# On the Importance of Strong Baselines in Bayesian Deep Learning

**Jishnu Mukhoti**
Department of Computer Science
University of Oxford
jishnu.mukhoti@cs.ox.ac.uk

**Pontus Stenetorp**
Department of Computer Science
University College London
p.stenetorp@cs.ucl.ac.uk

**Yarin Gal**
Department of Computer Science
University of Oxford
yarin@cs.ox.ac.uk

## Abstract

Like all sub-fields of machine learning, Bayesian Deep Learning is driven by empirical validation of its theoretical proposals. Given the many aspects of an experiment, it is always possible that minor or even major experimental flaws can slip by both authors and reviewers. One of the most popular experiments used to evaluate approximate inference techniques is the regression experiment on UCI datasets. However, in this experiment, models which have been trained to convergence have often been compared with baselines trained only for a fixed number of iterations. What we find is that if we take a well-established baseline and evaluate it under the same experimental settings, it shows significant improvements in performance. In fact, it outperforms or performs competitively with numerous to several methods that when they were introduced claimed to be superior to the very same baseline method. Hence, by exposing this flaw in experimental procedure, we highlight the importance of using identical experimental setups to evaluate, compare and benchmark methods in Bayesian Deep Learning.

## 1 Introduction

Empiricism is at the very core of machine learning research, where we demand that new methods and approaches compare favorably to previously introduced work. We expect this in terms of performance on either artificially generated data that highlights specific challenges or real-world data for specific tasks. In this process, we implicitly trust our fellow scientists and the reviewers of their work to note discrepancies (intentional or not, to err is after all human) in the experimental setting – such as for example, any kind of overfitting.[1] Recently, several studies have noted empirical shortcomings in the machine learning literature. For example, Henderson et al. (2017) noted that due to non-determinism, variance, and lack of significance metrics, it is difficult to judge whether claimed advances in reinforcement learning are empirically justified. Also, Melis et al. (2018) established that several years of assumed progress in language modeling did not in fact improve upon a standard stacked LSTM model if the hyperparameters for all models were tuned properly.

Bayesian Deep Learning, which applies the ideas of Bayesian inference to deep networks is an active area of machine learning research. Some of the popular techniques of approximate inference in deep networks include variational inference (VI) (Graves, 2011), probabilistic backpropagation

[1] http://hunch.net/?p=22

(PBP) (Hernández-Lobato and Adams, 2015) for Bayesian neural nets , dropout as an interpretation of approximate Bayesian inference (Gal and Ghahramani, 2016), Deep Gaussian Processes (DGP) as a multi-layer generalization of Gaussian Processes (Bui et al., 2016), Bayesian neural networks using Variational Matrix Gaussian (VMG) posteriors (Louizos and Welling, 2016), neural (Sun et al., 2017) and variants of Stochastic Gradient Hamiltonian Monte Carlo (SGHMC) methods (Springenberg et al., 2016).

In order to compare and benchmark these methods, there should be a well-defined experimental setup. One of the most popular experiments to evaluate the performance of such models is a regression experiment on several curated UCI datasets (Hernández-Lobato and Adams, 2015). This experiment is appealing for Bayesian neural nets because it uses predictive log-likelihood in addition to RMSE as an evaluation metric. Predictive log-likelihood can be used to judge the quality of the uncertainty estimates produced by the model (Gal and Ghahramani, 2016) and therefore, is a good evaluation metric for Bayesian neural networks.

In this study, we observe that there is a discrepancy in the setup used for the UCI regression experiment in some recent works including VMG (Louizos and Welling, 2016), HS-BNN (Ghosh and Doshi-Velez, 2017), PBP-MV (Sun et al., 2017) and SGHMC (Springenberg et al., 2016), where they incorrectly compare and benchmark the performance of their models relative to baselines obtained under a different experimental setting. In order to gauge the impact of this erroneous comparison, we choose MC dropout (Gal and Ghahramani, 2016), one of the simplest techniques of approximate inference, and rerun the regression experiments using the setup used by the above mentioned works. The experimental results indicate that the networks with dropout inference when trained under the same conditions outperform VMG, HS-BNN and SGHMC and are a close second to PBP-MV which performs the best. Hence, in this work, we expose a flaw in the experimental method adopted by several works for the UCI regression experiment and note that some of the presumed state-of-the-art methods might actually not be so if properly evaluated. We also make an appeal to researchers to use fair experimental settings when evaluating and benchmarking their techniques.

