5th Symposium on Advances in Approximate Bayesian Inference, 2023 1–36

Function-Space Regularization for Deep Bayesian Classification

Anonymous Authors
Anonymous Institution

Abstract

Bayesian deep learning approaches assume model parameters to be latent random variables and infer posterior distributions to quantify uncertainty, increase safety and trust, and prevent overconfident and unpredictable behavior. However, weight-space priors are model-specific, can be difficult to interpret and are hard to specify. Instead, we apply a Dirichlet prior in predictive space and perform approximate function-space variational inference. To this end, we interpret conventional categorical predictions from stochastic neural network classifiers as samples from an implicit Dirichlet distribution. By adapting the inference, the same function-space prior can be combined with different models without affecting model architecture or size. We illustrate the flexibility and efficacy of such a prior with toy experiments and demonstrate scalability, improved uncertainty quantification and adversarial robustness with large-scale image classification experiments.

1. Introduction

Deep learning has enabled powerful classification models capable of working with complex data modalities and scaling to large data sets (Krizhevsky et al., 2012; Goodfellow et al., 2016). The aim of *Bayesian* neural networks (BNNs) is to provide these complex models with priors for regularization, generalization and uncertainty quantification (UQ) useful in prediction tasks (Gal, 2016; Wilson and Izmailov, 2020; Fortuin, 2021; Abdar et al., 2021). Predictive uncertainty is crucial for machine learning systems in real-world settings, as it provides a degree of safety (McAllister et al., 2017), trust (Lim et al., 2019), sample efficiency (Deisenroth and Rasmussen, 2011; Gal et al., 2017) and human-in-the-loop cooperation (Filos et al., 2019). In this work, we leverage function-space variational inference [1] (Sun et al., 2019) (fVI) to implement regularization for classification tasks. Function-space priors can explicitly affect the predictive distribution and do not depend on the particular model parameterization, whereas weight-space priors are implicit and model-specific. Given any stochastic neural network capable of producing multiple predictions, such as Monte Carlo dropout (Gal and Ghahramani, 2016) or deep ensembles (Lakshminarayanan et al., 2017), we estimate a Dirichlet predictive distribution from several categorical outputs via maximum likelihood. This approach retains the same mean prediction of conventional deep learning classifiers, while also capturing the information contained in the variance of the outputs. The Dirichlet predictive distribution can then be used to specify a function-space prior to regularize classification. Various prior work which uses the Dirichlet distribution and function-space regularization (Malinin and Gales, 2018; Malinin et al., 2020; Joo et al., 2020; Sensoy et al., 2018, 2021) can be viewed as a special case of fVI. We demonstrate that our method improves uncertainty quantification and adversarial robustness across a range of popular models and datasets for both small- and large-scale inference.

1. We use *function-space* VI instead of *functional* VI to avoid the overloaded term with different connotations.

© A. Authors.

Authors

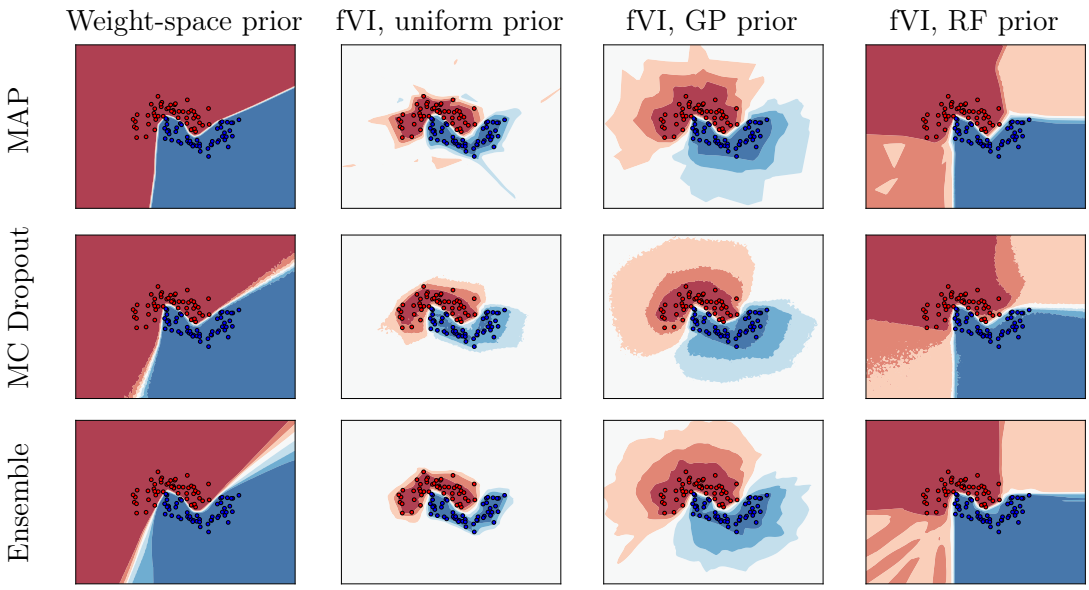
Weight-space prior
fVI, uniform prior
fVI, GP prior
fVI, RF prior
MAP
MC Dropout
Ensemble

Figure 1: Toy classification problem using the Two Moons dataset. The leftmost column shows the undesirable overconfidence of standard weight-space priors outside of the data distribution. The second column illustrates how our function-space inference approach combined with a uniform Dirichlet prior adequately increases model uncertainty outside of the observed data. The third and fourth column demonstrate that our approach can also be combined with priors based on Gaussian processes (GP) or random forests (RF).

