Posterior distribution analysis for Bayesian inference in neural networks

Pavel Myshkov
Department of Computer Science
University College London
pavel.myshkov.14@ucl.ac.uk

Simon Julier
Department of Computer Science
University College London
s.julier@ucl.ac.uk

Abstract

This study explores the posterior predictive distributions obtained with various Bayesian inference methods for neural networks. The quality of the distributions is assessed both visually and quantitatively using Kullback–Leibler (KL) divergence, Kolmogorov–Smirnov (KS) distance and precision-recall scores. We perform the analysis using a synthetic dataset that allows for a more detailed examination of the methods, and validate the findings on larger datasets. We find that among the recently proposed techniques, the simpler ones – Stochastic Gradient Langevin Dynamics (SGLD) and MC Dropout – are able to consistently provide good approximations to the "true" posterior, at the same time not requiring extensive tuning of parameters.

1 Introduction

The combination of Bayesian inference and deep learning has gained a lot of attention recently. One of the key advantages of this combination is the ability to quantify uncertainty of the predictions. Moreover, certain applications involving decision making would gain more from a better estimation of the shape of the predictive distribution or its properties, especially variance, than from a higher accuracy of the prediction itself. However, the quality of the posterior predictive distributions is usually not studied in detail, with comparisons made only based on the prediction errors. We aim to fill this gap by providing a careful study of the distributions obtained with these methods, as well assessing the quality of the distributions depending on the computational time spent running the methods. These results should help practitioners choose methods best suited to their specific project needs, and help researchers in further development of these techniques.

Let us consider a supervised learning problem in which we are given a set of training examples $\mathcal{D} = \{x_i, y_i\}_{i=1}^N$, where x_i is the i'th input feature vector and y_i is the corresponding output. To perform Bayesian inference with a neural network we construct a posterior distribution over the network parameters $P(\theta|\mathcal{D}) \propto P(\mathcal{D}|\theta)P(\theta)$, where $P(\theta)$ is the prior distribution of the network parameters and $P(\mathcal{D}|\theta) = \prod_{(x_i,y_i)\in\mathcal{D}} P(y_i|x_i,\theta)$ is the likelihood term. The posterior predictive distribution of y given a test point x is then $P(y|x,\mathcal{D}) = \int P(y|x,\theta)P(\theta|\mathcal{D})\mathrm{d}\theta$.

Direct computation of the posterior distribution is intractable, and to make the Bayesian inference we need to either use a deterministic approximation scheme where a simpler form of the posterior distribution is assumed, or MCMC sampling. A deterministic approximation is made in Probabilistic BackPropagation (PBP) [1], an online version of the expectation propagation (EP) [2] scheme, and in Bayes By Backprop (BBB) [3], which is a variational inference method that uses the reparameterisation trick [4] to optimise the variational free energy. Importantly, both these methods assume the posterior distribution to be factorised Gaussian, and therefore may not perform well in problems where the ability learn correlated uncertainties of the parameters is important.

Workshop on Bayesian Deep Learning, NIPS 2016, Barcelona, Spain.

The MCMC methods able to sample from the posterior of a neural network are based on Hamiltonian dynamics. Specifically, the Hybrid Monte Carlo (HMC) [5] is a "gold standard" method for Bayesian inference in neural networks. However, HMC is a batch method that requires computations over the whole dataset at each step, and therefore is not suited for large problems. A simple replacement of the data dependent calculations to use mini-batches leads to an incorrect stationary distribution [6]. To address this problem, a number of MCMC samplers that use stochastic gradients have been proposed recently. A simplest variant, Stochastic Gradient Langevin Dynamics (SGLD) [7], is a one step Hamiltonian sampler that in essence performs a stochastic gradient descent with an added Gaussian noise. Its extension, preconditioned SGLD (pSGLD) [8], aims to address the low mixing rate of SGLD by flattenning out the curvatures of the target density surface. Another extension, Stochastic Gradient HMC (SG-HMC) [6], adds a friction term to compensate for the noise introduced by the stochasticity of the gradient. All these methods are shown to theoretically convergence to the true posterior under the assumption of infinite schedule of decreasing step sizes. Since this assumption can not be met in practice there will be a bias in the sample representation of the predictive distributions.

