Being a Bit Frequentist Improves Bayesian Neural Networks

Agustinus Kristiadi
University of Tübingen
agustinus.kristiadi@uni-tuebingen.de

Matthias Hein
University of Tübingen
matthias.hein@uni-tuebingen.de

Philipp Hennig
University of Tübingen and MPI for Intelligent Systems, Tübingen
philipp.hennig@uni-tuebingen.de

Abstract

Despite their compelling theoretical properties, Bayesian neural networks (BNNs) tend to perform worse than frequentist methods in classification-based uncertainty quantification (UQ) tasks such as out-of-distribution (OOD) detection. In this paper, based on empirical findings in prior works, we hypothesize that this issue is because even recent Bayesian methods have never considered OOD data in their training processes, even though this "OOD training" technique is an integral part of state-of-the-art frequentist UQ methods. To validate this, we treat OOD data as a first-class citizen in BNN training by exploring several ways of incorporating OOD data in Bayesian inference. We show in experiments that OOD-trained BNNs are competitive to, if not better than recent frequentist baselines. This work thus provides strong baselines for future work in Bayesian deep learning.

1 Introduction

Uncertainty quantification (UQ) allows learning systems to "know when they do not know". Both the Bayesian and frequentist deep learning communities address similar UQ functionality (in particular out-of-distribution (OOD) detection), but it appears that even recent Bayesian neural networks [BNNs, 1–3, etc.] tend to underperform compared to the state-of-the-art frequentist UQ methods [4–8, etc.]. Figure 1 shows this observation in a standard benchmarks for OOD detection: Outlier Exposure [4], a popular frequentist method, performs much better than BNNs and even Deep Ensemble [9], which has been considered as a strong baseline in Bayesian deep learning [10].

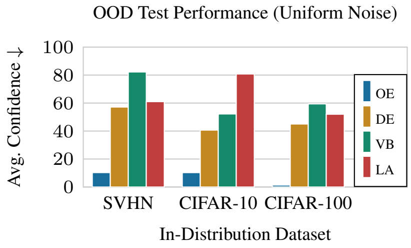

Figure 1: Avg. confidence on uniform OOD test data (lower is better). All methods have similar accuracy and confidence on the in-distribution test sets.

This paper is thus dedicated to answer the question of "how can we bring the performance of BNNs on par with that of recent frequentist UQ methods?" Our working hypothesis is that the disparity between them is *not* due to some fundamental advantage of the frequentist viewpoint, but to the more mundane practical fact that recent frequentist UQ methods leverage OOD data in their training process, via the so-called "OOD training" technique. The benefits of this technique are well-studied, both for improving generalization [11] and more recently, for OOD detection [5–8]. But while OOD data have been used for tuning the hyperparameters of BNNs [12], it appears that even recently proposed deep Bayesian methods have not considered OOD training. A reason for this may be because of that

Bayesian Deep Learning workshop, NeurIPS 2021.

it is unclear how can one can incorporate OOD data in the Bayesian inference itself. Thus, in this work, we explore some options of incorporating OOD data to BNN training.

2 OOD Training

We focus on classification tasks. Let $F : \mathbb{R}^n \times \mathbb{R}^d \to \mathbb{R}^c$ defined by $(x, \theta) \mapsto F(x; \theta)$ be an ℓ-layer, c-class NN with any activation function. Here, $\mathbb{R}^n$, $\mathbb{R}^d$, and $\mathbb{R}^c$ are the input, parameter, and output spaces of the network, respectively. Let $P(X)$ and $P(Y|X)$ be unknown probability distributions on $\mathbb{R}^n$ and $\{1, \dots, c\}$, respectively. We assume that an i.i.d. dataset $\mathcal{D} := \{(x^{(i)}, y^{(i)})\}_{i=1}^m$ sampled from the previous distributions. Let $\sigma : \mathbb{R}^c \to \Delta^c$ be the softmax function. A common choice of likelihood function for c-class classification networks is the softmax-Categorical likelihood: Given $(x, y) \in \mathcal{D}$ and θ, the softmax output $\sigma(F(x; \theta))$ can be interpreted as a probability vector and the *Categorical log-likelihood* over it can be defined by $\log p_{\text{Cat}}(y|x, \theta) := \log \sigma_{y_k}(F(x; \theta))$.

Let U be the *data region*, i.e. a subset of the input space $\mathbb{R}^n$ where thedistribution $P(X)$ assigns non-negligible mass. Suppose $V := \mathbb{R}^n \setminus U$ is the remaining subset of the input space $\mathbb{R}^n$ that has low mass under $P(X)$, i.e. it is the *OOD region*. It has recently been shown that any point estimate of F can induce an arbitrarily overconfident prediction on V [6, 13]. Moreover, empirical evidence shows that BNNs often yield suboptimal results in this regime, as Fig. 1 shows.

In an adjacent field, the frequentist community has proposed a technique, referred to here as *OOD training*, to address this issue. The core idea is to "expose" the network to a particular kind OOD data and let it generalize to unseen outliers. Suppose $\mathcal{D}_{\text{out}} := \{\widehat{x}_i \in V\}_{i=1}^{m_{\text{out}}}$ is a collection of m_{out} points sampled from some distribution on V. Then, one can incorporate these OOD samples into the standard MAP objective via an additional objective function $\mathcal{L}$:

$$\arg\max_{\theta} \log p_{\text{Cat}}(\mathcal{D}|\theta) + \log p(\theta) + \mathcal{L}(\theta; \mathcal{D}_{\text{out}}). \tag{1}$$

For instance, Hendrycks et al. [4] define $\mathcal{L}$ to be the negative cross-entropy between the softmax output of F under $\mathcal{D}_{\text{out}}$ and the uniform discrete distribution. Empirically, this frequentist robust training scheme obtains state-of-the-art performance in OOD detection benchmarks [4, 7, 8, etc.].

3 OOD Training for BNNs

Here we explore two methods for incorporating OOD data in BNN training. Two additional methods are presented in the appendix.