2. Dirichlet Function Priors

Let $\mathcal{D} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^N$ be the training data consisting of N observed pairs of input data $\boldsymbol{x}_n \in \mathcal{X}$ and corresponding K-dimensional, one-hot class label vectors $\boldsymbol{y}_n \in \mathcal{Y}$. A neural network $\boldsymbol{\phi}$ with weights $\boldsymbol{w}$ defines a deterministic function $\boldsymbol{f}$ which maps an input $\boldsymbol{x} \in \mathcal{X}$ to an element $\boldsymbol{f}_{\boldsymbol{x}} \in \Delta^{K-1}$, where Δ^{K-1} denotes the $K-1$ simplex. More precisely, we write $\boldsymbol{f}_{\boldsymbol{x}} = \boldsymbol{\sigma}\left(\boldsymbol{\phi}\left(\boldsymbol{x}; \boldsymbol{w}\right)\right)$ and $\boldsymbol{y} \sim \mathrm{Cat}\left(\cdot | \boldsymbol{f}_{\boldsymbol{x}}\right)$, where $\boldsymbol{\sigma}$ is the softmax function. In conventional maximum likelihood (ML) training, the weights $\boldsymbol{w}$ are optimized by maximizing

$$\log \prod_{\mathcal{D}} \mathrm{Cat}\left(\boldsymbol{y} | \boldsymbol{f}_{\boldsymbol{x}}\right) = \sum_{\mathcal{D}} \sum_{k=1}^{K} y_k \log f_{\boldsymbol{x}k}, \tag{1}$$

where $\prod_{\mathcal{D}}$ and $\sum_{\mathcal{D}}$ denote $\prod_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}$ and $\sum_{(\boldsymbol{x},\boldsymbol{y})\in\mathcal{D}}$, and $\boldsymbol{\phi}$, $\boldsymbol{\sigma}$, and $\boldsymbol{w}$ are implicit in $\boldsymbol{f}$.

In Bayesian deep learning, $\boldsymbol{w}$ becomes a random variable and the goal is to estimate its posterior weight distribution. In general, exact inference is intractable and various approximations employ different parameterizations. In this paper, we assume that samples from a weight distribution $p(\boldsymbol{w})$ are available but an explicit density is not. This makes our method particularly generic and compatible with most BNNs and stochastic models.

Dirichlet Posterior Predictive Bayesian neural networks and stochastic deep learning models for classification typically make predictions by first sampling from a weight distribution $p(\boldsymbol{w})$, then predicting a softmax output for each weight sample, and finally averaging those predictions to produce a posterior categorical predictive.

2

However, taking the average *throws away* the epistemic uncertainty of the classifier. Instead, we interpret categorical predictions as samples from a *Dirichlet* distribution $p(\boldsymbol{f}_{\boldsymbol{x}})$, which allows us to leverage those samples to estimate a Dirichlet distribution over probability vectors instead of computing the average. Figure 2 shows how the Dirichlet density captures the variance of the samples. Later, we describe how we estimate this density via maximum likelihood. For now, let us assume that, given any input $\boldsymbol{x}$, the model predicts a corresponding Dirichlet distribution $p(\boldsymbol{f}_{\boldsymbol{x}})$, which is induced by the weight distribution $p(\boldsymbol{w})$.

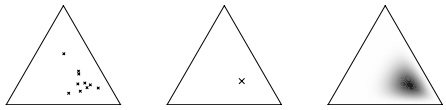

Figure 2: Treating model predictions as samples from a simplex (left), the mean reduction (middle) discards information which is present in the variance. A Dirichlet distribution (right) fitted to the same samples can capture the uncertainty with its density function.

Implicit Stochastic Processes The model's capability to predict a K-dimensional Dirichlet distribution $p(\boldsymbol{f}_{\boldsymbol{x}})$ for any $\boldsymbol{x} \in \mathcal{X}$ implicitly defines a stochastic process whose state space is the $K-1$ simplex Δ^{K-1} and whose index set is $\mathcal{X}$ (Ma et al., 2019). This stochastic process, despite using the Dirichlet distribution, is *not* a Dirichlet process (Teh, 2010). A Dirichlet process with index set $\mathcal{X}$ requires that any finite subset $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_L\} \subset \mathcal{X}$ follows a joint L-dimensional Dirichlet, whereas our implicit stochastic process defines a K-dimensional Dirichlet for each $\boldsymbol{x} \in \mathcal{X}$. For us, the finite collection $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_L\}$ would produce an element from Δ^{K-1} to the power of L and the whole implicit stochastic process could be rigorously defined as a random variable from Δ^{K-1} to the power of $\mathcal{X}$ (see Appendix B).

Function-Space Regularization To apply regularization in function space, we use the function-space evidence lower bound objective (fELBO) (Sun et al., 2019),

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{f} \sim q}\left[\log p(\mathcal{D}|\boldsymbol{f})\right] - D_{\mathrm{KL}}[q(\boldsymbol{f}|\boldsymbol{\theta}) \,||\, p(\boldsymbol{f})], \tag{2}$$

which resembles the conventional evidence lower bound objective (ELBO) (Jordan et al., 1999; Hoffman et al., 2013). To compute the likelihood term, we stay faithful to the backbone model and use M samples to estimate the expected categorical log-likelihood,

$$\mathbb{E}_{\boldsymbol{f} \sim q}\left[\log p(\mathcal{D}|\boldsymbol{f})\right] \approx \tfrac{1}{M} \textstyle\sum_{n,m=1}^{N,M} \log p\left(\boldsymbol{y}_n, \boldsymbol{f}_{\boldsymbol{x}_n}^{(m)}\right), \tag{3}$$

which is identical to the likelihood term in conventional ELBO optimization for BNNs. The novelty of our approach manifests in the KL term, which requires computing a function-space KL divergence (fKL) between stochastic processes. Sun et al. (2019) derived this divergence as the supremum over regular KL divergences evaluated at all possible finite sets $\boldsymbol{X} \subset \mathcal{X}$,

$$D_{\mathrm{KL}}[q \,||\, p] = \sup_{\boldsymbol{X} \subset \mathcal{X}, |\boldsymbol{X}| < \infty} D_{\mathrm{KL}}[q(\boldsymbol{f}_{\boldsymbol{X}}|\boldsymbol{\theta}) \,||\, p(\boldsymbol{f}_{\boldsymbol{X}})]. \tag{4}$$

However, the supremum is generally intractable because there are infinitely many possible finite measurement sets. A tractable approximation (Sun et al., 2019; Bruinsma et al., 2021) replaces the supremum with an expectation,

$$D_{\mathrm{KL}}[q \,||\, p] \approx \mathbb{E}_{\boldsymbol{S}}\, D_{\mathrm{KL}}[q(\boldsymbol{f}_{\boldsymbol{S}}|\boldsymbol{\theta}) \,||\, p(\boldsymbol{f}_{\boldsymbol{S}})], \tag{5}$$

where $\boldsymbol{S} \subset \mathcal{X}$ is a randomly sampled, finite measurement set of size L, which contains all the points that the stochastic processes are conditioned on. For us, this is the training data, but we can improve it further by adding unlabeled data (see Section 3).

