Supplementary Material for Learnable Bernoulli Dropout for Bayesian Deep Learning

Shahin Boluki[†] **Randy Ardywibowo**[†] **Siamak Zamani Dadaneh**[†]
Mingyuan Zhou[‡] **Xiaoning Qian**[†]
[†] Texas A&M University [‡]The University of Texas at Austin

Abstract

This supplementary file contains the pseudo-code for our learnable Bernoulli dropout (LBD), run-time comparison with other existing dropout models, as well as additional experimental results. The code for the experiments can be found at the following **link**.

1 Pseudo-code of LBD training

Algorithm 1 outlines the major steps for learning dropout rates and other network parameters in our LBD framework.

Result: $\boldsymbol{\theta}$ that minimizes $\mathbb{E}_{\boldsymbol{z}}\big[\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{z} \,|\, \mathcal{D})\big]$
where $\boldsymbol{\theta} = \{\boldsymbol{\psi}, \boldsymbol{\alpha}\}$ consists of dropout mask parameters $\boldsymbol{\alpha}$, and all other parameters $\boldsymbol{\psi}$;
Initialize $\boldsymbol{\theta}$ randomly;
while *not converged* **do**
  Sample a mini-batch of size M from the data;
  Sample $u_{ijk} \sim \text{Unif}_{[0,1]}$ for $i = 1, ..., M$, $j = 1, ..., L$, $k = 1, ..., K_{j-1}$;
  $\boldsymbol{g}_{\boldsymbol{\psi}} = \nabla_{\boldsymbol{\psi}} \mathcal{L}(\boldsymbol{\theta}, 1_{[\boldsymbol{u} < \sigma(\boldsymbol{\alpha})]})$;
  $\boldsymbol{g}_{\boldsymbol{\alpha}} = \big(\mathcal{L}(\boldsymbol{\theta}, 1_{[\boldsymbol{u} > \sigma(-\boldsymbol{\alpha})]}) - \mathcal{L}(\boldsymbol{\theta}, 1_{[\boldsymbol{u} < \sigma(\boldsymbol{\alpha})]})\big)(\boldsymbol{u} - \mathbf{1/2})$;
  $\boldsymbol{\psi} = \boldsymbol{\psi} - r_{\boldsymbol{\psi}} \boldsymbol{g}_{\boldsymbol{\psi}}$;
  $\boldsymbol{\alpha} = \boldsymbol{\alpha} - r_{\boldsymbol{\alpha}} \boldsymbol{g}_{\boldsymbol{\alpha}}$
end

Algorithm 1: Parameter estimation in Learnable Bernoulli Dropout scheme.

2 Additional toy example results

The bias and mean squared error of the estimates for α_2 from LBD and Concrete calculated by 200 Monte Carlo samples for the toy example in Section 3.1 are shown in Figure 1.

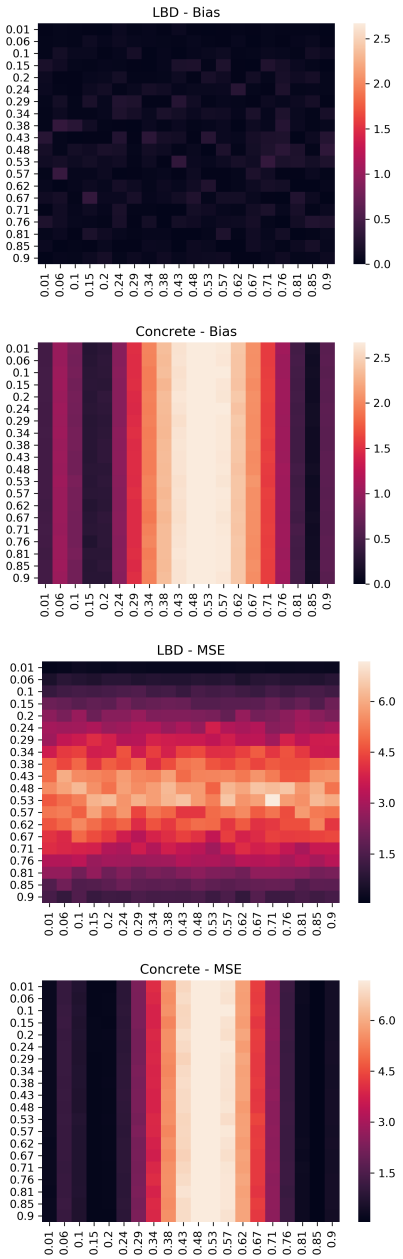

Figure 1: Comparison of gradient estimates for dropout parameters in the toy example.

