# PROBABILISTIC AI - DT8122- PROJECT ASSIGNMENT - UNCERTAINTY QUANTIFICATION

**Jean-Michel Amath Sarr**
Department of Mathematics and Computer Science
Faculty of Science and Technology
Cheikh Anta Diop University
BP 5005, Dakar, Fann, Senegal
jmas902010@gmail.com

July 12, 2019

**ABSTRACT**

We have implemented two generative models: Dropout as a Bayesian approximation and Weight uncertainty in neural network. Our results suggest that using the Expected Lower Bound (ELBO) as the optimization objective during training is not relevant when having in mind general uncertainty metrics such as the Prediction Interval Coverage Probability (PICP), and the Mean Prediction Interval Width (MPIW). Because of the lack of constraint on the PICP. We think that a better strategy would be to craft a unifying metric as it has been proposed in the literature.

***K*eywords** Probabilistic modeling · Deep learning · Uncertainty quantification

## 1 Model Choice

We choose to implement Dropout as a Bayesian approximation (Dropout) [1] and Weight uncertainty in neural network (Bayes by Backprop or BbB for short) [3].

## 2 Inference Methods

### 2.1 First Implementation

We implemented a deep neural network with 3 layers and 100 hidden units. We trained BbB with pyro with 100 samples for the Monte Carlo (MC) gradient estimate, whereas we trained Dropout with pytorch and with a single MC estimate. The following is a quick recap of the model implemented.

#### 2.1.1 Bayes by Backprop

**Prior**
The prior is a mixture of multivariate Gaussian for every parameter in the network. We fixed $\pi = 0.5$ , $\sigma_1 = 1$ and finally $\sigma_2 = e^{-6}$. The prior according to the paper resemble a spike-and-lab prior, where the prior parameters are shared among all the weights.

$$w \sim \pi\mathcal{N}(0, \sigma_1) + (1 - \pi)\mathcal{N}(0, \sigma_2)$$

**Variational Approximation**
Each weight is sampled from a normal distribution with learnable parameters $\mu_w$ and $\sigma_w$ and having $\sigma_w > 0$

$$w|\mu, \sigma \sim \mathcal{N}(\mu_w, \sigma_w)$$

We optimized the Expected Lower Bond (ELBO) with the Adam optimizer and a learning rate of $0.005$. Each layer used the non-linear activation Scaled Exponential Linear Unit (SELU) because instead of RELU it does not stuck the gradient lower than 0 to 0.

#### 2.1.2 Dropout

Using the bayesian interpretation of dropout. The prior is a multivariate standard normal for every weight in the network, we did initialized the bias to zero. The posterior consist in sampling from a distribution over matrices where each columns are set to zero following a Bernoulli random variable .

$$z_i \sim p_i,\ w = \mathrm{M} \cdot \mathrm{diag}[z_i]$$

We trained the model using the Log ELBO treating the precision $\tau$ and the prior length scale $l$ in a single constant $\lambda$. In other words we used the following $\frac{l^2}{2\tau} = \lambda$. So our stochastic objective was

$$\mathcal{L} \propto -\frac{1}{N}\sum_{i=1}^{N} ||y_i - \hat{y_i}||_2^2 - \sum_{j=1}^{L} p_i \lambda ||M_j||_2^2 - \lambda ||m_j||_2^2$$

Furthermore, we systematically used Bayesian Optimization (BO) with each dataset to find the best learning rate and the best $\lambda$ for the model. Otherwise the training was too sensitive to the learning rate, also it follows the scheme in the original publication [1].

### 2.2 Second Implementation

#### 2.2.1 BbB

For both model we decided to use BO to select the hidden dimension of the neural network and the learning rate for BbB, as well as lambda for Dropout. The first surprise was that hidden unit output was often around 10 units with comparable for Dropout (see table 2). However it did not work out as well with BbB.

