BMA under Covariate Shift

Timofey Chernikov

MIPT, 2023

November 21, 2023

Introduction

Main idea

Bayesian neural networks (BNNs) with high-fidelity approximate inference via full-batch Hamiltonian Monte Carlo achieve poor generalization under covariate shift, even underperforming classical estimation. We explain this surprising result. We additionally show why the same issue does not affect many approximate inference procedures, or classical maxi- mum a-posteriori (MAP) training. Finally, we propose novel priors that improve the robustness of BNNs to many sources of covariate shift.

Introduction

Bayesian neural networks

A Bayesian neural network model is specified by the prior distribution $p(w)$ over the weights w of the model, and the likelihood function $p(y|x, w)$, where x represents the input features and y represents the target value. Following Bayes' rule, the posterior distribution over the parameters w is given by

$$p(w|D) = \frac{p(D|w)p(w)}{\int p(D|w')p(w')dw'}$$

$$p(y|x) = \int p(y|x, w)p(w|D)dw$$

Introduction

Maximum a-posteriori (MAP) estimation

In contrast with Bayesian model averaging, a MAP estimator uses the single setting of weights (hypothesis) that maximizes the posterior density $w_{MAP} = argmaxp(w|D) = argmax(logp(D|w) + logp(w))$, where the log prior can be viewed as a regularizer. In other words, the MAP estimator is a classical neural network approach.

Covariate shift.

In this paper, we focus on the covariate shift setting. We assume the training dataset D_{train} consists of i.i.d. samples from the distribution $p_{train}(x, y) = p_{train}(x)p(y|x)$. However, the test data may come from a different distribution $p_{test}(x, y) = p_{test}(x)p(y|x)$. For concreteness, we assume the conditional distribution $p(y|x)$ remains unchanged, but the marginal distribution of the input features $p_{test}(x)$ differs from $p_{train}(x)$.

Comparison

Methods

We evaluate BNNs against two deterministic baselines: a MAP solution approximated with stochastic gradient descent (SGD) with momentum and a deep ensemble of 10 independently trained MAP solutions. For BNNs, we provide the results using a Gaussian prior and a more heavy-tailed Laplace prior. Izmailov et al. conjectured that cold posteriors can improve the robustness of BNNs under covariate shift; to test this hypothesis, we provide results for BNNs with a Gaussian prior and cold posteriors at temperature 10^{-2}. For all BNN models, we run a single chain of HMC for 100 iterations discarding the first 10 iterations as burn-in, following Izmailov et al. We provide additional experimental details in Appendix A.

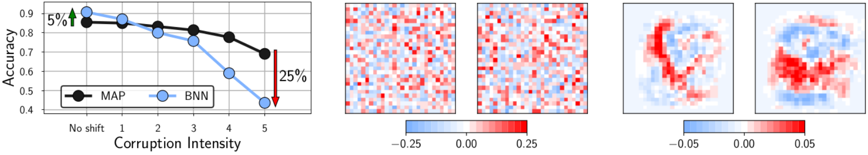
Methods

Neural network architectures

On both the CIFAR-10 and MNIST datasets we use a small convolutional network (CNN) inspired by LeNet-5, with 2 convolutional layers followed by 3 fully-connected layers. On MNIST we additionally consider a fully-connected neural network (MLP) with 2 hidden layers of 256 neurons each. We note that high-fidelity posterior sampling with HMC is extremely computationally intensive. Even on the small architectures that we consider, the experiments take multiple hours on 8 NVIDIA Tesla V-100 GPUs or 8-core TPU-V3 devices.

Test data corruption

We start by considering the scenario where the test data is corrupted by some type of noise. In this case, there is no semantic distribution shift: the test data is collected in the same way as the train data, but then corrupted by a generic transformation.



(a) ResNet-20, CIFAR-10-C (b) BNN weights (c) MAP weights

Figure 1: **Bayesian neural networks under covariate shift**. **(a)**: Performance of a ResNet-20 on the *pixelate* corruption in CIFAR-10-C. For the highest degree of corruption, a Bayesian model average underperforms a MAP solution by 25% (44% against 69%) accuracy. See Izmailov et al. [28] for details. **(b)**: Visualization of the weights in the first layer of a Bayesian fully-connected network on MNIST sampled via HMC. **(c)**: The corresponding MAP weights. We visualize the weights connecting the input pixels to a neuron in the hidden layer as a 28×28 image, where each weight is shown in the location of the input pixel it interacts with.

Domain shift

Next, we consider a different type of covariate shift where the test data and train data come from different, but semantically related distributions. First, we apply our CNN and MLP MNIST models to the SVHN test set. The MNIST-to- SVHN domain shift task is a common benchmark for unsupervised domain adaptation.

