\[- \log p(y|x, \theta) - \log p(\theta)\]
Laplace Approximation

\[
- \log p(y|x, \theta) - \log p(\theta) = H(\theta)
\]

small curvature, large posterior variance

\[
N \left( \hat{\theta}_{\text{MAP}}, H^{-1}(\hat{\theta}_{\text{MAP}}) \right)
\]
Laplace Approximation

\[-\log p(y|x, \theta) - \log p(\theta)\]
Laplace Approximation

\[ -\log p(y | x, \theta) - \log p(\theta) \]

\[ H(\theta) \]

large curvature, small posterior variance

\[ N \left( \hat{\theta}_{\text{MAP}}, H^{-1}(\hat{\theta}_{\text{MAP}}) \right) \]
Laplace Approximation

compute predictive distribution using posterior approximation

\[ p(\tilde{y} | \tilde{x}, \mathcal{D}) \approx \int_{\theta} p(\tilde{y} | \tilde{x}, \theta) \ N \left( \hat{\theta}_{\text{MAP}}, H^{-1}(\hat{\theta}_{\text{MAP}}) \right) \ d\theta \]

might need to approximate integral with sampling
Laplace Approximation

\[- \log p(y|x, \theta) - \log p(\theta) \]
Laplace Approximation

\[ -\log p(y) - \log p(\theta) \]

Legend: $\theta_1$, $\theta_2$
Laplace Approximation

\[-\log p(y | x, \theta) - \log \pi(\theta)\]

\(\theta_1\)

\(\theta_2\)
Sample-then-Optimize Laplace Approximation

\[
p(\theta) \rightarrow \hat{\theta}_1 \rightarrow \hat{\theta}_1 \approx \frac{1}{K} \sum_{k=1}^{K} N(\hat{\theta}_k, H^{-1}(\hat{\theta}_k))
\]
Laplace Redux – Effortless Bayesian Deep Learning

Erik Daxberger\textsuperscript{a,b}\textsuperscript{*}, Agustinus Kristiadi\textsuperscript{a}, Alexander Immer\textsuperscript{a,c,d}\textsuperscript{*}, Matthias Bauer\textsuperscript{a}, Philipp Hennig\textsuperscript{a,e}\textsuperscript{*}, Runa Eschenhagen\textsuperscript{a}\textsuperscript{,f}

\textsuperscript{a}University of Cambridge
\textsuperscript{b}MPI for Intelligent Systems, Tübingen
\textsuperscript{c}University of Tübingen
\textsuperscript{d}Department of Computer Science, ETH Zurich
\textsuperscript{e}Max Planck ETH Center for Learning Systems
\textsuperscript{f}DeepMind, London

Abstract

Bayesian formulations of deep learning have been shown to have compelling theoretical properties and offer practical functional benefits, such as improved predictive uncertainty quantification and model selection. The Laplace approximation (LA) is a classic, and arguably the simplest family of approximations for the intractable posteriors of deep neural networks. Yet, despite its simplicity, the LA is not as popular as alternatives like variational Bayes or deep ensembles. This may be due to assumptions that the LA is expensive due to the involved Hessian computation, that it is difficult to implement, or that it yields inferior results. In this work we show that these are misconceptions: we (i) review the range of variants of the LA including versions with minimal cost overhead; (ii) introduce \texttt{laplace}, an easy-to-use software library for PyTorch offering user-friendly access to all major flavors of the LA; and (iii) demonstrate through extensive experiments that the LA is competitive with more popular alternatives in terms of performance, while excelling in terms of computational cost. We hope that this work will serve as a catalyst to a wider adoption of the LA in practical deep learning, including in domains where Bayesian approaches are not typically considered at the moment.

Package \texttt{laplace}

The \texttt{laplace} package facilitates the application of Laplace approximations for entire neural networks, subnetworks of neural networks, or just their last layer. The package enables posterior approximations, marginal-likelihood estimation, and various posterior predictive computations. The library documentation is available at https://aleximmer.github.io/Laplace.

There is also a corresponding paper, \textit{Laplace Redux – Effortless Bayesian Deep Learning}, which introduces the library, provides an introduction to the Laplace approximation, reviews its use in deep learning, and empirically demonstrates its versatility and competitiveness. Please consider referring to the paper when using our library.
Practical Methods for UQ

⊗ Frequentism
  • bootstrap aggregation (bagging)
  • conformal prediction

⊗ Bayesianism
  • sample-then-optimize ensembling
  • variational inference w/ Laplace approximation
Practical Methods for UQ: Summary

🟥 Frequentism
- need to do more than maximum likelihood
- extra data: synthesized or from held-out set

🟥 Bayesianism
- need to do less for the sake of computation
- construct approximations localized to areas of high posterior density.
Hybrid Methods for UQ

can mix Bayesian and frequentist procedures!