Method 1: Extra "None Class" (`NC`)

The most straightforward yet philosophically clean way to incorporate unlabeled OOD data is by adding an extra class, corresponding to the "none class"—also known as the "dustbin" or "garbage" class [11]. Under this assumption, we only need to label all OOD data in $\mathcal{D}_{\text{out}}$ with the class $c+1$ and add them to the true dataset $\mathcal{D}$. That is, the new dataset is $\widetilde{\mathcal{D}} := \mathcal{D} \amalg \{(x_{\text{out}}^{(1)}, c+1), \dots, (x_{\text{out}}^{(m_{\text{out}})}, c+1)\}$, where $\amalg$ denotes disjoint union. Under this setting, we can directly use the Categorical likelihood and thus, a BNN with this assumption has a sound Bayesian interpretation.

Method 2: Frequentist-Loss Likelihood (`OE`)

Considering the effectiveness of OE, it is, therefore, tempting to give a direct Bayesian treatment upon it. But to do so, we first have to find a sound probabilistic justification of $\mathcal{L}$ in (1) since not all loss functions can be interpreted as (negative log-)likelihood functions. First, recall that OE's OOD objective—the last term in (1)—is given by

$$\mathcal{L}_{\text{OE}}(\theta;\mathcal{D}_{\text{out}}) := - \mathop{\mathbb{E}}_{x_{\text{out}} \sim \mathcal{D}_{\text{out}}} (H(\sigma(F(x_{\text{out}}; \theta)), u)) = \frac{1}{c\, m_{\text{out}}} \sum_{i=1}^{m_{\text{out}}} \sum_{k=1}^{c} \log \sigma_c(F(x_{\text{out}}^{(i)}; \theta)), \tag{2}$$

where $u := (1/c, \dots, 1/c)$ is the uniform probability vector of length c and H is the *cross-entropy* functional. Our goal here is to interpret (2) as a log-likelihood function: we aim at finding a log-likelihood function $\log p(\mathcal{D}_{\text{out}}|\theta)$ over $\mathcal{D}_{\text{out}}$ that has the form of $\mathcal{L}_{\text{OE}}$.

Table 1: OOD data detection in terms of FPR95. Values are averages over six OOD test sets and five prediction runs—lower is better. Best values of each categories are in bold.

Methods	MNIST	F-MNIST	SVHN	CIFAR-10	CIFAR-100
MAP	17.7	69.4	22.4	52.4	81.0
DE	10.6	61.4	10.1	32.3	73.3
OE	**5.4**	**16.2**	**2.1**	**22.8**	**54.0**
LL-VB	25.7	63.3	22.0	36.5	77.6
+NC	7.5	**15.0**	**1.4**	**28.0**	**49.9**
+OE	**6.8**	22.4	1.5	29.8	53.3
LA	19.4	68.7	17.1	53.6	81.3
+NC	6.6	**8.3**	1.5	**20.1**	**47.4**
+OE	**5.4**	17.0	**1.1**	23.3	53.9

We begin with the assumption that the Categorical likelihood is used to model both the in- and out-of-distribution data—in particular we use the standard integer labels for both $\mathcal{D}$ and $\mathcal{D}_\text{out}$. Now, recall that the OOD data ideally have the uniform confidence, that is, they are equally likely under all possible labels. But since we have assumed "hard" labels, we cannot use u directly as the label for $x_\text{out} \in \mathcal{D}_\text{out}$. To circumvent this, we redefine the OOD dataset $\mathcal{D}_\text{out}$ by assigning all c possible labels to each x_out, so that F is maximally confused about the correct label of x_out.

$$\mathcal{D}_\text{out} := \{(x_\text{out}^{(1)}, 1), \dots, (x_\text{out}^{(1)}, K), \dots, (x_\text{out}^{(m_\text{out})}, 1), \dots, (x_\text{out}^{(m_\text{out})}, K)\}. \tag{3}$$

Thus, given m_out unlabeled OOD data, we have $|\mathcal{D}_\text{out}| = c\, m_\text{out}$ OOD data points in our OOD training set. So, the negative log-Categorical likelihood over $\mathcal{D}_\text{out}$ is given by $-\log p(\mathcal{D}_\text{out}|\theta) = \sum_{i=1}^{m_\text{out}} \sum_{k=1}^{c} \log \sigma_k(F(x_\text{out}^{(i)};\theta))$. Comparing this to (2), we identify that $\log p(\mathcal{D}_\text{out}|\theta)$ is exactly $\mathcal{L}_\text{OE}$, up to a constant factor $1/(c\, m_\text{out})$, which can be thought as a tempering factor to $p(\mathcal{D}_\text{out}|\theta)$. We have thus obtained the probabilistic interpretation of OE's objective—this likelihood can then be soundly used in a Bayesian inference—albeit arising from applying a heuristic (3) to the data.

4 Experiments

We validate the approach via standard OOD detection benchmarks on MNIST, F-MNIST, SVHN, CIFAR-10, and CIFAR-100. For each of them, we use six OOD test sets and measure the performance via the FPR95 metric, which measure the false-positive rate at 95% true positive rate. We use the LeNet and WideResNet-16-4 architectures, trained in the usual manner. The OOD training set $\mathcal{D}_\text{out}$ is the 32×32 downsampled ImageNet dataset [14] as an alternative to the 80M Tiny Images dataset used by [4, 7], since the latter is not available anymore. As the base BNNs our methods are applied on, we use a simple last-layer mean-field variational Bayes (LL-VB) and all-layer diagonal Laplace approximation (LA). We use these simple BNNs to show that OOD training is effective even in this regime. Results with more sophisticated BNNs are in the appendix.

We present the OOD detection results in Table 3. As indicated in Fig. 1, OE is significantly better than even DE while retaining the computational efficiency of MAP. The vanilla Bayesian baselines (LL-VB, LA) achieve worse results than DE (and thus OE). But, when OOD training is employed to train these BNNs using the two methods we considered in the previous section, their performance improves. We observe that all Bayesian OOD training methods generally yield better results than DE and become competitive to OE. For more results, please refer to the appendix.

