The Role of Statistics in Machine Learning

A Perspective

Harrison B. Prosper

Department of Physics
Florida State University

Quarks To Cosmos with AI
Virtual Conference: July 12-16, 2021

Table of Contents

1 Introduction

2 Models and Loss

3 Uncertainty

4 Summary

We need to know when we don't know

For the most part, the field of machine learning has developed independently of the much older field of statistics.

However, over the past decade there has been a growing recognition of the importance of quantifying the quality, reliability, or accuracy of machine learning models.

Quantifying uncertainty is especially important in physics. However, even a cursory inspection of the living review `https://iml-wg.github.io/HEPML-LivingReview/` suggests that while uncertainty quantification of ML models is firmly on the stage, it has yet to reach its center.

Table of Contents

1 Introduction

2 Models and Loss

3 Uncertainty

4 Summary

It's mostly about loss

Most ML models are trained, that is fitted to data, $D = x_i, y_i$, $i = 1, \cdots, N$, by minimizing a function that statisticians refer to as the empirical risk,

$$R(\omega) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, f_i), \quad f_i \equiv f(x_i, \omega),$$

where $L(y, f)$ is a loss function and $f(x, \omega)$ is a model, which today is invariably a deep neural network with a huge number of parameters ω, such as ...

It's mostly about loss

...this one,

$f(x, \omega) = \mathrm{LSTM}(x, h, c, \omega)$

Inputs (x_t, h_{t-1}, c_{t-1})

$$g_t = \tanh(W_{ig} x_t + b_{ig} + W_{hg} h_{t-1} + b_{hg})$$
$$i_t = \sigma(W_{ii} x_t + b_{ii} + W_{hi} h_{t-1} + b_{hi})$$
$$f_t = \sigma(W_{if} x_t + b_{if} + W_{hf} h_{t-1} + b_{hf})$$

oututs (o_t, h_t, c_t)

$$o_t = \sigma(W_{io} x_t + b_{io} + W_{ho} h_{t-1} + b_{ho})$$
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$
$$h_t = o_t \odot \tanh(c_t),$$

being explored by our intrepid symbolic-AI hackathoners.

It's mostly about loss

$R(\omega)$ is minimized using some variation of stochastic gradient descent. The goal is to find a good approximation to the minimum not of the function $R(\omega)$ but rather of the risk functional,

$$R[f] = \int \int L(y, f)\, p(x, y)\, dx\, dy,$$

of which $R(\omega)$ is a Monte Carlo approximation.

It's mostly about loss

$R(\omega)$ is minimized using some variation of stochastic gradient descent. The goal is to find a good approximation to the minimum not of the function $R(\omega)$ but rather of the risk functional,

$$R[f] = \int \int L(y, f)\, p(x, y)\, dx\, dy,$$

of which $R(\omega)$ is a Monte Carlo approximation.

If the model $f(x, \omega)$ has enough *capacity*, then it will be possible to set its functional derivative

$$\begin{aligned} \frac{\delta R}{\delta f} &= \int \frac{\partial L}{\partial f}\, p(x, y)\, dy = 0, \\ &= p(x) \int \frac{\partial L}{\partial f}\, p(y|x)\, dy = 0. \end{aligned}$$

It's mostly about loss

Moreover, if

$$p(x) > 0 \quad \forall\, x,$$

then we conclude that the equation that is being solved implicitly using stochastic gradient descent is an *approximation* to

$$\boxed{\int \frac{\partial L}{\partial f}\, p(y|x)\, dy = 0.}$$

It's mostly about loss

Moreover, if

$$p(x) > 0 \quad \forall\, x,$$

then we conclude that the equation that is being solved implicitly using stochastic gradient descent is an *approximation* to

$$\boxed{\int \frac{\partial L}{\partial f}\, p(y|x)\, dy = 0.}$$

If the approximation is good, then the fitted model will generalize well. A model *generalizes* well if the fitted model $f(x, \hat{\omega})$ is a good approximation to the model $f(x, \omega^*)$, which is the solution to the equation above.

It's mostly about loss

Example 1: $L(f, y) = (y - f)^2$

In 1990, it was established[1] that the quadratic loss,

$$L(y, f) = (y - f)^2,$$

when used with the discrete targets $y \in \{1, 0\}$ associated with classes C_1 and C_2, respectively, and their associated priors $p(C_1)$ and $p(C_2)$, yields the result

$$f(x, \hat{\omega}) \approx p(C_1|x), = \frac{p(x|C_1)\, p(C_1)}{p(x|C_1)\, p(C_1) + p(x|C_2)\, p(C_2)},$$

which also follows from the cross entropy loss.