Proceedings of the 23[rd]International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

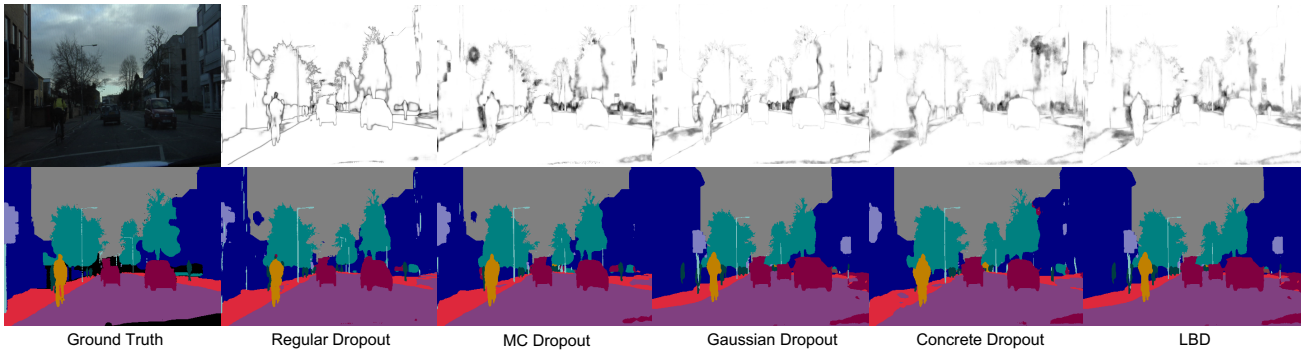

Figure 2: Example of image semantic segmentation results for FC-DenseNet-103 on the CamVid dataset

Table 1: Root Mean Square Error (RMSE) of different methods trained on various UCI datasets. Standard deviations of the RMSE are shown in parentheses

Dataset	N	d	**Average RMSE (Standard Deviation)**			
			MC	Gaussian	Concrete	LBD
Boston Housing	506	13	4.26 (1.21)	3.55 (1.06)	3.60 (1.27)	**3.35 (0.98)**
Concrete Compression Strength	1,030	8	7.47 (0.71)	4.91 (0.63)	4.94 (0.51)	**4.78 (0.46)**
Energy Efficiency	768	8	2.24 (0.21)	**0.87 (0.13)**	1.01 (0.15)	1.07 (0.17)
Naval Propulsion	11,934	16	**0.01 (0.00)**	**0.01 (0.00)**	**0.01 (0.00)**	**0.01 (0.00)**
Combined Cycle Power Plant	9,568	4	4.79 (0.22)	4.06 (0.16)	**4.05 (0.19)**	4.06 (0.13)
Protein Structure	45,730	9	4.83 (0.05)	4.09 (0.06)	4.06 (0.04)	**4.05 (0.06)**
Wine Quality Red	1,599	11	**0.63 (0.04)**	0.67 (0.04)	**0.63 (0.05)**	0.64 (0.05)
Yacht Hydrodynamics	308	6	4.87 (1.32)	1.99 (0.68)	1.78 (1.11)	**1.4 (0.56)**

3 Additional semantic segmentation results

Figure 2 provides the visualization of the example outputs from FC-DenseNet-103 with different dropout schemes for semantic segmentation on the CamVid dataset (Section 3.1.2). The bottom panels depict the segmentation results while the top panels visualize the uncertainty (predictive entropy).

4 Run-time comparison

Here, we discuss how much run-time change we observed empirically when using LBD compared with other dropout variants.

For image classification using VGG19 on the CIFAR-10 dataset (Section 3.2.1), all dropout variants had almost the same training run time.

For the semantic segmentation with FC-DenseNet-103 on CamVid dataset (Section 3.2.2), LBD, Concrete and Gaussian dropouts had the same training run time, with a 4.5% increase in run time compared with the regular dropout.

In the collaborative filtering experiments (Section 3.3), the SIVAE with LBD (SIVAE+LBDrop), SIVAE with Concrete dropout (SIVAE+CDrop), and SIVAE with Gaussian dropout (SIVAE+GDrop) all had the same training run time.

5 Regression on UCI Datasets

In addition to the reported results in the tasks described in the main text, we also have tested different dropout schemes for regression on various UCI datasets [1]. In this setting, following [2] we implemented our LBD on a fully connected network with 2 hidden layers, with 50 hidden units each. We compared our method with other dropout methods, such as Concrete dropout, Gaussian dropout, and MC dropout, on the same architecture. We randomly split each dataset into 90% training and 10% testing data for 20 times (5 times for Naval Propulsion and Protein Structure). In each split, we train the fully connected architecture using the different dropout methods for 65 epochs using a single sample for each batch. The RMSE of each method is shown in Table 1. As we can see, all the methods that optimize the dropout parameters perform very similarly. The improvement of one method over the others does not appear to be statistically significant.

References

[1] Dheeru Dua and Casey Graff. UCI machine learning repository, 2017.

[2] José Miguel Hernández-Lobato and Ryan Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, pages 1861–1869, 2015.