5 Conclusion

We raised an important observation regarding contemporary BNNs' performance in uncertainty quantification, in particular in OOD detection tasks. We noticed that BNNs tend to underperform compared to non-Bayesian UQ methods. We hypothesized that this issue is due to the fact that recent frequentist UQ methods utilize an auxiliary OOD training set. To validate this, we explored ways to incorporate OOD training data into BNNs while still maintaining a reasonable Bayesian interpretation. Our experimental results showed that using OOD data in approximate Bayesian inference significantly improved the performance of BNNs, making them competitive or even better than non-Bayesian counterparts. In particular, we found that the most philosophically Bayesian-compatible way of OOD training—simply add an additional "none class"—performs best. We hope that the studied methods can be strong baselines for future work in the Bayesian deep learning community.

Acknowledgments and Disclosure of Funding

The authors gratefully acknowledge financial support by the European Research Council through ERC StG Action 757275 / PANAMA; the DFG Cluster of Excellence "Machine Learning - New Perspectives for Science", EXC 2064/1, project number 390727645; the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039A); and funds from the Ministry of Science, Research and Arts of the State of Baden-Württemberg. AK is grateful to the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for support. AK is also grateful to Felix Dangel, Jonathan Schmidt, and other members of the Methods of Machine Learning group for feedback.

References

[1] Kazuki Osawa, Siddharth Swaroop, Mohammad Emtiyaz E Khan, Anirudh Jain, Runa Eschenhagen, Richard E Turner, and Rio Yokota. Practical Deep Learning with Bayesian Principles. In *NeurIPS*, 2019.

[2] Marcin Tomczak, Siddharth Swaroop, and Richard Turner. Efficient Low Rank Gaussian Variational Inference for Neural Networks. In *NeurIPS*, 2020.

[3] Michael Dusenberry, Ghassen Jerfel, Yeming Wen, Yian Ma, Jasper Snoek, Katherine Heller, Balaji Lakshminarayanan, and Dustin Tran. Efficient and Scalable Bayesian Neural Nets with Rank-1 Factors. In *ICML*, 2020.

[4] Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep Anomaly Detection with Outlier Exposure. In *ICLR*, 2019.

[5] Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training Confidence-Calibrated Classifiers for Detecting Out-of-Distribution Samples. In *ICLR*, 2018.

[6] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why Relu Networks Yield High-Confidence Predictions Far Away from the Training Data and How to Mitigate the Problem. In *CVPR*, 2019.

[7] Alexander Meinke and Matthias Hein. Towards Neural Networks that Provably Know when They don't Know. In *ICLR*, 2020.

[8] Julian Bitterwolf, Alexander Meinke, and Matthias Hein. Certifiably Adversarially Robust Detection of Out-of-Distribution Data. In *NeurIPS*, 2020.

[9] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles. In *NIPS*, 2017.

[10] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty under Dataset Shift. In *NeurIPS*, 2019.

[11] Xiang Zhang and Yann LeCun. Universum Prescription: Regularization Using Unlabeled Data. In *AAAI*, 2017.

[12] Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being Bayesian, Even Just a Bit, Fixes Overconfidence in ReLU Networks. In *ICML*, 2020.

[13] Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, 2015.

[14] Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A Downsampled Variant of ImageNet as an Alternative to the CIFAR Datasets. *arXiv preprint arXiv:1707.08819*, 2017.

[15] Christian Thiel. Classification on Soft Labels is Robust Against Label Noise. In *International Conference on Knowledge-Based and Intelligent Information and Engineering Systems*, 2008.

[16] Ludmila Kuncheva. *Fuzzy Classifier Design*. Springer Science & Business Media, 2000.

[17] Neamat El Gayar, Friedhelm Schwenker, and Günther Palm. A Study of the Robustness of KNN Classifiers Trained Using Soft Labels. In *IAPR Workshop on Artificial Neural Networks in Pattern Recognition*, 2006.

[18] Thomas Minka. Estimating a Dirichlet distribution, 2000.

[19] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In *NIPS*, 2018.

[20] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *CVPR*, 2016.

[21] Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. EMNIST: Extending MNIST to Handwritten Letters. In *International Joint Conference on Neural Networks*, 2017.

[22] Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep Learning for Classical Japanese Literature. *arXiv preprint arXiv:1812.01718*, 2018.

[23] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. LSUN: Construction of a Large-Scale Image Dataset Using Deep Learning with Humans in the Loop. *arXiv preprint arXiv:1506.03365*, 2015.

[24] Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. Multi30K: Multilingual English-German Image Descriptions. In *ACL Workshop on Vision and Language*, 2016.

[25] Samuel R Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. A Large Annotated Corpus for Learning Natural Language Inference. In *EMNLP*, 2015.

[26] Dan Hendrycks and Thomas Dietterich. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. In *ICLR*, 2019.

[27] Sergey Zagoruyko and Nikos Komodakis. Wide Residual Networks. In *BMVC*, 2016.

[28] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. In *ICLR*, 2017.

[29] Guodong Zhang, Shengyang Sun, David Duvenaud, and Roger Grosse. Noisy Natural Gradient as Variational Inference. In *ICML*, 2018.

[30] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. In *EMNLP*, 2014.

[31] Andrey Malinin and Mark Gales. Reverse KL-Divergence Training of Prior Networks: Improved Uncertainty and Adversarial Robustness. In *NIPS*, 2019.

[32] Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient Pseudo-Independent Weight Perturbations on Mini-Batches. In *ICLR*, 2018.

[33] Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical Stochastic Gradient MCMC for Bayesian Deep Learning. In *ICLR*, 2020.

Appendix A Additional OOD-Training Methods for BNNs

Method 3: Soft Labels (SL)

In this method, we simply assume that the data have "soft labels", i.e. the labels are treated as general probability vectors, instead of restricted to integer labels [15].[1] Thus, we can assume that the target Y is a Δ^c-valued random variable, where Δ^c is the $(c-1)$-probability simplex. Under this assumption, since one-hot vectors are also elements of Δ^c (they represent the c corners of Δ^c), we do not have to redefine $\mathcal{D}$ other than to one-hot encode the original integer labels.