[1] E. A. Wan, "Neural network classification: a Bayesian interpretation," in IEEE Transactions on Neural Networks, vol. 1, no. 4, pp. 303-305, 1990

It's mostly about loss

Example 2: $L(f, y) = (y - f)^2$

If the targets y come from a continuous set, then the quadratic loss yields a model that approximates

$$f(x, \hat{\omega}) \approx \int y \, p(y|x) \, dy,$$

that is, the mean of the posterior density

$$p(y|x) = p(x|y) \, p(y) / p(x),$$

which, note, necessarily depends on the prior $p(y)$.

It's mostly about loss

Example 3: $L(f, y) = f\,\delta(y-1) + (f \log f - f)\,\delta(y)$

This loss function, with $p(C_1) = p(C_2)$, yields the result

$$\begin{aligned} f &= \exp(-r), \text{ where} \\ r &= \frac{p(C_1|x)}{p(C_2|x)} = \frac{p(x|C_1)}{p(x|C_2)}, \end{aligned}$$

that is, the negative log of the ML model approximates the likelihood ratio.

It's mostly about loss

Example 3: $L(f, y) = f\,\delta(y-1) + (f \log f - f)\,\delta(y)$

This loss function, with $p(C_1) = p(C_2)$, yields the result

$$
\begin{aligned}
f &= \exp(-r), \quad \text{where} \\
r &= \frac{p(C_1|x)}{p(C_2|x)} = \frac{p(x|C_1)}{p(x|C_2)},
\end{aligned}
$$

that is, the negative log of the ML model approximates the likelihood ratio.

The point here is that every ML model that uses the *same* training data and is fitted with the *same* loss function will approximate the *same* mathematical function.

Table of Contents

1 Introduction

2 Models and Loss

3 Uncertainty

4 Summary

The Bayesian approach

Since 2010, more than 2,500 papers have been published on uncertainty quantification, though few by particle physicists.

The most recent papers seem to be converging on the use of the Bayesian approach to quantify both aleatoric (i.e., statistical) as well as epistemic (i.e., model) uncertainty.

The Bayesian approach is conceptually simple: given a model $f(x, \omega)$, one computes the posterior density $p(\omega|D)$ of its parameters.

From $p(\omega|D)$ different measures of uncertainty can be computed for any quantity that depends on ω.

The Bayesian approach

Since 2010, more than 2,500 papers have been published on uncertainty quantification, though few by particle physicists.

The most recent papers seem to be converging on the use of the Bayesian approach to quantify both aleatoric (i.e., statistical) as well as epistemic (i.e., model) uncertainty.

The Bayesian approach is conceptually simple: given a model $f(x, \omega)$, one computes the posterior density $p(\omega|D)$ of its parameters.

From $p(\omega|D)$ different measures of uncertainty can be computed for any quantity that depends on ω.

The devil, as always, is in the details and the computational burden...

The Bayesian approach

The posterior density of the model parameters is given by Bayes' theorem,

$$p(\omega|D) = \frac{p(D, \omega)}{p(D)} = \frac{p(Y|X, \omega)\, p(X|\omega)\, \pi(\omega)}{p(Y|X)\, p(X)},$$

where $D \equiv X, Y$ and X and Y represent all values of x and y, respectively. The function $\pi(\omega)$ is a prior density over the model parameter space.

In practice, the training data X are independent of the model parameters; therefore, $p(X|\omega) = p(X)$ and we can write

$$\boxed{p(\omega|D) = \frac{p(Y|X, \omega)\, \pi(\omega)}{p(Y|X)}}.$$

The Bayesian approach

For relatively small models it is possible to represent the posterior density, $p(\omega|D)$, as a point cloud sampled using, for example, a Markov chain Monte Carlo (MCMC) method such as Hamiltonian Monte Carlo (HMC) or a variant.

The Bayesian approach

For relatively small models it is possible to represent the posterior density, $p(\omega|D)$, as a point cloud sampled using, for example, a Markov chain Monte Carlo (MCMC) method such as Hamiltonian Monte Carlo (HMC) or a variant.