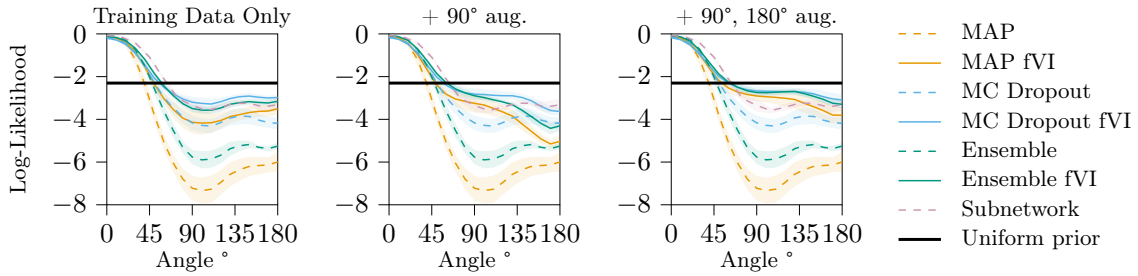

Figure 3: Comparison of rotated MNIST log-likelihood for models trained with fVI using different measurement sets. Colored lines denote the mean and shaded areas denote two standard deviations over 10 seeds. The subnetwork results are taken from Daxberger et al. (2021), who used a ResNet-18 rather than a MLP.

Dirichlet and KL Divergence Estimation Assuming $\boldsymbol{f}_{\boldsymbol{x}}^{(m)} \sim \mathrm{Dir}(\cdot|\boldsymbol{\alpha}_{\boldsymbol{x}})$, we compute a maximum likelihood estimate (MLE) of $\boldsymbol{\alpha}_{\boldsymbol{x}}$ using M samples $\boldsymbol{f}_{\boldsymbol{x}}^{(m)}$. To this end, we consider $\boldsymbol{\alpha}_{\boldsymbol{x}}$ in terms of two separate but dependent parameters: the Dirichlet mean $\bar{\boldsymbol{\alpha}}_{\boldsymbol{x}} = \boldsymbol{\alpha}_{\boldsymbol{x}}/z_{\boldsymbol{x}}$ and the Dirichlet precision $z_{\boldsymbol{x}}$. By matching the first moment of the empirical distribution of $\boldsymbol{f}_{\boldsymbol{x}}$, we obtain $\bar{\boldsymbol{\alpha}}_{\boldsymbol{x}} \approx \frac{1}{M}\sum_{m=1}^{M} \boldsymbol{f}_{\boldsymbol{x}}^{(m)}$. To estimate $z_{\boldsymbol{x}}$, we fix $\bar{\boldsymbol{\alpha}}_{\boldsymbol{x}}$ and employ a fast, iterative, quasi-Newton algorithm (Minka, 2000) using M predictive samples $\boldsymbol{f}_{\boldsymbol{x}}^{(1:M)} = \{\boldsymbol{f}_{\boldsymbol{x}}^{(1)}, \ldots, \boldsymbol{f}_{\boldsymbol{x}}^{(M)}\}$,

$$\left(z_{\boldsymbol{x}}^{(t+1)}\right)^{-1} = \left(z_{\boldsymbol{x}}^{(t)}\right)^{-1} + \left(z_{\boldsymbol{x}}^{(t)}\right)^{-2} \frac{\partial_{z_{\boldsymbol{x}}} \mathcal{L}(z_{\boldsymbol{x}}^{(t)})}{\partial^2_{z_{\boldsymbol{x}}} \mathcal{L}(z_{\boldsymbol{x}}^{(t)})}, \qquad \mathcal{L}(z_{\boldsymbol{x}}^{(t)}) \quad = \mathcal{L}_{\mathrm{Dir}}\left(\boldsymbol{f}_{\boldsymbol{x}}^{(1:M)}, \boldsymbol{\alpha}_{\boldsymbol{x}}^{(t)}\right), \tag{6}$$

where $z_{\boldsymbol{x}}^{(t)}$ and $\boldsymbol{\alpha}_{\boldsymbol{x}}^{(t)} = \bar{\boldsymbol{\alpha}}_{\boldsymbol{x}}/z_{\boldsymbol{x}}^{(t)}$ are the Dirichlet precision and concentration at iteration t, and $\mathcal{L}_{\mathrm{Dir}}$ is the Dirichlet log-likelihood $\log \prod_{m=1}^{M} \mathrm{Dir}\left(\boldsymbol{f}_{\boldsymbol{x}}^{(m)} \middle| \boldsymbol{\alpha}_{\boldsymbol{x}}\right)$. With $\bar{\boldsymbol{\alpha}}_{\boldsymbol{x}}$ and $z_{\boldsymbol{x}}$ estimated, $\boldsymbol{\alpha}_{\boldsymbol{x}} = \bar{\boldsymbol{\alpha}}_{\boldsymbol{x}}/z_{\boldsymbol{x}}$, $q(\boldsymbol{f}_{\boldsymbol{x}}|\boldsymbol{\theta}) = \mathrm{Dir}\left(\boldsymbol{f}_{\boldsymbol{x}}|\boldsymbol{\alpha}_{\boldsymbol{x}}\right)$ and we compute the KL divergence as

$$D_{\mathrm{KL}}[q||p] \approx \frac{1}{M} \sum_{l,m=1}^{L,M} \left(\log q\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)} \middle| \boldsymbol{\theta}\right) - \log p\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}\right)\right),$$

where $\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}$ is the m-th prediction of the model evaluated at the l-th measurement item $\boldsymbol{s}_l \in \boldsymbol{S}$, and $\log q(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}|\boldsymbol{\theta})$ and $\log p(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)})$ are the log-likelihood of $\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}$ under the variational Dirichlet posterior and Dirichlet prior respectively. Further details about optimization and prior specification are discussed in Appendix C.