- data augmentation: sampling new data by applying bespoke transformations to original dataset.
- apply conformal prediction to posterior predictive distribution: frequentist correction to a Bayesian model
IV. Evaluating
Uncertainty Quantification
Evaluating UQ

-Calibration: can the model forecast its own performance?
-Coverage: does the model meet the given error level?
Calibration
does the model confidence reflect its empirical accuracy?

reliability diagram
Calibration

does the model confidence reflect its empirical accuracy?

reliability diagram
Calibration

does the model confidence reflect its empirical accuracy?

perfect calibration
Calibration

does the model confidence reflect its empirical accuracy?

under-confident
Calibration

does the model confidence reflect its empirical accuracy?

over-confident
Calibration: Over-Confidence

[Diagram showing calibration gap and accuracy vs. confidence for ResNet (2016) on CIFAR-100]

[Guo et al., ICML 2017]
On Calibration of Modern Neural Networks

Chuan Guo 1, Geoff Pleiss 1, Yu Sun 1, Killian Q. Weinberger 1

Abstract

Confidence calibration — the problem of predicting probability estimates representative of the true correctness likelihood — is important for classification models in many applications. We discover that modern neural networks, unlike those from a decade ago, are poorly calibrated. Through extensive experiments, we observe that depth, width, weight decay, and Batch Normalization are important factors influencing calibration. We evaluate the performance of various post-processing calibration methods on state-of-the-art architectures with image and document classification datasets. Our analysis and experiments not only offer insights into neural network learning, but also provide a simple and straightforward recipe for practical settings: on most datasets, temperature scaling — a single-parameter variant of Platt Scaling — is surprisingly effective at calibrating predictions.

Evaluating model calibration in classification

Jouzas Valnienaitis
Uppsala University; Yenner Inc.

David Widmann
Uppsala University

Carl Anderson
Uppsala University

Fredrik Lindsten
Linköping University

Jacob Roll
Yenner Inc.

Thomas B. Schön
Uppsala University

Abstract

Probabilistic classifiers output a probability distribution on target classes rather than just a class prediction. Besides providing a clear separation of prediction and decision making, the main advantage of probabilistic models is their ability to represent uncertainty about predictions. In safety-critical applications, it is pivotal for a model to possess an adequate sense of uncertainty, which for probabilistic classifiers translates into outputting probability distributions that are consistent with the empirical frequencies observed from realized outcomes. A classifier with such a property is called calibrated. In this work, we develop a general theoretical calibration evaluation framework grounded in probability theory, and point out subtleties present in model evaluation calibration that lead to refined interpretations of existing evaluation techniques. Lastly, we propose new ways to quantify and visualize model calibration in probabilistic classification, including novel multidimensional reliability diagrams.
Coverage

on what fraction of the data do the confidence / credible sets cover the true label?

\[
\sum_{m=1}^{M} \left[ y_m^* \in C(x_m) \right] \geq \frac{(1 - \alpha)}{M}
\]
Coverage

On what fraction of the data do the confidence / credible sets cover the true label?

\[
\sum_{m=1}^{M} \mathbb{1} \left[ y_m^* \in C(x_m) \right] \geq \frac{(1 - \alpha)}{M}
\]

Average size of the set is measured as well, since we want sets to be efficient.
Evaluating UQ

Calibration: can the model forecast its own performance?

Coverage: does the model meet the given error level?
V. Summary
Types of Uncertainty

Modeling Paradigms
- Frequentism: random data
- Bayesianism: random parameters

Practical Methods
- Frequentism: use ‘extra’ data, possibly synthesized
- Bayesianism: reduce computation with local approximations

Evaluation
- calibration: can the model forecast its own performance?
- coverage: does the model meet the tolerated level of error?
Types of Uncertainty

Modeling Paradigms

- Frequentism: random data
- Bayesianism: random parameters

Practical Methods

- Frequentism: use ‘extra’ data, possibly synthesized
- Bayesianism: reduce computation with local approximations

Evaluation

- calibration: can the model forecast its own performance?
- coverage: does the model meet the tolerated level of error?
Types of Uncertainty

Modeling Paradigms
- Frequentism: random data
- Bayesianism: random parameters

Practical Methods
- Frequentism: use ‘extra’ data, possibly synthesized
- Bayesianism: reduce computation with local approximations

Evaluation
- calibration: can the model forecast its own performance?
- coverage: does the model meet the tolerated level of error?
Types of Uncertainty

Modeling Paradigms
- Frequentism: random data
- Bayesianism: random parameters

Practical Methods
- Frequentism: use ‘extra’ data, possibly synthesized
- Bayesianism: reduce computation with local approximations

Evaluation
- calibration: can the model forecast its own performance?
- coverage: does the model meet the tolerated level of error?
Open Problems

- better methods for Bayesian computations
- guarantees in the era of deep learning
- setting more informative Bayesian priors
- quantifying uncertainty in structured, multi-step, or otherwise correlated tasks.
Thank You!  Questions?