Finally, MC Dropout [9] can be used to obtain an approximation to the posterior predictive distribution by applying a standard dropout technique [10]. In terms of the target loss, all these methods have been shown to achieve extremely competitive results, with Dropout and PBP usually outperforming the other methods on standard datasets [9], [1]. However, in terms of the the quality of the uncertainty quantification, the performance of these these methods is unclear. In the rest of this paper we will analyse these distributions and assess their qualities using various metrics.

2 Evaluation framework

In this study we consider regression problems with the noise modelled as Gaussian, i.e. $P(y_i|x_i, \theta) = N(f(x_i), \sigma_n^2)$, where $f(x|\theta)$ is the neural network and σ_n^2 is the variance of the noise. Priors on network parameters are modelled as independent zero mean Gaussians with variance σ_w^2 common for all weights. We use feed-forward neural networks with ReLU [11] transfer functions, as it is a popular choice in the modern deep learning architectures and because other transfer functions saturate quickly, leaving less room for prediction uncertainty [9]. In all experiments the networks are initialised with a truncated Gaussian and operate on a normalised data.

2.1 Methods

We use a multi-chain HMC to construct an accurate posterior predictive distribution that we then treat as the "true" predictive distribution. This computationally expensive procedure limits the size of the problems we can consider. However, HMC relies on computing gradients via back propagation and therefore benefits from a GPU implementation, which we use to address more interesting scenarios that are not easily accessible with standard implementations of HMC.

In each experiment we allow for a long burn in stage for the HMC during which step sizes are tuned to match the target acceptance rate of 90-95%. When possible, we use Gibbs sampling to learn the prior variance of the noise and weights jointly with network parameters. This process does not work in all scenarios, failing especially often when there is far less data than network weights. In such situations the parameters of the prior distributions on the noise and weights can explain the data quicker and easier than the parameters of the network. This problem is discussed in more detail in [3], [12]. When we are unable to learn the variances automatically we search for best values based on test errors, and we do the same with other parameters such as the number of integration steps.

For the target methods we use the recommended values for parameters. For BBB and Dropout we use Adam optimiser as recommended by the authors of these methods. For MCMC methods we set the step sizes and variances to those obtained form running the HMC and anneal step sizes as $a(b+t)^{-0.51}$ that satisfies the decreasing step size schedule required by these methods if the sampling was continued infinitely.

2.2 Metrics

To quantitatively measure the quality of the predictive distributions against the "true" distribution we use several metrics:

- KL-divergence is a popular measure of difference between two distributions. To calculate it for regression problems we first apply nonparametric kernel density estimation with a Gaussian kernel, since the observed predictive distributions are usually unimodal and smooth. We place an upper bound of 1 to handle the situations where the supports of the densities do not overlap well. This usually occurs when the predicted means differ significantly and such cases are already reflected in the higher RMSE values, so we do not want to severely penalise KL metric for this. Hence the rather small bound.
- Kolmogorov-Smirnov distance for 2-sample tests (KS distance) which measures the maximum point-wise difference between two empirical distribution functions. It is bounded in $[0, 1]$ and handles situations of not overlapping supports naturally, but is poorly sensitive to deviations occurring in the tails.
- Finally, we consider how uncertainty estimations will likely be applied in practical problems. An obvious use would be to estimate confidence intervals together with predictions and make a decision based on whether the confidence interval contains a certain threshold value. For example, a financial firm might refuse to invest in an asset whose 90% confidence interval for predicted return contains a negative threshold value. In light of this we construct the usual precision and recall metrics by estimating the probability that a uniformly distributed threshold covered by the target interval is also covered by the true interval, and vice versa. We use 90% confidence intervals and also compute F1 scores for easier comparison.