3. Experiments

In this section, we present an empirical evaluation of our proposed approach to fVI-based regularization for classification tasks, comparing the performance of several models against their conventional training procedure. We first visualize the effects of a Dirichlet prior on a toy problem, and then perform image classification on the MNIST (LeCun et al., 2010), CIFAR10 and CIFAR100 (Krizhevsky, 2009) data sets. We use feedforward multilayer perceptrons (MLPs) or convolutional neural networks (CNNs), detailed at the respective sections. Metrics

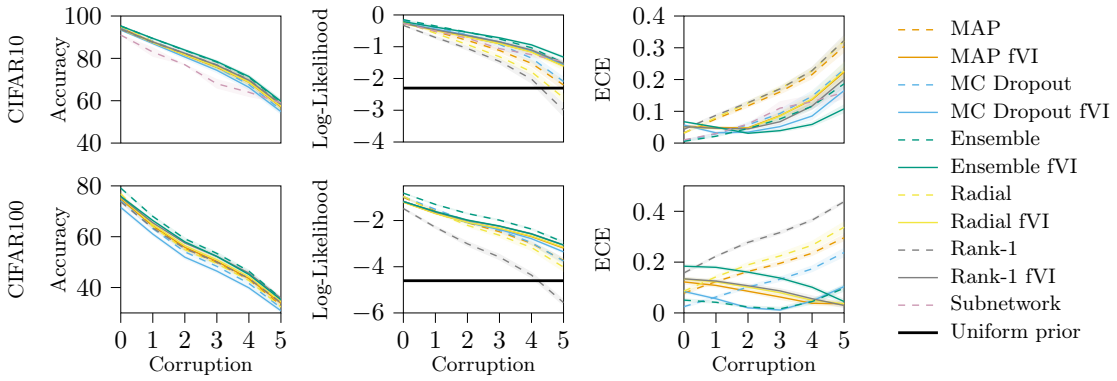

Figure 4: Image classification on corrupted CIFAR10 and CIFAR100. All models use a ResNet-18 architecture. For CIFAR10, there is a clear benefit of fVI priors over weight-space for log-likelihood. For CIFAR100, the higher label dimensionality results in stronger regularization from the uniform prior. This indicates prior specification requires more care for high-dimensional classification. Subnetwork results are taken from Daxberger et al. (2021).

include classification accuracy, log-likelihood (LLH) and expected calibration error (ECE) (Naeini et al., 2015), which estimates the calibration of accuracy versus confidence through binning the predicted class probabilities. Appendix E contains additional details.

Toy Problem To visualize the effects of function-space variational inference, we conducted a toy experiment with the Two Moons data set and MLP models. We used MAP, MC Dropout (Gal and Ghahramani, 2016), and deep ensemble (Lakshminarayanan et al., 2017) models, training each with their regular weight-space method, a uniform function prior, a GP function prior and a random forest function prior. Figure 1 shows that weight-space training leads to overconfident extrapolation, while our inference approach combined with Dirichlet function priors adequately increases predictive uncertainty outside of the observed data. While the weight-space approach learns a decision boundary that bisects the data, the function-space approaches learn richer boundaries which capture the data distribution more accurately and display properties that resemble the respective prior.

Rotated MNIST Following Ovadia et al. (2019), we train on the MNIST handwritten digit classification data set (LeCun et al., 2010) and evaluate on constructed test data with rotations of up to 180°, which simulates a challenging OOD scenario due to the absence of data augmentation. For this experiment, we used the same MLP models as for the toy problem. The log-likelihood between models is shown in Figure 3. In terms of classification error, weight-space inference and fVI yield the same performance. In terms of log-likelihood, fVI consistently outperforms their weight-space counterparts as the data becomes more OOD. Subnetwork linearized Laplace (Daxberger et al., 2021) is also reported as a competitive baseline, however, these results were obtained using a ResNet-18.

Assessing Measurement Set Design To illustrate the importance of the measurement set, we train the fVI models for rotated MNIST using three different measurement sets: the training data, additional 90° augmentation, and additional 90° and 180° augmentation.

While simply using the training data without rotations already outperforms the weight-space counterparts, a direct comparison in Figure 3 illustrates that performance can be further increased if an appropriate measurement set, i.e. example OOD data, is available. With the enriched measurement sets, the OOD performance move closer to that of the prior, indicating more accurate inclusion in the fELBO. Sets for greater OOD performance could be designed through manual data augmentation, unlabeled data, or synthetic data generation. Note, for all other image classification experiments, we use the training data as the measurement set.

Image Classification under Corruption We used the regular train splits of the CIFAR10 and CIFAR100 (Krizhevsky, 2009) as training data and their corrupted versions (Hendrycks and Dietterich, 2019) as OOD test data. CIFAR10 and CIFAR100 consist of natural color images of animals and vehicles. Their corrupted versions perturb the images at five increasing levels of severity by changing the brightness, contrast or saturation, or adding noise, blur or other artifacts, such that classification becomes more difficult. For this experiment, we ResNet-18 CNN models (He et al., 2016). In addition to the previous MAP, MC Dropout and deep ensemble model types, we also evaluate our fVI approach on Radial BNNs (Farquhar et al., 2020), as an effective variant of MFVI, and Rank-1 BNNs (Dusenberry et al., 2020), which combine ensembles and VI. In Appendix D, we investigate a similar setting where the corruptions are replaced by adversarial attacks of varying strength.