Unfortunately, for large models MCMC is too slow and the trend now is to focus on tractable approximations $q(\omega|\theta)$ of $p(\omega|D)$ from which it is easy, and fast, to sample points ω_k.

The Bayesian approach

For relatively small models it is possible to represent the posterior density, $p(\omega|D)$, as a point cloud sampled using, for example, a Markov chain Monte Carlo (MCMC) method such as Hamiltonian Monte Carlo (HMC) or a variant.

Unfortunately, for large models MCMC is too slow and the trend now is to focus on tractable approximations $q(\omega|\theta)$ of $p(\omega|D)$ from which it is easy, and fast, to sample points ω_k.

The parameters θ are chosen to get the best match between $q(\omega|\theta)$ and $p(\omega|D)$, in principle by minimizing the Kullback-Leibler (KL) divergence

$$\mathrm{KL}(q|p) = \int q(\omega|\theta) \log \left[\frac{q(\omega|\theta)}{p(\omega|D)} \right] \, d\omega,$$

between them.

The Bayesian approach

Alas we can't minimize

$$\text{KL}(q|p) = \int q(\omega|\theta) \, \log \left[\frac{q(\omega|\theta)}{p(\omega|D)} \right] \, d\omega,$$

because it depends on the very thing, $p(\omega|D)$, we wish to approximate!

The Bayesian approach

Alas we can't minimize

$$\text{KL}(q|p) = \int q(\omega|\theta) \, \log \left[\frac{q(\omega|\theta)}{p(\omega|D)} \right] \, d\omega,$$

because it depends on the very thing, $p(\omega|D)$, we wish to approximate!

But, we can write the KL divergence as

$$\begin{aligned} \text{KL}(q|p) &= \log p(Y|X) \\ &\quad + \text{KL}(q|\pi) - \int q(\omega|\theta) \, \log p(Y|X, \omega) \, d\omega. \end{aligned}$$

The negative of the term in blue is called the the evidence lower bound (ELBO), which contains *known* quantities.

Monte Carlo dropout

Therefore, optimizing the function

$$\text{ELBO}(\theta) = \text{KL}(q|\pi) - \int q(\omega|\theta) \log p(Y|X,\omega) \, d\omega,$$

with respect to θ to find the best fit of $q(\omega|\theta)$ to the posterior density $p(\omega|D)$ is feasible, in principle, and, crucially, it is equivalent to optimizing the original KL divergence. But, alas, this is very challenging for large models and data sets.

[2] Yarin Gal and Zoubin Ghahramani, arXiv:1506.02142v6, 2016

Monte Carlo dropout

Therefore, optimizing the function

$$\text{ELBO}(\theta) = \text{KL}(q|\pi) - \int q(\omega|\theta) \log p(Y|X,\omega)\, d\omega,$$

with respect to θ to find the best fit of $q(\omega|\theta)$ to the posterior density $p(\omega|D)$ is feasible, in principle, and, crucially, it is equivalent to optimizing the original KL divergence. But, alas, this is very challenging for large models and data sets.

In 2016, Gal and Ghahramani[2] introduced an astonishingly simple method to quantify the uncertainty of any ML model, $f(x, z, \omega)$, trained using dropout random variables z before every layer.

[2] Yarin Gal and Zoubin Ghahramani, arXiv:1506.02142v6, 2016

Monte Carlo dropout

Gal and Ghahramani showed that if the dropout variables are used *after* training then M forward passes through a model for given input data x,

$$\hat{y}_m = f(x, z_m, \hat{\omega}), \quad m = 1, \cdots, M,$$

yields a sample $\{\hat{y}_m\}$ that is a point cloud approximation to the posterior density $p(\omega|D)$.