All methods represent the predictive distributions by samples, with the exception of PBP, for which we draw samples explicitly from the Gaussians with means and variances computed by PBP.

3 Analysis on synthetic data

We start our analysis by considering a one dimensional regression on a synthetically generated data set. One of the advantages of this problem is that the quality of the produced distributions can be assessed visually. We assume the data generating process to be $y = (1 + x)\sin(10\tanh(x)) + \epsilon$, where $\epsilon \sim N(0,\ 0.04)$ and x is sampled from a Gaussian $N(0,\ 4)$ truncated to $[-4,\ 4]$. The training set consists of 50 data points, while the test set contains 1200 points uniformly distributed in $[-5, 5]$. The varying curvature of the target function should help better understand whether the model underfits or overfits the data, whereas the varying density of data points should give rise to uncertainty in estimations.

We fit the data using a neural network with 2 hidden layers, each containing 100 neurons. The "true" posterior distribution is obtained from a 6-chain HMC with 50 integration steps and a persistent momentum of 0.5 to further suppress the random walk behaviour. A total of $60,000$ samples were drawn, with the first 50% in the burn in phase.

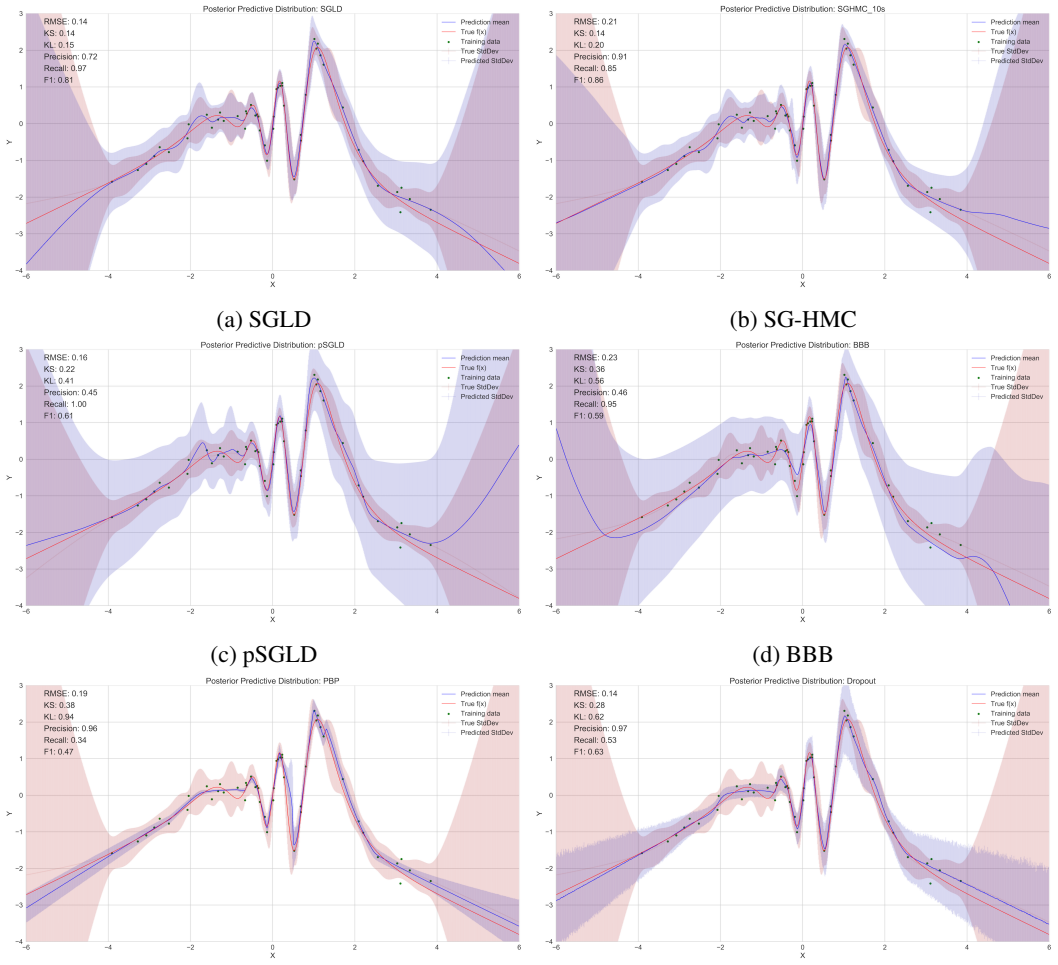
Since BBB and PBP assume a factorised Gaussian approximation of the posterior we can briefly assess whether the HMC posterior possesses these properties. Applying χ^2-tests, for 96% of weights the samples suggest the hypothesis of normality should be rejected with p-value below 0.01. In addition, 36% of pairs of weights have an absolute correlation coefficient of 0.25 or higher (we only take samples obtained by one HMC chain for these tests and apply 1:100 thinning).