## 2 Regression Experiments

For our experiment, we follow the setup proposed in Hernández-Lobato and Adams (2015). In particular we perform the non-linear regression experiments as they have been adopted by numerous subsequent works, Gal and Ghahramani (2016), Bui et al. (2016), Louizos and Welling (2016), Ghosh and Doshi-Velez (2017), Sun et al. (2017), Springenberg et al. (2016) etc., to evaluate their approximate inference techniques. We perform the experiments on all the datasets from Hernández-Lobato and Adams (2015) except the *YearPredictionMSD* dataset. The *YearPrediction-MSD* dataset is very large with 515,345 instances each of which has 90 dimensions. Hence, tuning network hyperparameters over this dataset requires an inordinate amount of time. For the other datasets, there are multiple train/test splits: 20 splits for all of them except *Protein Structure* for which there are 5 splits. The network architecture used has a single hidden layer with 50 hidden units for all the datasets except *Protein Structure* for which there are 100 hidden units.

There are two ways in which this experiment has been conducted in the past. In one of the settings, the networks are trained for a fixed number of iterations (specifically, 40 epochs), and the average running time of the networks are noted and compared. This setting was used by for example Hernández-Lobato and Adams (2015), Gal and Ghahramani (2016), and Bui et al. (2016). In this work, we refer to this as the *timed setting* of the experiment. Another variant of the experiment was used by Louizos and Welling (2016), Ghosh and Doshi-Velez (2017), Sun et al. (2017), Springenberg et al. (2016), and others; where the networks are trained to convergence with a higher number of training iterations. Finally, the test set RMSE and log-likelihood values are used as metrics for evaluation.

Given these two settings, it is quite natural to wonder if comparing models trained under the timed setting with those trained to convergence is fair. The idea is that the networks which have been trained for a fixed number of iterations might not have converged and will perform poorly compared to the ones which have been trained to convergence. In order to test this hypothesis, we use networks with MC dropout (Gal and Ghahramani, 2016), one of the de facto standard baselines for approximate inference. The purpose of the experiment is twofold. Firstly, we want to see if net-

Table 1: Regression experiment results: Average test performance using RMSE

| **Dataset** | **Dropout (Timed Setting)** | **Dropout (Convergence)** | **Dropout (Hyperparameter tuning)** | **VMG** | **HS-BNN** | **PBP-MV** |
|---|---|---|---|---|---|---|
| Boston Housing | $2.97 \pm 0.19$ | $2.83 \pm 0.17$ | $2.90 \pm 0.18$ | $\mathbf{2.70 \pm 0.13}$ | $3.32 \pm 0.66$ | $3.11 \pm 0.15$ |
| Concrete Strength | $5.23 \pm 0.12$ | $4.93 \pm 0.14$ | $\mathbf{4.82 \pm 0.16}$ | $4.89 \pm 0.12$ | $5.66 \pm 0.41$ | $5.08 \pm 0.14$ |
| Energy Efficiency | $1.66 \pm 0.04$ | $1.08 \pm 0.03$ | $0.54 \pm 0.06$ | $0.54 \pm 0.02$ | $1.99 \pm 0.34$ | $\mathbf{0.45 \pm 0.01}$ |
| Kin8nm | $0.10 \pm 0.00$ | $0.09 \pm 0.00$ | $0.08 \pm 0.00$ | $0.08 \pm 0.00$ | $0.08 \pm 0.00$ | $\mathbf{0.07 \pm 0.00}$ |
| Naval Propulsion | $0.01 \pm 0.00$ | $\mathbf{0.00 \pm 0.00}$ | $\mathbf{0.00 \pm 0.00}$ | $\mathbf{0.00 \pm 0.00}$ | $\mathbf{0.00 \pm 0.00}$ | $\mathbf{0.00 \pm 0.00}$ |
| Power Plant | $4.02 \pm 0.04$ | $4.00 \pm 0.04$ | $4.01 \pm 0.04$ | $4.04 \pm 0.04$ | $4.03 \pm 0.15$ | $\mathbf{3.91 \pm 0.04}$ |
| Protein Structure | $4.36 \pm 0.01$ | $4.27 \pm 0.01$ | $4.27 \pm 0.02$ | $4.13 \pm 0.02$ | $4.39 \pm 0.04$ | $\mathbf{3.94 \pm 0.02}$ |
| Wine Quality Red | $0.62 \pm 0.01$ | $\mathbf{0.61 \pm 0.01}$ | $0.62 \pm 0.01$ | $0.63 \pm 0.01$ | $0.63 \pm 0.04$ | $0.64 \pm 0.01$ |
| Yacht Hydrodynamics | $1.11 \pm 0.09$ | $0.70 \pm 0.05$ | $\mathbf{0.67 \pm 0.05}$ | $0.71 \pm 0.05$ | $1.58 \pm 0.23$ | $0.81 \pm 0.06$ |