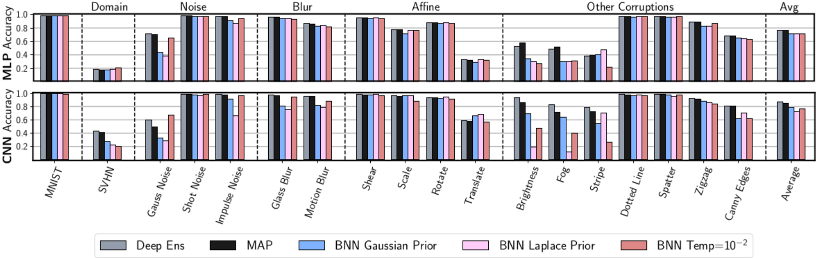


Figure 2: **Robustness on MNIST.** Accuracy for deep ensembles, MAP and Bayesian neural networks trained on MNIST under covariate shift. **Top**: Fully-connected network; **bottom**: Convolutional neural network. While on the original MNIST test set BNNs provide competitive performance, they underperform deep ensembles on most of the corruptions. With the CNN architecture, all BNN variants lose to MAP when evaluated on SVHN by almost 20%.

Domain shift

Next, we apply our CIFAR-10 CNN model to the STL-10 dataset. Both datasets contain natural images with 9 shared classes between the two datasets. We report the accuracy of the CIFAR-10 models on these 9 shared classes in STL-10.

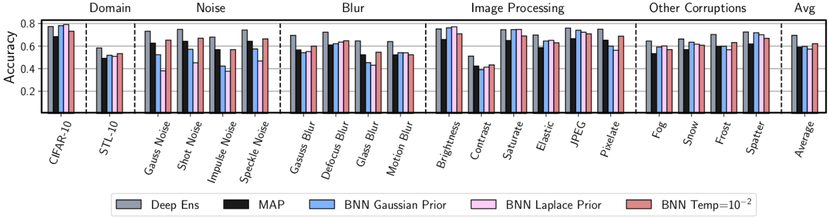
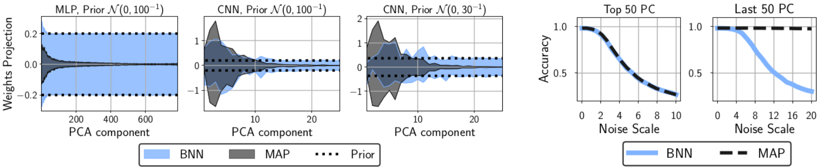


Figure 3: **Robustness on CIFAR-10.** Accuracy for deep ensembles, MAP and Bayesian neural networks using a CNN architecture trained on CIFAR-10 under covariate shift. For the corruptions from CIFAR-10-C, we report results for corruption intensity 4. While the BNNs with both Laplace and Gaussian priors outperform deep ensembles on the in-distribution accuracy, they underperform even a single MAP solution on most corruptions.

Understanding BNNs under covariate shift

We identify the linear dependencies in the input features as one of the key issues undermining the robustness of BNNs.



(a) Weight projections

(b) Robustness along PCs

Figure 4: **Bayesian inference samples weights along low-variance principal components from the prior, while MAP sets these weights to zero. (a)**: The distribution (mean ± 2 std) of projections of the weights of the first layer on the directions corresponding to the PCA components of the data for BNN samples and MAP solution using MLP and CNN architectures with different prior scales. In each case, MAP sets the weights along low-variance components to zero, while BNN samples them from the prior. **(b)**: Accuracy of BNN and MAP solutions on the MNIST test set with Gaussian noise applied along the 50 highest and 50 lowest variance PCA components of the train data (left and right respectively). MAP is very robust to noise along low-variance PCA directions, while BMA is not; the two methods are similarly robust along the highest-variance PCA components.

Data empirical covariance prior

Assuming the input features are all preprocessed to be zero-mean, we have $\Sigma = \frac{1}{n-1}\sum x_i x_i^T$. For fully-connected networks, we propose to use the *EmpCov* prior $p(w) = N(0, \alpha\Sigma + \varepsilon I)$ on the weights w^1 of the first layer of the network. The hyperparameter $\alpha > 0$ determines the scale of the prior.

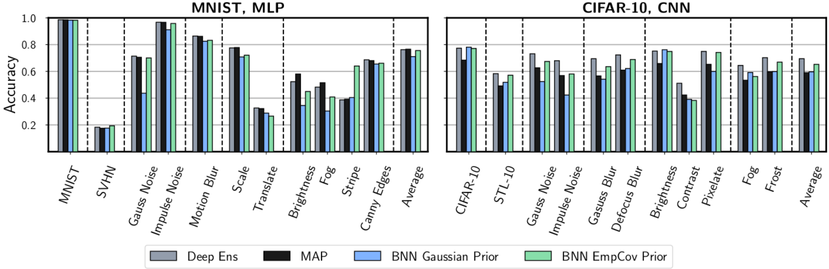


Figure 5: **EmpCov prior improves robustness.** Test accuracy under covariate shift for deep ensembles, MAP optimization with SGD, and BNN with Gaussian and *EmpCov* priors. **Left**: MLP architecture trained on MNIST. **Right**: CNN architecture trained on CIFAR-10. The *EmpCov* prior provides consistent improvement over the standard Gaussian prior. The improvement is particularly noticeable on the *noise* corruptions and domain shift experiments (SVHN, STL-10).

Literature

1. **Main article** Dangers of Bayesian Model Averaging under Covariate Shift