The predictive distributions produced by the six methods considered in this study are shown on Fig. 1, and Fig. 2 shows the calculated metrics against the computational time spent. The results obtained using the final 50% of the samples are summarised in Table 1.

Looking at the distribution plots we see that SGLD and SG-HMC produce the predictive distributions that visually are the closest to the "true" one. The bias introduced by the stochasticity of the gradient results in a slightly overestimated variance and therefore a lower precision score.

For SG-HMC these results were obtained with using only 10 integration steps. An increase in the number of integration steps leads to an underestimation of the variance (e.g. recall 0.81 with 20 steps). This is likely caused by the friction that gradually eliminates the initial momentum and hence the ends of the trajectories are mostly controlled by the gradient of the energy.

The preconditioned SGLD offers better convergence in terms of RMSE, but it seems that the simplifying assumptions made for the computational considerations in formulation of this method



(a) SGLD (b) SG-HMC

(c) pSGLD (d) BBB

(e) PBP (f) Dropout

Figure 1: Posterior predictive distributions obtained on the synthetic data set. Green dots indicate the training data points, red line is the true target function, and red shaded area shows the mean and 3 standard deviations of the predictions made by HMC. Blue line and shaded areas are the mean and 3 standard deviations of the predictions produced by the respective methods. The learned noise variance (constant) is not shown for visualisation purposes.

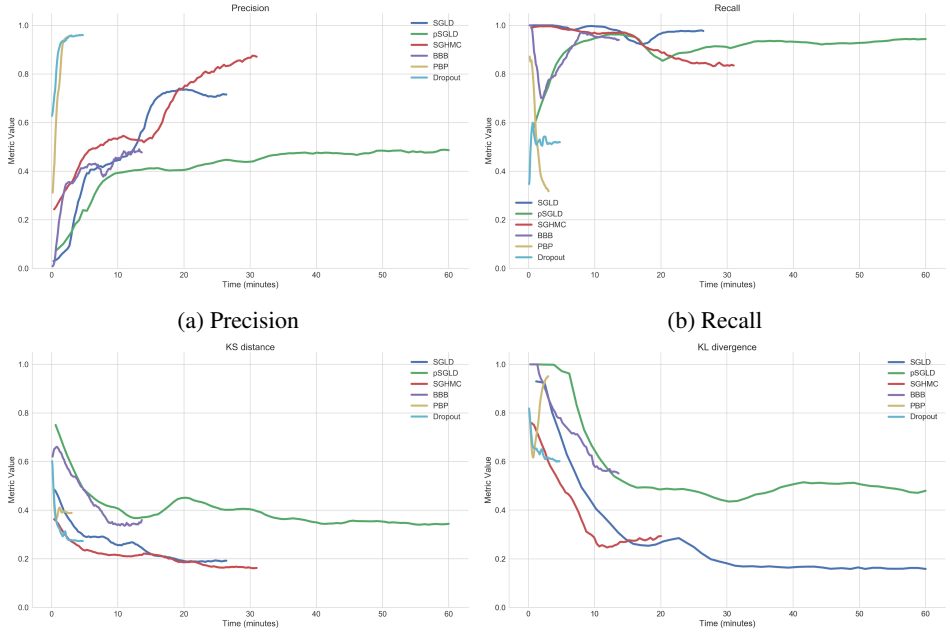
negatively affect the shape of the predictive distribution. Both SG-HMC and pSGLD require additional tuning of parameters (friction and curvature respectively) as compared to SGLD.