Figure 4 shows the results for CIFAR10 and CIFAR100 under corruption. The general trend is that the function-space prior frequently provides gains in OOD uncertainty quantification with only a small decrease in (uncorrupted) test performance. This trade-off between accuracy and robustness has been observed and discussed in the adversarial robustness setting (Tsipras et al., 2019; Yang et al., 2020) and it remains an open problem if and how both qualities can be achieved in practice. Moreover, the shared function-space prior resulted in remarkable consistency across models, compared to the variety seen in weight-space priors. Due to the predictive prior, the ECE was significantly higher for zero corruption, but this underfitting provided superior ECE when OOD. For CIFAR100, the higher prior regularization due to higher dimensionality (see Appendix F), resulted in reduced benefit over weight-space models compared to CIFAR10, with improved performance only evident at stronger corruptions.

4. Conclusion

We propose an approach to function-space regularization for deep Bayesian classification, which enables the use of Dirichlet predictive priors to improve uncertainty quantification. Our approach provides a generic view of prior work on Dirichlet-based classifiers with function-space regularization, and can be applied to a general class of BNNs and stochastic models without altering their underlying architectures and mechanisms. Experiments demonstrate that our approach generally outperforms the corresponding weight-space priors in terms of uncertainty quantification and adversarial robustness. Different measurement sets can trade-off scalability against OOD uncertainty quantification by extending the fKL evaluation beyond the training data. Future research should improve measurement sets for fVI, for example, by developing effective methods for constructing them to reflect the test distribution, e.g. through using data augmentation or unlabeled data.

References

Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U. Rajendra Acharya, Vladimir Makarenkov, and Saeid Nahavandi. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 2021.

Laurence Aitchison. A statistical theory of cold posteriors in deep neural networks. In *International Conference on Learning Representations*, 2021.

Naveed Akhtar and A. Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

M. S. Andersen, J. Dahl, and L. Vandenberghe. Cvxopt: A python package for convex optimization. URL www.cvxopt.org.

D. Barber and Christopher Bishop. Ensemble learning in bayesian neural networks. In *Generalization in Neural Networks and Machine Learning*, 1998.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International Conference on Machine Learning*, 2015.

Wessel Bruinsma, James Requeima, Andrew Y. K. Foong, Jonathan Gordon, and Richard E Turner. The gaussian neural process. In *Third Symposium on Advances in Approximate Bayesian Inference*, 2021.

David R. Burt, Sebastian W. Ober, Adrià Garriga-Alonso, and Mark van der Wilk. Understanding variational inference in function-space. In *Advances in Approximate Bayesian Inference*, 2021.

Erik Daxberger, Eric Nalisnick, James U Allingham, Javier Antoran, and Jose Miguel Hernandez-Lobato. Bayesian deep learning via subnetwork inference. In *International Conference on Machine Learning*, 2021.

Alexander G. de G. Matthews, James Hensman, Richard Turner, and Zoubin Ghahramani. On sparse variational methods and the kullback-leibler divergence between stochastic processes. In *International Conference on Artificial Intelligence and Statistics*, 2016.

Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *International Conference on Machine Learning*, 2011.

John S. Denker and Yann LeCun. Transforming neural-net output levels to probability distributions. In *Advances in Neural Information Processing Systems*, 1991.

Asen L Dontchev and R Tyrrell Rockafellar. *Implicit functions and solution mappings*, volume 543. Springer, 2009.

Michael W. Dusenberry, Ghassen Jerfel, Yeming Wen, Yi-An Ma, Jasper Snoek, Katherine Heller, Balaji Lakshminarayanan, and Dustin Tran. Efficient and scalable bayesian neural nets with rank-1 factors. In *International Conference on Machine Learning*, 2020.

Sebastian Farquhar, Michael Osborne, and Yarin Gal. Radial bayesian neural networks: Beyond discrete support in large-scale bayesian deep learning. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Angelos Filos, Sebastian Farquhar, Aidan N. Gomez, Tim G. J. Rudner, Zachary Kenton, Lewis Smith, Milad Alizadeh, Arnoud de Kroon, and Yarin Gal. Benchmarking bayesian deep learning with diabetic retinopathy diagnosis. https://github.com/OATML/bdl-benchmarks, 2019.

Vincent Fortuin. Priors in bayesian deep learning: A review, 2021.

Yarin Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, 2016.

Yarin Gal, Riashat Islam, and Zoubin Ghahramani. Deep bayesian active learning with image data. In *International Conference on Machine Learning*, 2017.

M.N. Gibbs and D.J.C. Mackay. Variational gaussian process classifiers. *IEEE Transactions on Neural Networks*, 2000.

Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2014.

Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

Alex Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems*, 2011.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

James Hensman, Alexander Matthews, and Zoubin Ghahramani. Scalable variational gaussian process classification. In *International Conference on Artificial Intelligence and Statistics*, 2015.

Daniel Hernandez-Lobato and Jose Miguel Hernandez-Lobato. Scalable gaussian process classification via expectation propagation. In *International Conference on Artificial Intelligence and Statistics*, 2016.

José Miguel Hernández-Lobato and Ryan P. Adams. Probabilistic backpropagation for scalable learning of bayesian neural networks. In *International Conference on Machine Learning*, 2015.

Marius Hobbhahn, Agustinus Kristiadi, and Philipp Hennig. Fast predictive uncertainty for classification with bayesian deep networks, 2021.

Matthew D. Hoffman, David M. Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 2013.

Alexander Immer, Maciej Korzepa, and Matthias Bauer. Improving predictions of bayesian neural nets via local linearization. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Pavel Izmailov, Alexander Novikov, and Dmitry Kropotov. Scalable gaussian processes with billions of inducing inputs via tensor train decomposition. In *International Conference on Artificial Intelligence and Statistics*, 2018.

