juliacon

Effortless Bayesian Deep Learning through Laplace Redux

Patrick Altmeyer[1]

[1]Delft University of Technology

ABSTRACT

Treating deep neural networks probabilistically comes with numerous advantages including improved robustness and greater interpretability. These factors are key to building artificial intelligence (AI) that is trustworthy. A drawback commonly associated with existing Bayesian methods is that they increase computational costs. Recent work has shown that Bayesian deep learning can be effortless through Laplace approximation. We propose a small Julia package, ‘LaplaceRedux.jl‘ that implements this new approach for deep neural network trained in ‘Flux.jl‘.

Keywords

Probabilistic Machine Learning, Laplace Approximation, Deep Learning, Artificial Intelligence

1. Background

Over the past decade Deep Learning (DL) has arguably been one of the dominating subdisciplines of Artificial Intelligence. Despite the tremendous success of deep neural networks, practitioners and researchers have also pointed to a vast number of pitfalls that have so far inhibited the use of DL in safety-critical applications. Among other things these pitfalls include a lack adversarial robustness [4] and an inherent opaqueness of deep neural networks, often described as the Black-Box problem. The number of parameters relative to the size of the available data is generally huge:

> [...] deep neural networks are typically very underspecified by the available data, and [...] parameters [therefore] correspond to a diverse variety of compelling explanations for the data. [8]

A scenario like this very much calls for treating predictions from deep learning models probabilistically [8]. It is therefore not surprising that interest in Bayesian Deep Learning has grown in recent years as researchers have tackled the problem from a wide range of angles including: MCMC (see `Turing`), Variational Inference [1], Monte Carlo Dropout [3] and Deep Ensembles [6]. Laplace Redux ([5],[2]) is one of the most recent and promising approaches to Bayesian neural networks (BNN).

2. Laplace Approximation for Deep Learning

Let $\mathcal{D} = \{x, y\}_{n=1}^N$ denote our feature-label pairs and let $f(x; \theta) = y$ denote some deep neural network specified by its parameters θ. We are interested in eventually estimating the posterior predictive distribution given by the following Bayesian model average (BMA):

$$p(y|x,\mathcal{D}) = \int p(y|x,\theta)p(\theta|\mathcal{D})d\theta \tag{1}$$

To do so we first need to compute the weight posterior $p(\theta|\mathcal{D})$. Laplace Approximation (LA) relies on the fact that the second-order Taylor expansion of this posteriour amounts to a multivariate Gaussian centered around the maximum a posteriori (MAP) estimate $\hat{\theta} = \arg\max_\theta p(\theta|\mathcal{D})$ with covariance equal to the inverse Hessian of our loss function evaluated at the mode $\hat{\Sigma} = (\mathbf{H}(\hat{\theta}))^{-1}$. In other words, we can train our deep neural network in the usual way by minimizing the negative log likelihood $\ell(\mathbf{w}) = -\log p(y|x,\mathcal{D})$. To obtain Gaussian LA weight posterior we then only need to compute the Hessian evaluated at the obtained MAP estimate.

Laplace Approximation itself actually dates back to the 18th century, but despite its simplicity it has long been neglected by the deep learning community. One reason for this may be that for large neural networks with many parameters the exact Hessian computation is prohibitive. On can rely on linearised approximations of the Hessian, but those still scale quandratically in the number of parameters. Fortunately, recent work has shown that block-diagonal factorizations can be successfully applied in this context [2].

Another reason for why LA has been neglected in the past, is that early attempts at using LA for deep learning actually failed: simply sampling from the LA weight posterior to compute the exact BNN posterior predictive distribution in Equation 1 does not work when using approximations for the Hessian [7]. Instead we rely on a linear expansion of predictive around mode as demonstrated by Immer et al. (2020) [5]. Formally, we use a locally linearized version of our BNN:

3. Conclusions

Bleh

4. Acknowledgements

5. References

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

[2] Erik Daxberger, Agustinus Kristiadi, Alexander Immer, Runa Eschenhagen, Matthias Bauer, and Philipp Hennig. Laplace Redux-Effortless Bayesian Deep Learning. 34.

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

[4] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. arxiv:1412.6572.

[5] Alexander Immer, Maciej Korzepa, and Matthias Bauer. Improving predictions of bayesian neural networks via local linearization. arxiv:2008.08400.

[6] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. arxiv:1612.01474.

[7] Neil David Lawrence. Variational inference in probabilistic models.

[8] Andrew Gordon Wilson. The case for Bayesian deep learning. arxiv:2001.10995.