Now let us turn our attention to the OOD training data. The fact that these data should be predicted with maximum entropy suggests that the suitable label for any $x_{\text{out}} \in \mathcal{D}_{\text{out}}$ is the uniform probability vector $u := (1/c, \dots, 1/c)$ of length c—the center of Δ^c. Therefore, we can redefine $\mathcal{D}_{\text{out}}$ as the set $\{(x_{\text{out}}^{(i)}, u)\}_{i=1}^m$, and then define a new joint dataset $\widetilde{\mathcal{D}} := \mathcal{D} \amalg \mathcal{D}_{\text{out}}$ containing both the soft-labeled in- and out-distribution training data. Note that without the assumption that Y is a simplex-valued r.v., we cannot assign the label u to the OOD training data, and thus we cannot naturally convey our intuition that we should be maximally uncertain over OOD data.

Under this assumption, we have to adapt the likelihood. A straightforward choice for simplex-valued r.v.s. is the Dirichlet likelihood $p_{\text{Dir}}(y|x,\theta) := \text{Dir}(y|\alpha(F(x;\theta)))$ where we have make the dependence of α to the network output $F(x;\theta)$ explicit. So, we obtain the log-likelihood function

$$\log p_{\text{Dir}}(y|x,\theta) = \log\Gamma\left(\alpha_0\right) - \log\Gamma(\alpha_k(F(x;\theta))) + \sum_{k=1}^{c}(\alpha_k(F(x;\theta)) - 1)\log y_k, \tag{1}$$

where $\alpha_0 := \sum_{k=1}^c \alpha_k(F(x;\theta))$. Therefore, the log-likelihood for $\widetilde{\mathcal{D}}$ is given by $\log p_{\text{Dir}}(\widetilde{\mathcal{D}}|x,\theta) = \sum_{i=1}^m \log p_{\text{Dir}}(y^{(i)}|x^{(i)},\theta) + \sum_{i=1}^{m_{\text{out}}} \log p_{\text{Dir}}(u|x_{\text{out}}^{(i)},\theta)$, which can readily be used in a Bayesian inference.

One thing left to discuss is the definition of $\alpha(F(x;\theta))$. An option is to decompose it into mean and precision [18]. We do so by writing $\alpha_k(F(x;\theta) = \gamma\,\sigma_k(F(x;\theta))$ for each $k = 1,\dots,c$, where γ is the precision (treated as a hyperparameter) and the softmax output $\sigma(F(x;\theta))$ represents the mean—which is valid since it is an element of Δ^c. The benefits are two-fold: First, since we focus solely on the mean, it is easier for optimization [18]. Indeed, we found that the alternatives, such as $\alpha_k(F(x;\theta)) = \exp(F_k(x;\theta))$ yield worse results. Second, after training, we can use the softmax output of F as usual without additional steps, i.e. when making prediction, we can treat the network as if it was trained using the standard softmax-Categorical likelihood.

Method 4: Mixed Labels (ML)

There is a technical issue when using the Dirichlet likelihood for the in-distribution data: It is known that the Dirichlet likelihood does not work well with one-hot encoded vectors and harder to optimize than the Categorical likelihood [19]. To see this, notice in (1) that the logarithm is applied on y_k, in contrast to $\sigma_k(F(x;\theta))$ in the Categorical likelihood. If y is a one-hot encoded vector, this implies that for all but one $k \in \{1,\dots,c\}$, the expression $\log y_k$ is undefined and thus the entire log-likelihood also is. While one can mitigate this issue via e.g. label smoothing [19, 20], ultimately we found that models with the Dirichlet likelihood generalize worse than their Categorical counterparts. Fortunately, the Dirichlet log-likelihood (1) does not suffer from this issue when used for OOD data because their label u is the uniform probability vector—in particular, all components of u are strictly larger than zero.

Motivated by these observations, we combine the best of best worlds in the stability of the Categorical likelihood in modeling "hard" one-hot encoded labels (or equivalently, integer labels) and the flexibility of the Dirichlet likelihood in modeling soft labels. To this end, we assume that all the in-distribution data in $\mathcal{D}$ have the standard integer labels, while all the OOD data in $\mathcal{D}_{\text{out}}$ have soft labels. Then, assuming $\widetilde{\mathcal{D}} = \mathcal{D} \amalg \mathcal{D}_{\text{out}}$, we define the following "mixed" log-likelihood:

$$\log p(\widetilde{\mathcal{D}}|\theta) := \sum_{i=1}^{m} \log p_{\text{Cat}}(y^{(i)}|x^{(i)},\theta) + \sum_{i=1}^{m_{\text{out}}} \log p_{\text{Dir}}(u|x_{\text{out}}^{(i)},\theta).$$

[1] The term "soft label" here is different than "fuzzy label" [16, 17] where it is not constrained to sum to one.

The implicit assumption of this formulation is that, unlike the two previous methods, we have two distinct generative processes for generating the labels of input points in U and V. Data in $\mathcal{D}$ can thus have a different "data type" than data in $\mathcal{D}_{\text{out}}$. This method can therefore be interpreted as solving a multi-task or multi-modal learning problem.

Appendix B OOD Test Sets

For image-based OOD detection tasks, we use the following test sets on top of MNIST, F-MNIST, SVHN, CIFAR-10, and CIFAR-100:

- E-MNIST: Contains handwritten letters ("a"-"z")—same format as MNIST [21].
- K-MNIST: Contains handwritten Hiragana scripts—same format as MNIST [22].
- LSUN-CR: Contains real-world images of classrooms [23].
- CIFAR-GR: Obtained by converting CIFAR-10 test images to grayscale.
- F-MNIST-3D: Obtained by converting single-channel F-MNIST images into three-channel images—all these three channels have identical values.
- UNIFORM: Obtained by drawing independent uniformly-distributed random pixel.
- SMOOTH: Obtained by permuting, smoothing, and contrast-rescaling the original (i.e. the respective in-distribution) test images [6].

Meanwhile, for text classification, we use the following OOD test set, following [4]:

- MULTI30K: Multilingual English-German image description dataset [24].
- WMT16: Machine-translation dataset, avaliable at `http://www.statmt.org/wmt14/translation-task.html`.
- SNLI: Collection of human-written English sentence pairs manually labeled for balanced classification with the labels entailment, contradiction, and neutral [25].

Finally, for dataset-shift robustness tasks, we use the standard dataset:

- CIFAR-10-C: Contains 19 different perturbations—e.g. snow, motion blur, brightness rescaling—with 5 level of severity for a total of 95 distinct shifts [26, 10].