Taejong Joo, Uijung Chung, and Min-Gwan Seo. Being bayesian about categorical probability. In *International Conference on Machine Learning*, 2020.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Machine Learning*, 1999.

Mohammad Emtiyaz E Khan, Alexander Immer, Ehsan Abedi, and Maciej Korzepa. Approximate inference turns deep networks into gaussian processes. In *Advances in Neural Information Processing Systems*, 2019.

D. P. Kingma and L. J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Durk P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, 2015.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, 2017.

Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *ATT Labs*, 2010.

Zhan Wei Lim, Mong Li Lee, Wynne Hsu, and Tien Yin Wong. Building trust in deep learning system towards automated disease detection. In *AAAI Conference on Artificial Intelligence*, 2019.

Chao Ma, Yingzhen Li, and Jose Miguel Hernandez-Lobato. Variational implicit processes. In *International Conference on Machine Learning*, 2019.

David J. C. MacKay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3), 1992.

Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In *Advances in Neural Information Processing Systems*, 2018.

Andrey Malinin and Mark Gales. Reverse kl-divergence training of prior networks: Improved uncertainty and adversarial robustness. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, 2019.

Andrey Malinin, Bruno Mlodozeniec, and Mark Gales. Ensemble distribution distillation. In *International Conference on Learning Representations*, 2020.

Rowan McAllister, Yarin Gal, Alex Kendall, Mark van der Wilk, Amar Shah, Roberto Cipolla, and Adrian Weller. Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning. In *International Joint Conference on Artificial Intelligence*, 2017.

Dimitrios Milios, Raffaello Camoriano, Pietro Michiardi, Lorenzo Rosasco, and Maurizio Filippone. Dirichlet-based gaussian processes for large-scale calibrated classification. In *Advances in Neural Information Processing Systems*, 2018.

Thomas Minka. Estimating a dirichlet distribution, 2000.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *AAAI Conference on Artificial Intelligence*, 2015.

Radford M. Neal. *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, CAN, 1995.

Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, 2018.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, 2019.

Pingbo Pan, Siddharth Swaroop, Alexander Immer, Runa Eschenhagen, Richard E Turner, and Mohammad Emtiyaz Khan. Continual deep learning by functional regularisation of memorable past. *Advances in Neural Information Processing Systems*, 2020.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

Tim Pearce, Felix Leibfried, Alexandra Brintrup, Mohamed Zaki, and Andy Neely. Uncertainty in neural networks: Approximately bayesian ensembling. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Nicholas G. Polson, James G. Scott, and Jesse Windle. Bayesian inference for logistic models using pólya–gamma latent variables. *Journal of the American Statistical Association*, 2013.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2005.

Thomas W Rauber, Tim Braun, and Karsten Berns. Probabilistic distance measures of the Dirichlet and Beta distributions. *Pattern Recognition*, 2008.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *International Conference on Learning Representations*, 2018.

Geoffrey Roeder, Yuhuai Wu, and David Duvenaud. Sticking the landing: Simple, lower-variance gradient estimators for variational inference. In *Advances in Neural Information Processing Systems*, 2017.

Tim G. J. Rudner, Zonghao Chen, and Yarin Gal. Rethinking function-space variational inference in bayesian neural networks. In *Advances in Approximate Bayesian Inference*, 2021.

Hugh Salimbeni, Stefanos Eleftheriadis, and James Hensman. Natural gradients in practice: Non-conjugate variational inference in gaussian process models. In *International Conference on Artificial Intelligence and Statistics*, 2018.

Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. In *Advances in Neural Information Processing Systems*, 2018.

Murat Sensoy, Maryam Saleki, Simon Julier, Reyhan Aydogan, and John Reid. Misclassification risk and uncertainty quantification in deep classifiers. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, 2021.

Jiaxin Shi, Shengyang Sun, and Jun Zhu. A spectral approach to gradient estimation for implicit distributions. In *International Conference on Machine Learning*, 2018.

Jiaxin Shi, Mohammad Emtiyaz Khan, and Jun Zhu. Scalable training of inference networks for Gaussian-process models. In *International Conference on Machine Learning*, 2019.

Shengyang Sun, Guodong Zhang, Jiaxin Shi, and Roger Grosse. Functional variational bayesian neural networks. In *International Conference on Learning Representations*, 2019.

T. Tao. *An Introduction to Measure Theory*. Graduate studies in mathematics. American Mathematical Society, 2013. ISBN 9781470409227.

Yee Whye Teh. *Dirichlet Process*. Springer US, 2010.

Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.

Ziyu Wang, Tongzheng Ren, Jun Zhu, and Bo Zhang. Function space particle optimization for Bayesian neural networks. In *International Conference on Learning Representations*, 2019.

Joe Watson, Jihao Andreas Lin, Pascal Klink, Joni Pajarinen, and Jan Peters. Latent derivative bayesian last layer networks. In *International Conference on Artificial Intelligence and Statistics*, 2021a.

Joe Watson, Jihao Andreas Lin, Pascal Klink, and Jan Peters. Neural linear models with functional gaussian process priors. In *Advances in Approximate Bayesian Inference*, 2021b.

Yeming Wen, Dustin Tran, and Jimmy Ba. Batchensemble: an alternative approach to efficient ensemble and lifelong learning. In *International Conference on Learning Representations*, 2020.

Florian Wenzel, Théo Galy-Fajou, Christan Donner, Marius Kloft, and Manfred Opper. Efficient gaussian process classification using pólya-gamma data augmentation. In *AAAI Conference on Artificial Intelligence*, 2019.

Florian Wenzel, Kevin Roth, Bastiaan S Veeling, Jakub Świątkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. How good is the bayes posterior in deep neural networks really? In *International Conference on Machine Learning*, 2020.

Andrew Gordon Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. In *Advances in Neural Information Processing Systems*, 2020.