BBB and PBP both are able to fit the data well in terms of RMSE, but the quality of the distribution is not as good as that of the sampling methods. However, this data set is not well suited for these methods, especially for PBP which is geared towards large scale learning and assumes only a small number of passes (below 100) through the data during learning. To fit this data set more than 500 passes are required.

Finally, dropout produces an excellent RMSE and Precision scores. The variance is considerably underestimated, but not at the level of PBP. In addition, as can be seen from the time plots, it shows noticeable advantage over the other methods in terms of computational speed.

4 Experiments on standard data sets

To further study the discussed methods we evaluate the performance on several regression data sets. We use a single layer network with 50 units for smaller data sets of Boston Housing (506 inputs) and Energy Efficiency (768 inputs), and a 2 layer network with 100 neurons in each layer for the larger



(a) Precision

(b) Recall

(c) KS distance

(d) KL divergence

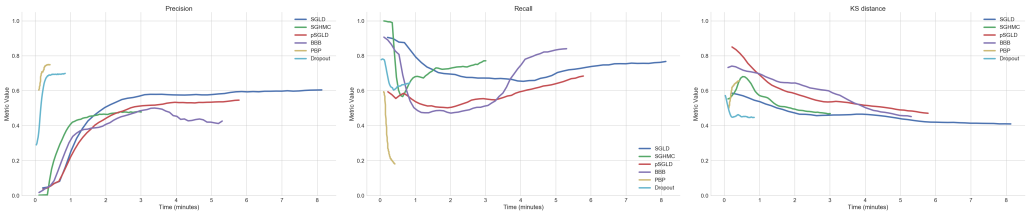
Figure 2: Precision, Recall, KS and KL values obtained on the synthetic data set and shown against the computational time spent. The values are computed based on a 10% wide sliding window over generated samples that upon passing the burn in point extends from that point onwards so that the last value on the plot corresponds to the metric values computed using the last 50% of the samples.

Table 1: Results on the synthetic data

Method	RMSE	KS	KL	Precision	Recall	F1
SGLD	0.14	**0.14**	**0.15**	0.72	**0.97**	0.81
SGHMC	0.20	0.14	0.18	0.90	0.87	**0.86**
pSGLD	0.16	0.22	0.41	0.45	1.00	0.61
BBB	0.21	0.35	0.55	0.48	0.95	0.60
PBP	0.19	0.38	0.94	0.96	0.34	0.47
Dropout	**0.14**	0.27	0.63	**0.97**	0.53	0.63

Power Plant data set (9,568 inputs). Each experiment is run 4 times using a sample of 75% of input points for training and 25% for testing. The results are summarised in Table 2, and Figures 3, 4 and 5 show the metrics computed on the respective data sets against the computational time spent running the methods.

The results obtained on these datasets generally agree with what we concluded from the analysis performed on the synthetic data. The more sophisticated stochastic methods, pSGLD, SG-HMC,



(a) Precision

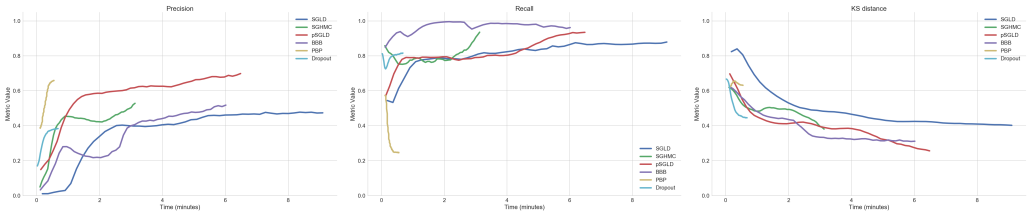
(b) Recall

(c) KS distance

Figure 3: Precision, Recall and KS distance computed for the Boston Housing data set against the computational time spent.