Appendix C Details

Training: Non-Bayesian For MNIST and F-MNIST, we use a five-layer LeNet architecture. Meanwhile, for SVHN, CIFAR-10, and CIFAR-100, we use WideResNet-16-4 [27]. For all methods, the training procedures are as follows. For LeNet, we use Adam with initial learning rate of 1×10^{-3} and annealed it using the cosine decay method [28] along with weight decay of 5×10^{-4} for 100 epochs. We use a batch size of 128 for both in- and out-distribution batch, amounting to an effective batch size of 256 in the case of OOD training. The standard data augmentation pipeline (random crop and horizontal flip) is applied to both in-distribution and OOD data. For WideResNet-16-4, we use SGD instead with an initial learning rate of 1×10^{-1} and Nesterov momentum of 0.9 along with the dropout regularization with rate 0.3—all other hyperparameters are identical to LeNet. Finally, we use 5 ensemble members for DE.

Training: Bayesian For both LA, VB, and their variants (i.e. LA+X and VB+X), we use the identical setup as in the non-Bayesian training above. Additionally, for LA and LA+X, we use the diagonal Fisher matrix as the approximate Hessian. Moreover, we tune prior variance by minimizing the validation Brier score. All predictions are done using 20 MC samples. For VB and VB+X, we use a diagonal Gaussian variational posterior for both the last-layer weight matrix and bias vector. Moreover, the prior is a zero-mean isotropic Gaussian with prior precision 5×10^{-4} (to emulate the choice of the weight decay in the non-Bayesian training). The trade-off hyperparameter τ of the ELBO is set to the standard value of 0.1 [1, 29]. We do not use weight decay on the last layer since the regularization of its parameters is done by the KL-term of the ELBO. Lastly, we use 5 and 200 MC samples for computing the ELBO and for making predictions, respectively.

Table 2: Test accuracy / ECE, averaged over five prediction runs. Best values in each categories are in bold.

	MNIST	**F-MNIST**	**SVHN**	**CIFAR10**	**CIFAR100**
MAP	99.4 / 6.4	92.4 / 13.9	97.4 / 8.9	94.8 / 10.0	76.7 / 14.3
DE	**99.5** / 8.6	**93.6 / 3.6**	**97.6 / 3.5**	**95.7 / 4.5**	**80.0 / 1.9**
OE	99.4 / **5.3**	92.3 / 12.1	97.4 / 10.6	94.6 / 13.2	76.7 / 15.0
VB	**99.5** / 11.2	92.4 / 3.7	97.5 / 5.7	94.9 / 5.8	**75.4 / 8.3**
+NC	99.4 / 12.6	92.2 / 3.3	97.5 / **4.1**	94.4 / 5.5	74.1 / 10.7
+SL	**99.5** / 10.5	**93.1** / 6.3	**97.6** / 9.3	93.0 / 11.0	71.4 / 13.0
+ML	99.3 / 11.8	92.0 / 2.5	**97.6** / 4.2	**95.0** / 4.9	**75.4** / 10.4
+OE	99.4 / **10.0**	92.3 / 3.0	**97.6** / 5.7	94.8 / **4.6**	74.2 / 8.9
LA	99.4 / 7.6	92.5 / 11.3	97.4 / 3.3	**94.8** / 7.5	76.6 / 8.3
+NC	99.4 / 5.4	92.4 / 8.5	97.3 / 4.6	94.0 / **6.6**	76.2 / 6.1
+SL	**99.7** / 12.1	**93.2 / 3.2**	**97.5** / 7.4	93.6 / 10.2	72.3 / 7.1
+ML	99.4 / 7.5	92.5 / 5.9	97.4 / **2.9**	**94.8** / 6.9	76.5 / **4.4**
+OE	99.4 / **4.8**	92.3 / 7.4	97.4 / 3.2	94.6 / 8.8	**76.7 / 4.4**

Table 3: OOD data detection in terms of FPR95. Values are averages over six OOD test sets and five prediction runs—lower is better. Best values of each categories are in bold.

Methods	**MNIST**	**F-MNIST**	**SVHN**	**CIFAR-10**	**CIFAR-100**
MAP	17.7	69.4	22.4	52.4	81.0
DE	10.6	61.4	10.1	32.3	73.3
OE	**5.4**	**16.2**	**2.1**	**22.8**	**54.0**
VB	25.7	63.3	22.0	36.5	77.6
+NC	7.5	15.0	**1.4**	**28.0**	**49.9**
+SL	**2.7**	**4.2**	1.8	40.4	62.3
+ML	7.4	19.6	**1.4**	29.1	50.2
+OE	6.8	22.4	1.5	29.8	53.3
LA	19.4	68.7	17.1	53.6	81.3
+NC	6.6	8.3	1.5	**20.1**	**47.4**
+SL	**2.2**	**4.1**	**1.0**	38.5	60.9
+ML	5.5	14.3	1.1	21.8	52.5
+OE	5.4	17.0	1.1	23.3	53.9

Text Classification The network used is a two-layer Gated Recurrent Unit [GRU, 30] with 128 hidden units on each layer. The word-embedding dimension is 50 and the maximum vocabulary size is 10000. We put an affine layer on top of the last GRU output to translate the hidden units to output units. Both the LA and VB are applied only on this layer. We use batch size of 64 and Adam optimizer with learning rate of 0.01 without weight decay, except for LA in which case we use weight decay of 5×10^{-4}. The optimization is done for 5 epochs, following [4].

Appendix D Additional Results

D.1 Generalization and Calibration

We present the generalization and calibration performance in Table 2. We note that generally, all the proposed methods attain comparable accuracy to and are better calibrated than the vanilla MAP/OE models. However, the "soft label" method tends to underperform in both accuracy and ECE—this can be seen clearly on CIFAR-100. This issue appears to be because of the numerical issue we have discussed before in Appendix A. Note that this issue seems to also plague other Dirichlet-based methods [19, 31]. Overall, it appears that Bayesian OOD training with NC, ML, and OE is not harmful to the in-distribution performance—they are even more calibrated than the frequentist OE.

Table 4: OOD data detection on text classification tasks. Values are averages over five prediction runs and additionally, three OOD test sets for FPR95.