Yao-Yuan Yang, Cyrus Rashtchian, Hongyang Zhang, Russ R. Salakhutdinov, and Kamalika Chaudhuri. A closer look at accuracy vs. robustness. In *Advances in Neural Information Processing Systems*, 2020.

Guodong Zhang, Shengyang Sun, David Duvenaud, and Roger Grosse. Noisy natural gradient as variational inference. In *International Conference on Machine Learning*, 2018.

Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical stochastic gradient mcmc for bayesian deep learning. In *International Conference on Learning Representations*, 2020.

Appendix A. Related Work

In this section, we summarize related work on Bayesian classification and function-space inference, and discuss previous research which is of particular relevance to our work.

Bayesian Classification Compared to regression, classification is non-trivial for Bayesian methods due to the nonlinear link function required to predict the class labels. As a result, closed-form Bayesian models, such as Gaussian processes (GP), require approximate inference methods such as the Laplace approximation (Rasmussen and Williams, 2005), variational inference (Gibbs and Mackay, 2000; Hensman et al., 2015; Salimbeni et al., 2018; Izmailov et al., 2018), and expectation propagation (Hernandez-Lobato and Hernandez-Lobato, 2016). The Pólya-Gamma data augmentation trick (Polson et al., 2013) has enabled scalable closed-form variational training of sparse Gaussian process classifiers (Wenzel et al., 2019). Gaussian processes have also been used with a Dirichlet predictive using a log-normal approximation (Milios et al., 2018).

Classification with Bayesian neural networks is possible through a wide range of approximate inference methods, including Markov chain Monte Carlo (Neal, 1995; Zhang et al., 2020), (mean-field) variational inference (MFVI) (Graves, 2011; Blundell et al., 2015; Zhang et al., 2018), Laplace approximations (MacKay, 1992; Denker and LeCun, 1991; Ritter et al., 2018; Khan et al., 2019; Immer et al., 2021; Daxberger et al., 2021), ensembles (Lakshminarayanan et al., 2017; Osband et al., 2018; Barber and Bishop, 1998; Pearce et al., 2020), expectation propagation (Hernández-Lobato and Adams, 2015) and Monte Carlo dropout (Gal and Ghahramani, 2016; Kingma et al., 2015). Radial BNNs (Farquhar et al., 2020) are motivated as a practical alternative to MFVI BNNs that uses Gaussian weight priors and posteriors. By sampling weights in a radial fashion, they avoid the pathologies encountered when sampling high-dimensional Gaussian distributions. Rank-1 BNNs (Dusenberry et al., 2020) combine ensembles and weight priors. Using the shared BatchEnsemble structure (Wen et al., 2020) and Rank-1 covariance parameterizations, Rank-1 BNNs have a scalable memory requirement. Alternatively, the Laplace bridge (Hobbhahn et al., 2021) approximately maps a Dirichlet predictive density backwards through the softmax into a latent Gaussian predictive. A Gaussian-predictive BNN can then be trained using this latent approximation.

Alternative methods avoid propagating uncertainties by predicting Dirichlet concentrations directly with deep neural networks. Prior networks (Malinin and Gales, 2018, 2019) require categorical labels to be converted to Dirichlets, and resembles fVI as the objective consists of two KL divergences, for in- and outside the data distribution respectively. They can be used to distill a trained ensemble into a single model (Malinin et al., 2020). Similarly, belief matching (Joo et al., 2020) converts training labels to Dirichlets using Bayes rule. This method can also be viewed as fVI where the measurement set is the training data. Another method converts the training labels to categorical probabilities and uses a Bayes risk objective with KL regularization against a function-space prior (Sensoy et al., 2018, 2021). Compared to these methods, we introduce generic function-space regularization that allows us to use any BNN or stochastic model with the conventional categorical likelihood, avoiding the need to design networks and data representations that facilitate a model-specific training approach. A longer discussion and comparison is provided later in this section.

Function-Space Variational Inference Function-space variational inference generalizes conventional variational inference over finite weight distributions to inference over stochastic processes, which entails difficulties because the standard KL divergence between finite-dimensional probability distributions becomes an infinite-dimensional fKL divergence between stochastic processes. Gaussian processes (Rasmussen and Williams, 2005) are a rare exception where analytically tractable function-space inference is possible. Sparse GPs may be viewed as variational inference over functions (de G. Matthews et al., 2016), minimizing the fKL from its exact posterior via inducing points.

In functional variational BNNs (fBNNs), Sun et al. (2019) derive the fKL as a supremum over an infinite set of finite, marginal KL divergences. Burt et al. (2021) showed that this fKL can be infinite under certain conditions, for example when considering the divergence between two BNNs with different architectures. Sun et al. (2019) replace the intractable supremum by an expectation based on finite measurement sets. We explain this in more detail in Section 2 because our approach is based on this approximation. Further, Sun et al. (2019) they used a trained GP as explicit function-space prior, which can be viewed as a form of empirical Bayes, and employ the spectral Stein gradient estimator (SSGE) (Shi et al., 2018) to enable implicit function priors. Similar approaches take a mirror descent view for batch training (Shi et al., 2019).

Variational implicit processes (Ma et al., 2019) interpret parametric models with stochastic parameters as stochastic processes and introduce a wake-sleep procedure for inference in the regression setting with Gaussian likelihoods. Our generic view of Bayesian neural networks and other stochastic models can be formally understood within their stochastic process perspective of parametric models, although our inference approach is unrelated (see Appendix B).

Neural linear models have also been used with fVI, because closed-form Gaussian predictive distributions allow explicit computation of gradients (Watson et al., 2021a,b). Concurrent work (Rudner et al., 2021) has also adopted fVI for classification, by linearizing a neural network about a Gaussian weight distribution to estimate the fKL. This model works with a Gaussian (latent) predictive prior and posterior which loses the intuitive aspect of function-space priors. Moreover, the linearization requires computation of the Jacobian of the neural network function with respect to the model parameters, for which the memory requirement scales with the number of model parameters and outputs. Wang et al. (2019) propose particle optimization methods using finite function representations to learn a particle representation of the function-space posterior through the gradient flow of the log posterior. Function-space inference is also an attractive approach to continual learning (Pan et al., 2020).