Table 2: Results on the synthetic data

Method	RMSE	KS	KL	Precision	Recall
			Boston Housing		
SGLD	3.99 ± 0.004	$\mathbf{0.40 \pm 0.003}$	$\mathbf{0.56 \pm 0.003}$	0.64 ± 0.002	0.73 ± 0.002
SGHMC	4.65 ± 0.014	0.44 ± 0.002	0.65 ± 0.005	0.50 ± 0.002	0.80 ± 0.002
pSGLD	7.28 ± 0.010	0.44 ± 0.001	0.56 ± 0.001	0.57 ± 0.002	0.69 ± 0.004
BBB	4.52 ± 0.005	0.42 ± 0.000	0.69 ± 0.003	0.45 ± 0.002	$\mathbf{0.84 \pm 0.001}$
PBP	3.83 ± 0.000	0.67 ± 0.001	1.00 ± 0.001	$\mathbf{0.76 \pm 0.003}$	0.14 ± 0.000
Dropout	$\mathbf{3.62 \pm 0.003}$	0.44 ± 0.000	0.68 ± 0.003	0.71 ± 0.003	0.65 ± 0.003
			Energy Efficiency		
SGLD	1.63 ± 0.002	0.37 ± 0.001	0.51 ± 0.008	0.46 ± 0.009	0.90 ± 0.004
SGHMC	0.94 ± 0.006	0.23 ± 0.002	0.27 ± 0.004	0.62 ± 0.001	$\mathbf{0.98 \pm 0.003}$
pSGLD	$\mathbf{0.68 \pm 0.001}$	$\mathbf{0.17 \pm 0.001}$	$\mathbf{0.20 \pm 0.004}$	$\mathbf{0.73 \pm 0.007}$	0.97 ± 0.002
BBB	0.68 ± 0.003	0.30 ± 0.001	0.51 ± 0.017	0.60 ± 0.005	0.91 ± 0.003
PBP	1.69 ± 0.000	0.64 ± 0.000	1.00 ± 0.001	0.69 ± 0.002	0.21 ± 0.001
Dropout	0.86 ± 0.001	0.48 ± 0.002	0.65 ± 0.003	0.58 ± 0.003	0.76 ± 0.004
			Power Plant		
SGLD	4.25 ± 0.001	0.34 ± 0.001	0.64 ± 0.012	0.85 ± 0.000	0.62 ± 0.003
SGHMC	4.05 ± 0.001	$\mathbf{0.24 \pm 0.001}$	$\mathbf{0.30 \pm 0.001}$	0.84 ± 0.001	0.82 ± 0.002
pSGLD	4.11 ± 0.002	0.24 ± 0.003	0.35 ± 0.010	0.55 ± 0.010	$\mathbf{0.99 \pm 0.002}$
PBP	4.14 ± 0.000	0.38 ± 0.001	0.84 ± 0.005	$\mathbf{0.95 \pm 0.001}$	0.44 ± 0.001
Dropout	$\mathbf{3.92 \pm 0.000}$	0.31 ± 0.002	0.41 ± 0.003	0.64 ± 0.003	0.90 ± 0.001