Table 2: Regression experiment results: Average test performance using log likelihood

| **Dataset** | **Dropout (Timed Setting)** | **Dropout (Convergence)** | **Dropout (Hyperparameter tuning)** | **VMG** | **HS-BNN** | **PBP-MV** | **SGHMC (Tuned per dataset)** | **SGHMC (Scale Adapted)** |
|---|---|---|---|---|---|---|---|---|
| Boston Housing | $-2.46 \pm 0.06$ | $\mathbf{-2.40 \pm 0.04}$ | $\mathbf{-2.40 \pm 0.04}$ | $-2.46 \pm 0.09$ | $-2.54 \pm 0.15$ | $-2.54 \pm 0.08$ | $-2.49 \pm 0.15$ | $-2.54 \pm 0.04$ |
| Concrete Strength | $-3.04 \pm 0.02$ | $-2.97 \pm 0.02$ | $\mathbf{-2.93 \pm 0.02}$ | $-3.01 \pm 0.03$ | $-3.09 \pm 0.06$ | $-3.04 \pm 0.03$ | $-4.17 \pm 0.72$ | $-3.38 \pm 0.24$ |
| Energy Efficiency | $-1.99 \pm 0.02$ | $-1.72 \pm 0.01$ | $-1.21 \pm 0.01$ | $-1.06 \pm 0.03$ | $-2.66 \pm 0.13$ | $\mathbf{-1.01 \pm 0.01}$ | $--$ | $--$ |
| Kin8nm | $0.95 \pm 0.01$ | $0.97 \pm 0.00$ | $1.14 \pm 0.01$ | $1.10 \pm 0.01$ | $1.12 \pm 0.03$ | $\mathbf{1.28 \pm 0.01}$ | $--$ | $--$ |
| Naval Propulsion | $3.80 \pm 0.01$ | $3.91 \pm 0.01$ | $4.45 \pm 0.00$ | $2.46 \pm 0.00$ | $\mathbf{5.52 \pm 0.10}$ | $4.85 \pm 0.06$ | $--$ | $--$ |
| Power Plant | $-2.80 \pm 0.01$ | $-2.79 \pm 0.01$ | $-2.80 \pm 0.01$ | $-2.82 \pm 0.01$ | $-2.81 \pm 0.03$ | $\mathbf{-2.78 \pm 0.01}$ | $--$ | $--$ |
| Protein Structure | $-2.89 \pm 0.00$ | $-2.87 \pm 0.00$ | $-2.87 \pm 0.00$ | $-2.84 \pm 0.00$ | $-2.89 \pm 0.00$ | $\mathbf{-2.77 \pm 0.01}$ | $--$ | $--$ |
| Wine Quality Red | $-0.93 \pm 0.01$ | $\mathbf{-0.92 \pm 0.01}$ | $-0.93 \pm 0.01$ | $-0.95 \pm 0.01$ | $-0.95 \pm 0.05$ | $-0.97 \pm 0.01$ | $-1.29 \pm 0.28$ | $-1.04 \pm 0.17$ |
| Yacht Hydrodynamics | $-1.55 \pm 0.03$ | $-1.38 \pm 0.01$ | $-1.25 \pm 0.01$ | $-1.30 \pm 0.02$ | $-2.33 \pm 0.01$ | $-1.64 \pm 0.02$ | $-1.75 \pm 0.19$ | $\mathbf{-1.10 \pm 0.08}$ |

works with dropout inference, when trained to convergence perform better than the ones which were trained under the timed setting. Secondly, we also want to compare these new results with other works where models trained to convergence have been compared (either directly or indirectly) with baselines obtained under the timed setting. In particular, we compare with the following works:

1. VMG (Louizos and Welling, 2016) where models trained to convergence have been benchmarked against MC dropout baselines from the timed setting.