	ECE		FPR95	
Methods	**SST**	**TREC**	**SST**	**TREC**
MAP	20.8	17.2	100.0	96.3
DE	**2.5**	10.6	100.0	24.2
OE	13.0	**9.4**	**0.0**	**0.0**
LA	21.0	17.3	100.0	96.4
+NC	17.9	18.6	**0.0**	**0.0**
+SL	17.5	10.4	95.3	0.8
+ML	**11.4**	11.5	84.6	**0.0**
+OE	12.8	**8.4**	**0.0**	**0.0**

Table 5: OOD data detection under models trained with synthetic noises as $\mathcal{D}_{\text{out}}$. Values are FPR95, averaged over five prediction runs and all OOD test sets.

Methods	**SVHN**	**CIFAR-10**	**CIFAR-100**
MAP	22.4	52.4	81.0
OE	**11.4**	**31.0**	**60.1**
LA	17.1	53.6	81.3
+NC	10.5	**26.4**	64.5
+SL	93.7	37.9	68.6
+ML	14.4	28.4	61.0
+OE	**10.1**	35.3	**56.4**

D.2 OOD Detection

We present the OOD detection results on image classification datasets in Table 3. As indicated in Fig. 1, OE is in general significantly better than even DE while retaining the computational efficiency of MAP. The vanilla Bayesian baselines, represented by VB and LA, achieve worse results than DE (and thus OE). But, when OOD training is employed to train these BNNs using the four methods we considered, their performance improves. We observe that all Bayesian OOD training methods generally yield better results than DE and become competitive to OE. In particular, while the "soft-label" method (SL) is best for "easy" datasets (MNIST, F-MNIST), we found that the simplest "none class" method (NC) achieves the best results in general.

In Table 4, we additionally show the results on text classification datasets. We found that the OOD training methods consistently improve both the calibration and OOD-detection performance of the vanilla Bayesian methods, making them on par with OE. As before, the "none class" method performs the best in OOD detection. This is a reassuring result since NC is also the most philosophically clean (i.e. requires fewer heuristics) than the other three methods considered.

A common concern regarding OOD training is the choice of $\mathcal{D}_{\text{out}}$. As an attempt to address this, in Table 5 we provide results on OOD detection when the model is trained using a synthetic noise dataset, instead of the 32×32 ImageNet dataset. The noise dataset used here is the "smooth noise" dataset [6], obtained by permuting, blurring, and contrast-rescaling the original training dataset. We found that even with such a simple OOD dataset, we can still generally obtain better results than OE.

Finally, we show that OOD training is beneficial not only for the LA and VB baselines. In Table 6, we consider two recent (all-layer) BNNs: a VB with the flipout estimator [Flipout, 32] and the cyclical stochastic-gradient Hamiltonian Monte Carlo [CSGHMC, 33]. Evidently, OOD training improve their OOD detection performance by large margin. Moreover, we also note that OOD training also improves the performance of DE.

Table 6: OOD data detection with more sophisticated base models. Values are FPR95, averaged over five prediction runs and all OOD test sets.

Methods	**CIFAR-10**	**CIFAR-100**
Flipout	65.0	85.4
+NC	**40.9**	**56.2**
CSGHMC	60.3	81.0
+NC	**25.0**	**43.0**
DE	32.3	73.3
+NC	**17.0**	**44.4**

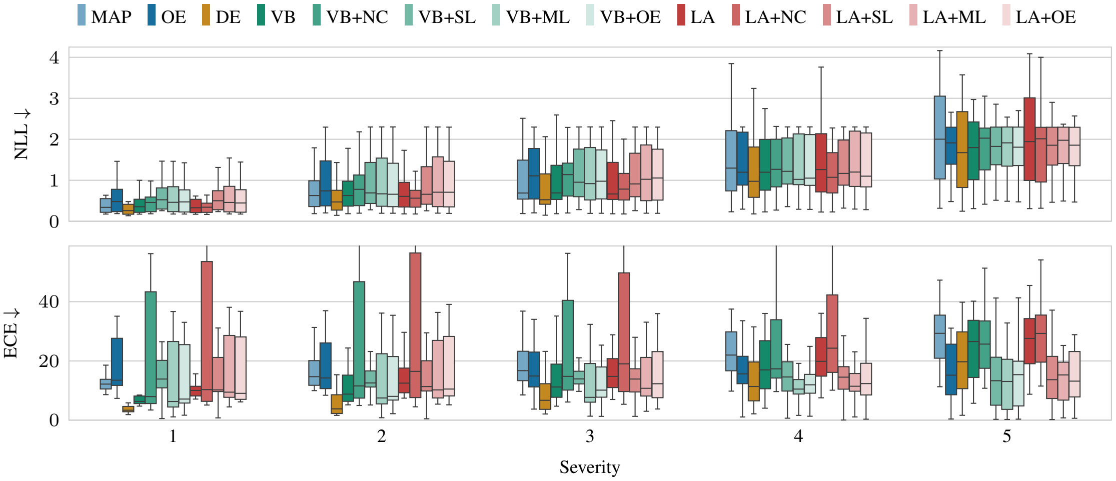

Figure 2: Dataset shift performance on the CIFAR-10-C dataset (lower is better).

D.3 Dataset-Shift Robustness

In this UQ task, OOD training is beneficial for both MAP and the vanilla Bayesian methods (VB, LA), making them competitive to the state-of-the-art DE's performance in larger severity levels, see Fig. 2. Moreover, the OOD-trained VB and LA are in general more calibrated than OE, which shows the benefit of the Bayesian formalism vis-á-vis the point-estimated OE. This indicates that *both* being Bayesian and considering OOD data during training are beneficial.

Even though it is the best in OOD detection, here we observe that NC is less calibrated in terms of ECE than its counterparts. This might be due to the incompatibility of calibration metrics with the additional class: When the data are corrupted, they become closer to the OOD data, and thus NC tends to assign higher probability mass to the last class which does not correspond to any of the true classes (contrast this to other the approaches). Therefore, in this case, the confidence over the true class becomes necessarily lower—more so than the other approaches. Considering that calibration metrics depend on the confidence of the true class, the calibration of NC thus suffers. One way to overcome this issue is to make calibration metrics aware of the "none class", e.g. by measuring calibration only on data that have low "none class" probability. We leave the investigation for future research.