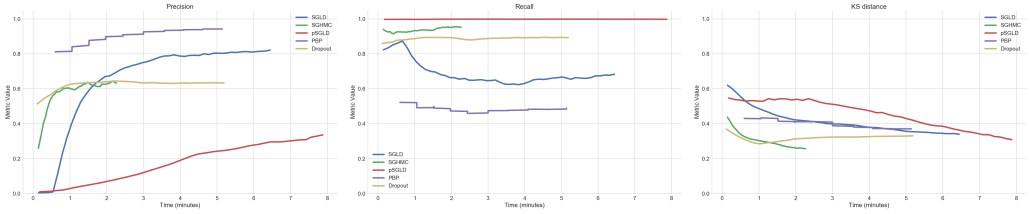


(a) Precision (b) Recall (c) KS distance

Figure 4: Precision, Recall and KS distance computed for the Energy Efficiency data set against the computational time spent.

can show both very strong and rather poor results. A careful parameter tuning might help improve performance in some cases, but these methods are computationally expensive and such a tuning will require a considerable computational overhead.

BBB achieves average scores rather quickly but is unable to improve them with additional training time. PBP shows high precision values but the uncertainty of the predictions decreases during



(a) Precision (b) Recall (c) KS distance

Figure 5: Precision, Recall and KS distance computed for the Energy Efficiency data set against the computational time spent.

learning, resulting in low recall scores. This is significantly improved on a larger Power Plant data set, confirming the statement that this method is aimed at large scale learning.

Finally, a more consistent performance is shown by SGLD and Dropout methods, and they effectively complement each other. Dropout shows ones of the smallest prediction errors, high precision and often converges significantly faster than the other methods. At the same time SGLD has consistently good KS/KL scores and judging by the plots is able to improve them if given more time to converge.

5 Conclusion

The analysis of the predictive distributions obtained with Bayesian inference methods can improve understanding of how well the uncertainty is quantified by a Bayesian neural network. By performing a careful analysis of these distributions we find that simpler methods, MC Dropout and SGLD, are able to produce good approximations to the predictive distributions while not requiring extensive tuning of parameters. Dropout usually underestimates the variance and thus its predictions are overconfident, and this should be taken into account if these confidence estimates are used in a decision making process. SGLD mixes rather slowly, but given enough time it can consistently obtain some of the best approximations to the posterior predictive distributions.

Further analysis into these methods should consider classification problems and include convolutional and recurrent neural networks architectures.

The code used for this analysis is available on Github.

References

[1] José Miguel Hernández-Lobato and Ryan P. Adams. "Probabilistic Backpropagation for Scalable Learning of Bayesian Neural Networks". In: *ICML*. 2015.

[2] Thomas P. Minka. "A family of algorithms for approximate Bayesian inference". In: *PhD thesis, Massachusetts Institute of Technology*. 2001.

[3] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. "Weight Uncertainty in Neural Networks". In: *CoRR* abs/1505.05424 (2015).

[4] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic Backpropagation and Approximate Inference in Deep Generative Models". In: *ICML*. 2014.

[5] Radford M Neal. "Bayesian Learning for Neural Networks". In: *PhD thesis, University of Toronto*. 1995.

[6] Tianqi Chen, Emily B. Fox, and Carlos Guestrin. "Stochastic Gradient Hamiltonian Monte Carlo". In: *ICML*. 2014.

[7] Max Welling and Yee Whye Teh. "Bayesian Learning via Stochastic Gradient Langevin Dynamics". In: *ICML*. 2011.

[8] Chunyuan Li, Changyou Chen, David E. Carlson, and Lawrence Carin. "Preconditioned Stochastic Gradient Langevin Dynamics for Deep Neural Networks". In: *AAAI*. 2016.

[9] Yarin Gal and Zoubin Ghahramani. "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning". In: *ICML*. 2016.

[10] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. "Dropout: a simple way to prevent neural networks from overfitting". In: *Journal of Machine Learning Research* 15 (2014), pp. 1929–1958.

[11] Vinod Nair and Geoffrey E. Hinton. "Rectified Linear Units Improve Restricted Boltzmann Machines". In: *ICML*. 2010.

[12] Andrew Gelman. "Objections to Bayesian Statistics". In: 2008.