2. Bayesian networks with horseshoe priors (HS-BNN) (Ghosh and Doshi-Velez, 2017) and probabilistic backpropagation with the Matrix Variate Gaussian (MVG) distribution (PBP-MV) (Sun et al., 2017) where the models were only benchmarked against VMG.

3. Stochastic Gradient Hamiltonian Monte Carlo methods (SGHMC) (Springenberg et al., 2016) where the results have been compared with Probabilistic Backpropagation (PBP) (Hernández-Lobato and Adams, 2015) and Variational Inference (VI) (Graves, 2011) baselines obtained under the timed setting.

In order to train our networks, we need to set two hyperparameters: i) the model precision parameter $\tau$ to evaluate the log-likelihood and ii) the dropout rate $d$. We perform the following two variations of the regression experiment:

1. **Convergence**: The networks are trained to convergence for 4,000 epochs and the hyperparameter values are obtained by Bayesian Optimization (BO) (as described in Gal and Ghahramani (2016)).

2. **Hyperparameter tuning**: The networks are trained to convergence (for 4,000 epochs) and the optimal hyperparameter values are obtained by performing grid search over a range of $(\tau, d)$ pairs and choosing the best pair based on performance over a validation set. The validation set is created by randomly choosing 20% of the data points in the training set.

The RMSE and log likelihood values obtained from the above experiments are compared in Table 1 and 2 respectively. It should be noted here that the RMSE and log likelihood values of VMG, HS-BNN, PBP-MV and SGHMC methods have been taken from their respective papers. The experimental results indicate that the *Convergence* and *Hyperparameter tuning* baseline experiments show a significant improvement in performance compared to the results in the *timed setting*. We also observe that the baseline outperform the other methods on the *Concrete Strength*, *Naval Propulsion Plants*, *Wine Quality Red* and the *Yacht Hydrodynamics* datasets in terms of RMSE. With respect to log likelihood, the baseline performs the best on the *Boston Housing*, *Concrete Strength*, and *Wine Quality Red* datasets. Finally, we also observe that on the other datasets the baseline is quite competitive second only to PBP-MV (Sun et al., 2017).

## 3 Conclusion

The RMSE and log-likelihood values obtained from the *Convergence* and *Hyperparameter tuning* experiments (as given in Tables 1 and 2) provide significantly better results for MC dropout. Therefore, we can conclude that the comparisons made by some of the works with baselines obtained from the timed setting of the experiment are indeed unrepresentative. It is important to note that when benchmarking a method, its performance should be evaluated using an experimental setup which is identical to the ones used to evaluate its peers or competitors. Hence, in this work, we expose a flaw in research methodology and encourage researchers to use strong and fairly computed baselines to benchmark their results. Furthermore, in Tables 1 and 2 we lay out stronger baselines for MC dropout inference which can be used for future comparisons. The source code for our experiments can be found at: `https://github.com/yaringal/DropoutUncertaintyExps`

## References

Bui, T., Hernández-Lobato, D., Hernandez-Lobato, J., Li, Y., and Turner, R. (2016). Deep gaussian processes for regression using approximate expectation propagation. In *International Conference on Machine Learning*, pages 1472–1481.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML-16)*.

Ghosh, S. and Doshi-Velez, F. (2017). Model selection in bayesian neural networks via horseshoe priors. *arXiv preprint arXiv:1705.10388*.

Graves, A. (2011). Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., and Meger, D. (2017). Deep reinforcement learning that matters. *arXiv preprint arXiv:1709.06560*.

Hernández-Lobato, J. M. and Adams, R. (2015). Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869.

Louizos, C. and Welling, M. (2016). Structured and efficient variational deep learning with matrix gaussian posteriors. In *International Conference on Machine Learning*, pages 1708–1716.

Melis, G., Dyer, C., and Blunsom, P. (2018). On the state of the art of evaluation in neural language models. In *International Conference on Learning Representations*.

Springenberg, J. T., Klein, A., Falkner, S., and Hutter, F. (2016). Bayesian optimization with robust bayesian neural networks. In *Advances in Neural Information Processing Systems*, pages 4134–4142.

Sun, S., Chen, C., and Carin, L. (2017). Learning structured weight uncertainty in bayesian neural networks. In *Artificial Intelligence and Statistics*, pages 1283–1292.