Bayesian Neural Networks

Ilya Zharikov, Roman Isachenko, Artem Bochkarev

Abstract

The recent advances in deep learning allow us to solve really hard real word problems with high accuracy. What these methods lack is interpretability of the model and uncertainty guarantees for the predictions. Bayesian framework allows to solve both this two issues, and with recent development in variational inference technique it is possible to take the best of two worlds and implement bayesian neural network. Our project is dedicated to testing different priors for neurons weights and exploring advantages and disadvantages of bayesian approach in deep learning.

Introduction

Deep learning [1] is one of the most promising and quickly advancing areas of machine learning research. Deep models proved to be superior to other algorithms in many fields such as image classification [2], speech recognition [3], unsupervised learning [4] or reinforcement learning [5]. Still, these models are mostly used as some black box in order to solve very difficult problems. We often want to know not only the prediction of the network on given imput, but we would like to understand the uncertainty [6] of the model on this object. It is also desireable to be able to do transfer learning [7], i.e. be able to obtain correct results on new types of data, using previous model as prior knowledge.

On the other side of the spectrum there is bayesian approach [8]. The main advantage of this framework is that we operate with parameters distribution, instead of their point estimate. This allows us to calculate uncertainties and other useful statistics. Another advantage of the bayesian approach is that we are free to select prior distribution for the parameters of our model [9], which may include real knowledge or some non-informative assumption. Finally, we may want to build hierarchical models because data is too complex and good priors are hard to guess.

With recent advances in computational technology and GPU it is very compelling to try mixing deep learning with bayesian framework. This would allow us to achive the same high performance of neural network, as well give us more freedom with different prior selection and interpretability of the results using uncertainty prediction. One of the obstacles of such approach is that full bayesian approach is very memory expensive, given huge parameters number (we need to store whole distribution instead of one number). What is more important, calculation of the model evidence is intractable for high-dimensional data. MCMC sampling methods [10] like Metropolis-Hastings or Gibbs are too computationally expensive to perform on big models.

In our project we decided to implement alternative way of approximation of posterior probability, namely ADVI [11]. The main idea of the method is that we constitute true posterior with some distribution from known parametric family and then we use optimization procedure to minimize their difference.

We conducted computational experiments on several datasets. We used PyMC3 [12] and Lasagne [13] for probabilistic and deep learning frameworks respectively. We used synthetic datasets as well as MNIST hand-written digits dataset. For this datasets we constructed bayesian neural network models, optimized them with ADVI, played with different priors and drew conclusions from model uncertainty.

Problem Statement

Bayesian approach

Let suppose that $\mathbf{X}$ is the observed data which comes from the unknown distribution $p(\mathbf{X})$. This distribution defines our model and is called model evidence. We assume that our model also contains latent variables $\boldsymbol{\theta}$. The likelihood function is given by conditional probability distribution $p(\mathbf{X}|\boldsymbol{\theta})$. If we know the prior distribution $p(\boldsymbol{\theta})$ over hidden variables, we could use Bayes' theorem to derive the posterior distribution $p(\boldsymbol{\theta}|\mathbf{X})$

$$p(\boldsymbol{\theta}|\mathbf{X}) = \frac{p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(\mathbf{X})}, \tag{1}$$

where the evidence function $p(\mathbf{X})$ could be obtained by integration over hidden variables

$$p(\mathbf{X}) = \int p(\mathbf{X}|\boldsymbol{\theta})p(\boldsymbol{\theta})p(\boldsymbol{\theta}).$$

The posterior distribution shows the transformed initial prior knowledge of $\boldsymbol{\theta}$ given the observed data $\mathbf{X}$.

The main goal of bayesian inference to compute the posterior distribution. In some cases we could derive the posterior in the closed form formula. However, it is impossible for complex models where the latent variables lies in high-dimensional space. The problem is the denominator of (1), since it is obtained by integrating over all possible values of $\boldsymbol{\theta}$.

Suppose that we have the posterior distribution $p(\boldsymbol{\theta}|\mathbf{X})$, then we could use the usual Maximum A Posteriori (MAP) approach to obtain point estimate for the parameters

$$\boldsymbol{\theta}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} \left[\ln p(\boldsymbol{\theta}|\mathbf{X})\right] = \arg\max_{\boldsymbol{\theta}} \left[\ln p(\mathbf{X}|\boldsymbol{\theta}) + \ln p(\boldsymbol{\theta})\right].$$

In the case of flat prior distribution $p(\boldsymbol{\theta}) = \text{const}$, this estimate coincides with the Maximum Likelihood Estimation (MLE) approach

$$\boldsymbol{\theta}_{\text{MLE}} = \arg\max_{\boldsymbol{\theta}} \left[\ln p(\mathbf{X}|\boldsymbol{\theta})\right].$$

Variational inference

One of the widely used approach to get posterior distribution is sampling methods such as Metropolis-Hastings, Gibbs, NUTS algorithms. The general idea behind sampling methods is to obtain procedure for generating samples from the true posterior distribution. Most of these methods is based on Monte Carlo Markov Chains (MCMC) procedures.

Contributions

Ilya Zharikov programmed main classes and routines, built project architecture. Also tried to implement experiment on CIFAR-10 and cats-vs-dogs dataset.

Roman Isachenko conducted experiments and drew conclusions from synthetic datasets.

Artem Bochkarev conducted experiments and drew conclusions from MNIST dataset.

Everyone were equally involved in creating presentation, writing this report, as well as general project development.

References

[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[3] George E Dahl, Dong Yu, Li Deng, and Alex Acero. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):30–42, 2012.

[4] Quoc V Le. Building high-level features using large scale unsupervised learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8595–8598. IEEE, 2013.

[5] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[6] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

[7] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[8] Christopher M Bishop. Pattern recognition. *Machine Learning*, 128:1–58, 2006.

[9] William J Browne, David Draper, et al. A comparison of bayesian and likelihood-based methods for fitting multilevel models. *Bayesian analysis*, 1(3):473–514, 2006.

[10] Larry Wasserman. *All of statistics: a concise course in statistical inference.* Springer Science & Business Media, 2013.

[11] Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *arXiv preprint arXiv:1603.00788*, 2016.

[12] John Salvatier, Thomas V Wiecki, and Christopher Fonnesbeck. Probabilistic programming in python using pymc3. *PeerJ Computer Science*, 2:e55, 2016.

[13] Sander Dieleman, Jan Schluter, Colin Raffel, Eben Olson, et al. Lasagne: First release., August 2015.