D.4 Costs

The additional costs associated with all the OOD-training methods presented here are negligible: Like other non-Bayesian OOD-training methods, the only overhead is the additional minibatch of OOD training data at each training iteration. That is, these costs are similar to when considering a standard training procedure with double the minibatch size. Additionally for LA, in its Hessian computation, one effectively computes it with twice the number of the original data. However, this only needs to be done *once* post-training.

Appendix E Non-Averaged Results

The detailed, non-averaged results for the FPR95 metric are in Table 7. For the full results of FPR95 with the SMOOTH noise dataset as $\mathcal{D}_{\text{out}}$ are in Table 8. Furthermore, the full results of the NLP experiment is in Table 9. Finally, detailed, non-averaged results for OOD detection with Flipout and CSGHMC are in Table 10.

Table 7: OOD data detection in terms of FPR95. Lower is better. Values are averages over five prediction runs.

				VB					LA					
Datasets	**MAP**	**OE**	**DE**	**Plain**	**NC1**	**NC2**	**SL**	**ML**	**OE**	**Plain**	**NC1**	**NC2**	**SL**	**ML**	**OE**
MNIST															
F-MNIST	11.8	**0.0**	5.3	12.5	0.1	**0.0**	**0.0**	0.4	1.1	12.0	0.2	**0.0**	**0.0**	0.1	**0.0**
E-MNIST	35.6	26.4	30.4	34.5	34.7	32.8	14.3	34.2	31.4	35.8	30.6	19.5	**12.6**	26.8	26.7
K-MNIST	14.4	5.9	7.7	14.0	10.5	8.1	2.1	9.7	8.5	14.5	8.9	6.5	**0.7**	5.8	5.9
CIFAR-Gr	0.2	**0.0**	**0.0**	0.2	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.2	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Uniform	44.3	**0.0**	19.8	93.1	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	54.2	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Smooth	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
F-MNIST															
MNIST	73.5	38.5	65.8	66.8	43.5	10.4	9.5	50.1	57.2	72.2	25.6	**3.5**	11.5	38.9	39.9
E-MNIST	73.6	21.0	58.6	68.1	18.7	3.0	5.0	34.0	40.6	72.2	6.0	**0.5**	4.6	14.7	23.1
K-MNIST	73.7	37.4	47.2	62.6	28.0	6.1	10.6	33.4	36.7	71.6	18.2	**1.8**	8.7	32.5	38.7
CIFAR-Gr	87.2	**0.0**	86.6	75.3	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	87.7	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Uniform	81.3	**0.0**	86.3	87.3	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	81.0	**0.0**	**0.0**	**0.0**	**0.0**	0.1
Smooth	26.8	**0.0**	24.2	19.6	**0.0**	**0.0**	**0.0**	0.2	0.1	27.3	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
SVHN															
CIFAR-10	18.9	0.1	9.5	15.0	0.3	0.1	0.2	**0.0**	0.1	15.4	0.4	**0.0**	**0.0**	**0.0**	0.1
LSUN-CR	19.7	**0.0**	8.3	17.2	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	15.5	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
CIFAR-100	21.8	0.2	11.6	18.1	0.5	0.2	0.5	**0.1**	0.2	17.6	0.6	**0.1**	0.2	0.2	**0.1**
FMNIST-3D	26.7	**0.0**	17.5	24.5	**0.0**	**0.0**	0.6	**0.0**	**0.0**	27.2	0.1	**0.0**	**0.0**	**0.0**	**0.0**
Uniform	30.0	**0.0**	6.4	48.2	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	17.0	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Smooth	17.3	12.0	6.9	9.1	7.7	9.7	9.5	8.3	8.4	10.1	8.1	**5.3**	5.9	6.6	6.4
CIFAR-10															
SVHN	34.5	10.0	33.9	33.5	30.6	5.3	59.4	18.3	33.9	35.5	12.7	**2.6**	47.2	8.7	10.8
LSUN-CR	53.3	28.0	44.0	49.4	25.9	8.3	43.7	36.8	34.8	53.8	17.5	**1.3**	41.2	30.1	28.4
CIFAR-100	61.2	57.8	**52.5**	58.4	58.5	55.3	63.3	56.8	57.1	61.4	59.6	62.6	62.2	60.4	57.9
FMNIST-3D	42.4	26.8	30.7	37.4	19.0	5.4	43.9	32.2	29.6	43.2	15.4	**2.8**	36.8	24.2	27.8
Uniform	87.7	**0.0**	**0.0**	13.8	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	92.8	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Smooth	35.1	14.2	32.9	26.4	34.0	8.1	31.9	30.3	23.1	34.9	15.5	**2.8**	43.6	7.5	14.9
CIFAR-100															
LSUN-CR	82.0	64.3	75.3	73.8	62.3	23.9	76.3	65.3	67.6	82.8	55.9	**13.5**	75.6	65.3	64.1
CIFAR-10	79.8	81.9	**76.4**	78.2	81.4	90.9	82.8	79.5	79.0	79.5	80.9	91.4	81.7	80.8	80.0
FMNIST-3D	65.8	58.5	61.8	57.1	41.0	**14.0**	72.0	51.7	56.0	66.1	58.6	21.1	69.0	59.2	59.3
Uniform	97.6	**0.0**	94.3	100.0	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	98.8	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Smooth	79.5	65.2	58.7	79.1	64.8	29.4	80.2	54.4	64.0	79.2	41.6	**2.6**	78.0	57.1	66.2

Table 8: OOD data detection under models trained with random noises [6] as $\mathcal{D}_{\text{out}}$. Values are FPR95, averaged over ten prediction runs—lower is better.