Monte Carlo dropout

Gal and Ghahramani showed that if the dropout variables are used *after* training then M forward passes through a model for given input data x,

$$\hat{y}_m = f(x, z_m, \hat{\omega}), \quad m = 1, \cdots, M,$$

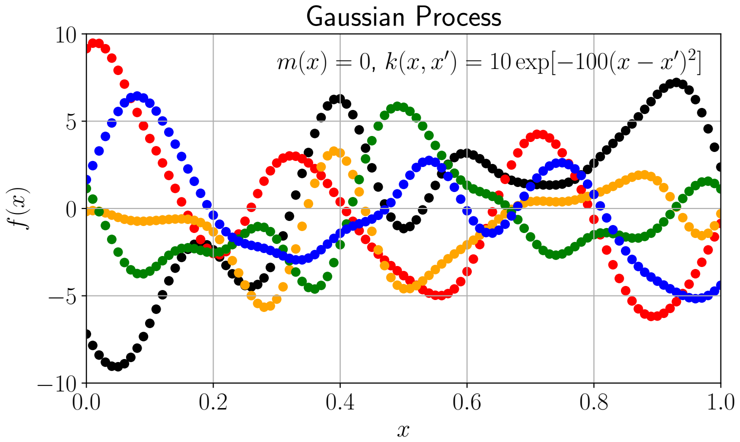
yields a sample $\{\hat{y}_m\}$ that is a point cloud approximation to the posterior density $p(\omega|D)$.

Specifically, the authors show that MC dropout is mathematically equivalent to approximating $p(\omega|D)$ using a function $q(\omega|\theta)$ constructed from a Gaussian process (GP).

Monte Carlo dropout as a Gaussian process

A GP is a multivariate Gaussian with an infinite number of random variables and is characterized by two functions

$$m(x) : \text{ mean function}, \quad k(x, x') \quad : \text{ covariance function.}$$



A *GP* can be used to approximate a probability density $p(f|D) = p(D|f)\,GP(f)$ over the space of functions.

So what's the problem?

So what's the problem?

The problem with all current methods is that the uncertainty estimates are not calibrated.

So what's the problem?

The problem with all current methods is that the uncertainty estimates are not calibrated.

- If a 68% credible interval is computed from a Bayesian method how reliable is it?

So what's the problem?

The problem with all current methods is that the uncertainty estimates are not calibrated.

- If a 68% credible interval is computed from a Bayesian method how reliable is it?
- With enough deep thinking one could, in principle, arrive at a prior $\pi(\omega)$ that would yield credible intervals that are reliable from a Bayesian viewpoint.

So what's the problem?

The problem with all current methods is that the uncertainty estimates are not calibrated.

- If a 68% credible interval is computed from a Bayesian method how reliable is it?
- With enough deep thinking one could, in principle, arrive at a prior $\pi(\omega)$ that would yield credible intervals that are reliable from a Bayesian viewpoint.
- However, most physicists would not find the credible intervals acceptable if they deviated greatly from 68% confidence intervals.

So what's the problem?

The problem with all current methods is that the uncertainty estimates are not calibrated.

- If a 68% credible interval is computed from a Bayesian method how reliable is it?
- With enough deep thinking one could, in principle, arrive at a prior $\pi(\omega)$ that would yield credible intervals that are reliable from a Bayesian viewpoint.
- However, most physicists would not find the credible intervals acceptable if they deviated greatly from 68% confidence intervals.

WANTED: *Feasible* and *general* ways to check the coverage of ML-based intervals, credible or otherwise.

Table of Contents

1 Introduction

2 Models and Loss

3 Uncertainty

4 Summary

Summary

- It widely recognized that uncertainty quantification of ML models is crucial and will become more so as we become more dependent on them.

Summary

- It widely recognized that uncertainty quantification of ML models is crucial and will become more so as we become more dependent on them.
- The machine learning/AI field seems to be converging on the use of Bayesian methods to quantify uncertainty.

Summary

- It widely recognized that uncertainty quantification of ML models is crucial and will become more so as we become more dependent on them.
- The machine learning/AI field seems to be converging on the use of Bayesian methods to quantify uncertainty.
- However, these methods need to be calibrated.

Summary

- It widely recognized that uncertainty quantification of ML models is crucial and will become more so as we become more dependent on them.
- The machine learning/AI field seems to be converging on the use of Bayesian methods to quantify uncertainty.
- However, these methods need to be calibrated.
- For many physicists, and many statisticians, the preferred way to do so is to calibrate *all* proposed intervals using frequentist methods.

Summary

- It widely recognized that uncertainty quantification of ML models is crucial and will become more so as we become more dependent on them.
- The machine learning/AI field seems to be converging on the use of Bayesian methods to quantify uncertainty.
- However, these methods need to be calibrated.
- For many physicists, and many statisticians, the preferred way to do so is to calibrate *all* proposed intervals using frequentist methods.
- The problem is how to do this in practice.