## 3 Results

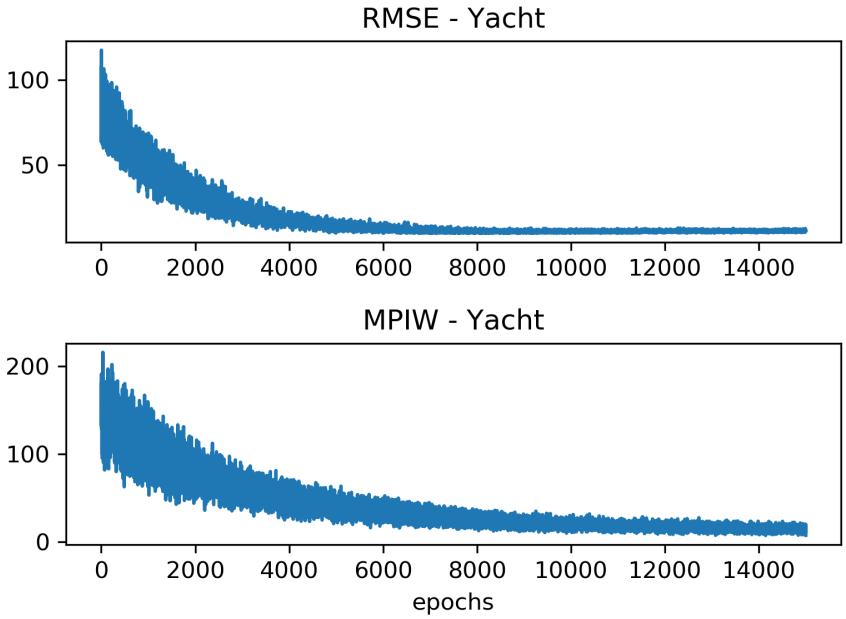
**First Implementation**: We did not try every trick possible to have good results, and as we can see the models behave poorly with some datasets, for instance, results on protein structure, power plant and year prediction msd datasets with respect to Root Mean Square Error (RMSE), Prediction Interval Coverage Probability, and Mean Prediction Interval Width were poor. We kept the hyperparameters fixed for all datasets. For example we kept $\pi = 0.5$, $\sigma_1 = 1$ and $\sigma_2 = e^{-6}$ for BbB, and we kept $p_i = \text{Bernoulli}(0.8)$ for Dropout. It means that we started training our models with the same prior for all datasets. We think that it is not a reasonable assumption.

**Second Implementation**: To improve on the previous result while keeping the same priors, we used BO with both models adding the hidden dimension in the loop. We were surprised by the dimension outputted by the algorithm. In most case instead of training with 100 hidden units, it went down to 10 parameters (see table 2). We still had poor results with power plant, protein structure and year prediction msd overall. This strategy seemed not to be effective with respect to BbB, where it worked out well for the energy efficiency and the naval propulsion dataset alone. Concerning Dropout this second approach may have been more successful: indeed, we got better results with the boston housing, protein and wine quality datasets. And similar results for energy efficiency and yacht hydrodynamics datasets. With less parameter overall.

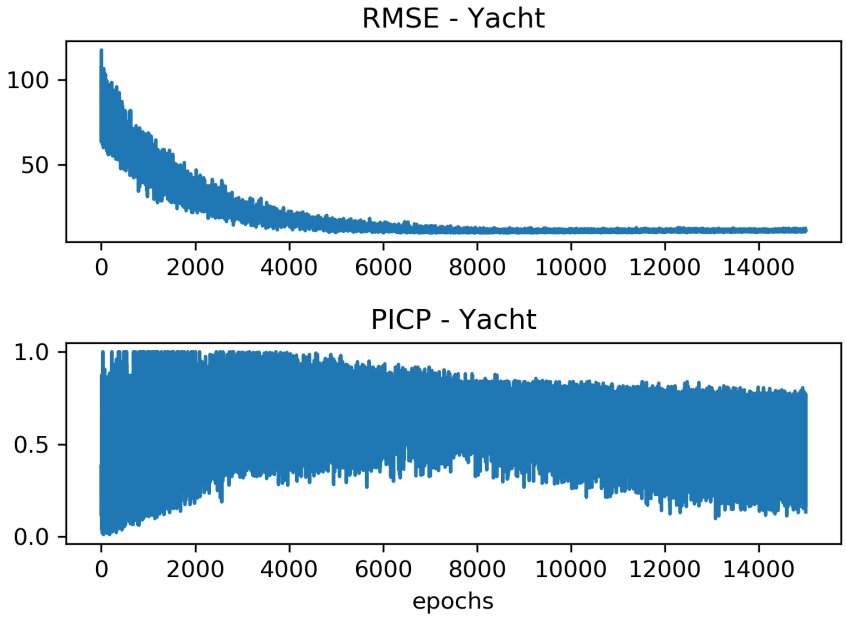
**General Remarks**: We find that using only the ELBO is not enough if we want to converge toward good performance in terms of PICP, MPIW and RMSE. In fact, during training every objective decrease, however, we would like to constraint the PICP in a way that in average it is still higher than some threshold (see figure 1 and 2). In practice, it is inconvenient to just look at the ELBO decreasing, because it does not give ints about the PICP evolution The right number of epochs becomes a hyperparameter to tune, and this process is very time-consuming. Also computing the PICP and the MPIW during training may cause runtime errors for memory reason. The idea in [2] to do inference as a constraint optimization problem makes sense to focus only on one metric.