				VB						**LA**					
Datasets	**MAP**	**OE**	**DE**	**Plain**	**NC1**	**NC2**	**SL**	**ML**	**OE**	**Plain**	**NC1**	**NC2**	**SL**	**ML**	**OE**
MNIST															
F-MNIST	11.8	6.8	5.3	12.5	6.5	6.5	0.6	11.9	10.3	12.0	8.2	4.3	**0.0**	6.3	6.8
E-MNIST	35.6	30.7	30.4	34.5	35.1	42.4	17.9	37.3	34.3	35.8	34.2	39.2	**15.3**	31.0	30.7
K-MNIST	14.4	7.8	7.7	14.0	14.5	17.1	1.1	15.8	14.0	14.5	10.6	9.5	**0.7**	8.5	7.8
CIFAR-Gr	0.2	**0.0**	**0.0**	0.2	**0.0**	**0.0**	**0.0**	**0.0**	0.1	0.2	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
Uniform	44.3	0.7	19.8	93.1	**0.0**	**0.0**	**0.0**	1.9	1.8	54.2	0.7	**0.0**	**0.0**	0.6	0.8
F-MNIST															
MNIST	73.5	62.2	65.8	66.8	62.2	34.6	30.4	59.1	60.4	72.2	60.8	**13.8**	24.3	55.4	57.4
E-MNIST	73.6	50.2	58.6	68.1	43.9	13.0	25.7	54.7	54.6	72.2	44.2	**3.3**	22.5	39.6	48.3
K-MNIST	73.7	47.4	47.2	62.6	31.4	8.2	19.1	35.6	38.1	71.6	33.9	**2.9**	20.9	31.7	43.0
CIFAR-Gr	87.2	0.5	86.6	75.3	0.1	**0.0**	0.2	0.8	1.1	87.7	0.7	**0.0**	0.2	0.7	1.0
Uniform	81.3	26.0	86.3	87.3	47.1	0.6	**0.0**	0.2	4.9	81.0	43.4	**0.0**	**0.0**	22.0	38.1
SVHN															
CIFAR-10	18.9	13.8	9.5	15.0	13.0	14.5	16.5	**8.4**	11.5	15.4	**8.4**	18.7	94.8	14.9	11.4
LSUN-CR	19.7	9.0	8.3	17.2	10.5	8.8	8.8	5.4	9.1	15.5	8.3	**3.5**	95.4	12.6	8.2
CIFAR-100	21.8	15.6	11.6	18.1	14.8	16.4	17.9	**10.2**	12.4	17.6	11.6	21.4	93.9	16.6	13.4
FMNIST-3D	26.7	29.8	**17.5**	24.5	31.1	34.8	30.4	30.0	25.3	27.2	34.6	70.5	95.1	23.3	27.7
Uniform	30.0	**0.0**	6.4	48.2	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	17.0	**0.0**	**0.0**	90.2	19.0	**0.0**
CIFAR-10															
SVHN	34.5	7.3	33.9	33.5	11.1	3.8	20.6	11.0	9.7	35.5	6.6	**0.8**	26.0	7.5	8.3
LSUN-CR	53.3	49.0	44.0	49.4	46.7	35.0	61.7	45.7	47.6	53.8	48.9	**27.8**	54.9	51.9	48.3
CIFAR-100	61.2	58.2	52.5	58.4	57.7	49.6	71.1	56.3	56.6	61.4	59.3	**49.4**	70.7	57.6	59.0
FMNIST-3D	42.4	44.9	30.7	37.4	39.4	24.3	62.7	43.3	44.0	43.2	40.2	**14.1**	57.8	40.8	46.4
Uniform	87.7	26.7	**0.0**	13.8	100.0	100.0	57.3	98.0	100.0	92.8	3.5	**0.0**	17.7	12.9	49.8
CIFAR-100															
LSUN-CR	82.0	79.7	75.3	73.8	80.9	83.9	91.5	77.9	**71.1**	82.8	82.0	85.3	78.5	72.8	79.7
CIFAR-10	79.8	80.5	**76.4**	78.2	81.3	86.9	93.9	80.0	81.4	79.5	80.6	88.9	82.1	78.8	80.2
FMNIST-3D	65.8	66.9	61.8	57.1	69.3	**42.1**	93.6	63.1	61.9	66.1	71.2	60.8	82.0	64.4	67.9
Uniform	97.6	73.3	94.3	100.0	99.7	**49.5**	95.4	99.5	99.8	98.8	88.4	100.0	100.0	88.9	54.0

Table 9: OOD data detection on text classification tasks. Values are FPR95, averaged over five prediction runs—lower is better.

				LA					
Datasets	**MAP**	**OE**	**DE**	**Plain**	**NC1**	**NC2**	**SL**	**ML**	**OE**
SST									
SNLI	100.0	**0.0**	100.0	100.0	**0.0**	**0.0**	97.0	89.6	**0.0**
Multi30k	100.0	**0.0**	100.0	100.0	**0.0**	**0.0**	99.5	83.5	**0.0**
WMT16	100.0	**0.0**	100.0	100.0	**0.0**	**0.0**	89.3	80.7	**0.0**
TREC									
SNLI	99.7	**0.0**	31.0	99.7	**0.0**	**0.0**	0.7	**0.0**	**0.0**
Multi30k	100.0	**0.0**	14.2	100.0	**0.0**	**0.0**	0.8	**0.0**	**0.0**
WMT16	89.2	**0.0**	27.3	89.3	**0.0**	**0.0**	0.8	**0.0**	**0.0**

Table 10: OOD data detection with more sophisticated base models. Values are FPR95, averaged over five prediction runs.

	Flipout		**CSGHMC**		**DE**	
Datasets	**Plain**	**NC**	**Plain**	**NC**	**Plain**	**NC**
CIFAR-10						
SVHN	72.1	39.6	56.8	16.4	33.9	**8.1**
LSUN-CR	63.7	37.5	56.7	24.0	44.0	**18.3**
CIFAR-100	74.5	70.4	63.4	63.1	52.5	**51.7**
FMNIST-3D	65.0	38.2	51.0	14.8	30.7	**10.3**
Uniform	53.8	**0.0**	87.0	**0.0**	0.0	**0.0**
Smooth	61.1	59.6	47.1	31.9	32.9	**13.6**
CIFAR-100						
LSUN-CR	85.8	55.6	79.3	**38.0**	75.3	54.0
CIFAR-10	86.1	87.0	82.1	84.2	76.4	**78.5**
FMNIST-3D	73.4	65.8	67.0	**45.5**	61.8	50.0
Uniform	99.7	**0.0**	93.8	**0.0**	94.3	**0.0**
Smooth	82.0	72.5	83.0	47.2	58.7	**39.3**