## 4 Findings and Discussion

While using pyro was really more convenient to train BbB, it was much more straightforward to use pytorch for Dropout. Essentially because there are no pyro primitives to express a mixture of Gaussian with respect to matrices in pyro

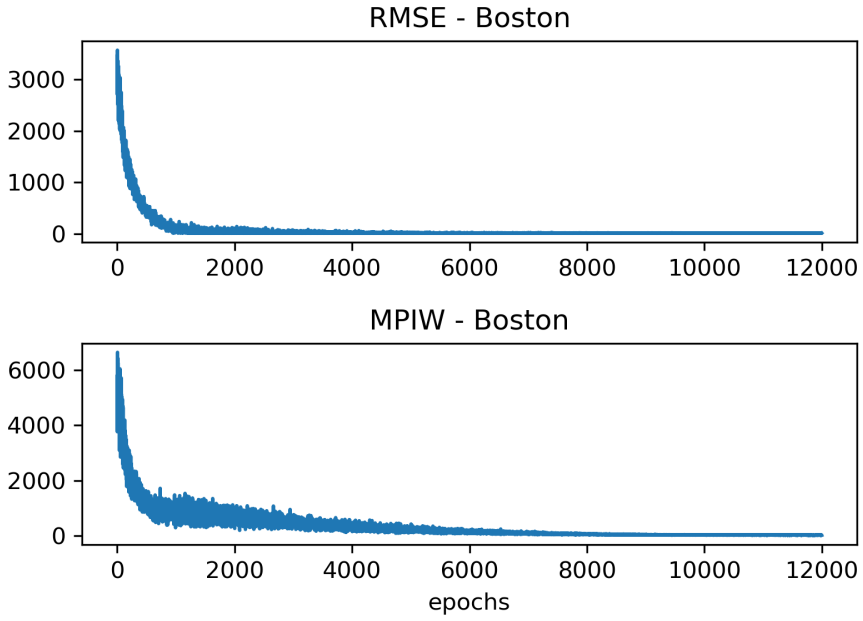



(a)

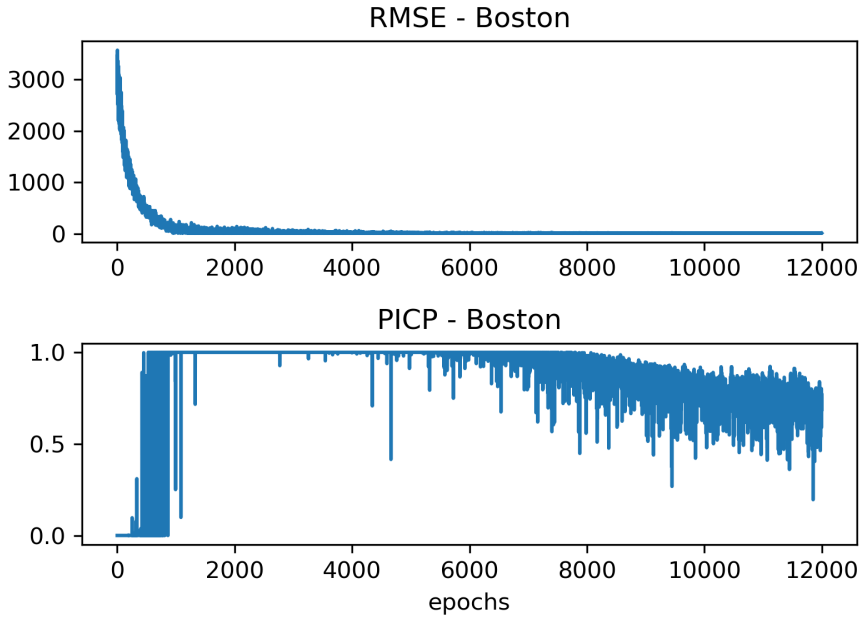



(b)

Figure 1: Dropout: (a) RMSE and MPIW during training decrease in a very similar way. (b) RMSE and PICP however do not decrease similarly. It seems that there is an optimal number of epochs to run the inference algorithm, here approximately around 8000 epochs. After this point the performance degrades it self.




(a)




(b)

Figure 2: Dropout: (a) We observe the same phenomenon as in Figure 1, i.e the fact that RMSE and MPIW decrease alike, (b) it seems also that the best epoch to stop training is around 7000.

Table 1: First Implementation

| | RMSE | | PICP | | MPIW | | Epochs | |
|---|---|---|---|---|---|---|---|---|
| Dataset | MC Dropout | BbB | MC Dropout | BbB | MC Dropout | BbB | MC Dropout | BbB |
| Concrete Strength | **10.06** | 12.57 | **0.97** | 0.58 | **13.9** | 26.79 | 35,000 | 35,000 |
| Boston Housing | 4.73 | **3.82** | **0.98** | 0.92 | 45.69 | **19.45** | 25,000 | 35,000 |
| Energy Efficiency | **5.5** | 9.12 | 0.59 | **0.68** | **17.96** | 21.45 | 30,000 | 35,000 |
| Kin8nm | **0.17** | 0.22 | 0.33 | **1.0** | **0.2** | 4.0 | 15,000 | 35,000 |
| Naval Propulsion | 32.3 | **1.55** | 1.0 | 1.0 | 300.4 | **102.82** | 25,000 | 25,000 |
| Power Plant | 21.11 | **4.35** | 1.0 | 1.0 | 282.6 | **31.55** | 25,000 | 35,000 |
| Protein Structure | 19413.6 | **15679** | 1.0 | 1.0 | **249568** | 1263488.8 | 25,000 | 35,000 |
| Wine Quality Red | 0.68 | 0.68 | 1.0 | 1.0 | 8.27 | **0.56** | 25,000 | 35,000 |
| Yacht Hydrodynamics | 1.96 | 1.96 | **1.0** | 0.8 | 16.74 | **7.57** | 25,000 | 35,000 |
| Year Prediction MSD | **9.26** | 15.96 | 1.0 | 1.0 | 562.8 | **311.3** | 35,000 | 35,000 |

Table 2: Second Implementation

| | RMSE | | PICP | | MPIW | | Hidden dimension | |
|---|---|---|---|---|---|---|---|---|
| Dataset | MC Dropout | BbB | MC Dropout | BbB | MC Dropout | BbB | MC Dropout | BbB |
| Concrete Strength | **13.14** | 124.9 | 0.90 | **1** | **53.15** | 9479.0 | 12 | 42 |
| Boston Housing | **4.79** | 5.56 | **0.92** | **0.62** | 26.36 | 12.19 | 10 | 10 |
| Energy Efficiency | 9.09 | **3.11** | **0.75** | 0.74 | 30.56 | **8.67** | 10 | 35 |
| Kin8nm | **0.16** | 0.38 | 0.54 | 1 | **0.32** | 5.50 | 10 | 10 |
| Naval Propulsion | 18.46 | **0.03** | 1.0 | 1.0 | 321 | **10.31** | 11 | 12 |
| Power Plant | **19.0** | 21.33 | 1.0 | 1.0 | **146.3** | 11287.9 | 35 | 40 |
| Protein Structure | **6.32** | 186104 | 1.0 | 1.0 | **133.85** | 3503597 | 12 | 12 |
| Wine Quality Red | **0.66** | 12.8 | 0.83 | **1** | **1.77** | 456.9 | 14 | 37 |
| Yacht Hydrodynamics | **2.22** | 7.5 | **1** | 0.96 | **17.8** | 33.38 | 10 | 11 |
| Year Prediction MSD | **36.05** | 1912 | **1.0** | 0 | 837.87 | 0 | 36 | 47 |

(see section 3.3 in [1] appendix). Because the variational inference mechanism is heavily based on the sampling pyro primitive, the best solution would have been to write a custom distribution. We note that training Dropout was much faster, mainly because we did use one sample to do the Monte Carlo approximation. We noticed a limitation when training both models, it is related to the choice of our minimization objective. When you increase the number of epochs RMSE, MIPW and PICP are going to decrease but we need PICP to somewhat be constraint. Because the current variational inference training does not take it into account. At the end, we will still have a measure of uncertainty, but it is not enough if we want good prediction intervals.

We report that this trade-off was much more sensitive in training the Dropout model. Overall training the Dropout model relies a lot more on tricks that we finally automate with BO than BbB that is more reliable once implemented. Using BO however did not yield good results with BbB.

## References

[1] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR.

[2] Tim Pearce, Alexandra Brintrup, Mohamed Zaki, and Andy Neely. High-quality prediction intervals for deep learning: A distribution-free, ensembled approach. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4075–4084, Stockholmsmässan, Stockholm Sweden, 10–15 Jul 2018. PMLR.

[3] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1613–1622, Lille, France, 07–09 Jul 2015. PMLR.

[4] Christos Louizos and Max Welling. Multiplicative normalizing flows for variational Bayesian neural networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 2218–2227, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[5] David Krueger, Chin-Wei Huang, Riashat Islam, Ryan Turner, Alexandre Lacoste, and Aaron Courville. Bayesian hypernetworks. *arXiv preprint arXiv:1710.04759*, 2017.