Prior Networks Malinin and Gales (2018, 2019) use a neural network with parameters $\boldsymbol{\theta}$ to directly predict the concentration parameters α_c of a Dirichlet distribution $\mathrm{p}(\boldsymbol{\mu}|\boldsymbol{x};\boldsymbol{\theta})$, given input $\boldsymbol{x}$, which is distinct from our approach of estimating a posterior Dirichlet from M categorical predictions. This model is not a Bayesian neural network in practice, as only point estimates for the weights are learned. To ensure $\alpha_c > 0$, an element-wise exponential operation is applied as the final layer of the neural network. Additionally, Prior Networks minimize an optimization objective consisting of two separate KL divergences, representing in- and out-of-distribution data respectively,

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{\mathrm{p_{in}}(\boldsymbol{x})}[D_{\mathrm{KL}}[\mathrm{Dir}(\boldsymbol{\mu}|\hat{\boldsymbol{\alpha}}) \mid\mid \mathrm{p}(\boldsymbol{\mu}|\boldsymbol{x};\boldsymbol{\theta})]] + \mathbb{E}_{\mathrm{p_{out}}(\boldsymbol{x})}[D_{\mathrm{KL}}[\mathrm{Dir}(\boldsymbol{\mu}|\tilde{\boldsymbol{\alpha}}) \mid\mid \mathrm{p}(\boldsymbol{\mu}|\boldsymbol{x};\boldsymbol{\theta})]]. \quad (7)$$

The first expectation $\mathbb{E}_{\mathrm{p_{in}}}$ accounts for the actual learning, i.e. fitting the training data, whereas the second expectation $\mathbb{E}_{\mathrm{p_{out}}}$ is supposed to regularize the model by matching a prior distribution. Accordingly, the first expectation is computed for the training data and can be compared to the expected log-likelihood term in our approach. Instead of maximizing the categorical log-likelihood of M observations, Prior Networks construct Dirichlet targets by smoothing categorical ground truth labels to define the Dirichlet mean and setting the precision as a hyperparameter during training. Although we also apply 'label smoothing' to the predictions, it is for numerical reasons and not for the construction of target distributions from labels. Additionally, Prior Networks treat the precision of their constructed *target* distribution as a hyperparameter, whereas we estimate the Dirichlet precision of our *predicted* variational posterior distribution via maximum likelihood. The second expectation is computed for OOD data and resembles the fKL term in our approach, where the OOD data is used as measurement set. In contrast to Prior Networks, our more general approach also allows the training data or mixtures of training data and OOD data as measurement sets, whereas Prior Networks explicitly compute their second expectation for OOD data only. Furthermore, both Prior Network expectations consider the KL divergence from the neural network predictive distribution (right) to the target or prior distribution (left), whereas, in our approach, and variational inference in general, the KL divergence from the prior distribution (right) to the variational posterior (left) is considered.

Belief Matching Joo et al. (2020) assume a Dirichlet prior which, together with the categorical ground truth class labels, define a target Dirichlet posterior. A neural network is used to directly predict concentration parameters of a Dirichlet posterior $q_{\boldsymbol{z}|\boldsymbol{x}}$ by replacing the final softmax layer with an element-wise exponential operation. To learn the target posterior, belief matching maximizes

$$l_{EB}(\boldsymbol{y}, \alpha^{\boldsymbol{W}}(\boldsymbol{x})) = \mathbb{E}_{q_{\boldsymbol{z}|\boldsymbol{x}}}[\log \boldsymbol{z}_{\boldsymbol{y}}] - D_{\mathrm{KL}}[q^{\boldsymbol{W}}_{\boldsymbol{z}|\boldsymbol{x}} \mid\mid p_{\boldsymbol{z}|\boldsymbol{x}}], \tag{8}$$

where $\mathbb{E}_{q_{\boldsymbol{z}|\boldsymbol{x}}}[\log \boldsymbol{z}_{\boldsymbol{y}}]$ is the expected log-likelihood of the training data and $D_{\mathrm{KL}} q^{\boldsymbol{W}}_{\boldsymbol{z}|\boldsymbol{x}} p_{\boldsymbol{z}|\boldsymbol{x}}$ is the KL divergence between the predicted Dirichlet posterior and the Dirichlet prior. Therefore, their objective matches our fELBO objective (Equation (2)) except for two differences: Firstly, belief matching computes both the expected log-likelihood and the KL divergence with respect to their single, directly predicted Dirichlet distribution, whereas we evaluate them as arithmetic averages of M stochastic categorical model outputs. Secondly, belief matching does not recognize the function-space aspect and instead only considers evaluation of the KL divergence using the training data, which resembles the fKL in our case where the measurement set is constrained to be the training data.

Evidential Deep Learning Sensoy et al. (2018) directly predict the concentration parameter of a Dirichlet distribution by using a neural network with ReLU activations as final layer to assert the positive constraint. Additionally, a loss function is derived via type-II maximum likelihood by integrating over a Dirichlet prior and the sum of squares between target labels $\boldsymbol{y}_i$ and predicted probabilities $\boldsymbol{p}_i$. Furthermore, a regularizing KL divergence term is added, resulting in a total loss function,

$$\mathcal{L}(\Theta) = \sum_{i=1}^{N} \left(\sum_{j=1}^{K} \left((y_{ij} - \hat{p}_{ij})^2 + \frac{\hat{p}_{ij}(1-\hat{p}_{ij})}{S_i + 1} \right) + \lambda_t D_{\mathrm{KL}}[\mathrm{Dir}(\boldsymbol{p}_i|\tilde{\boldsymbol{\alpha}}_i) \mid\mid \mathrm{Dir}(\boldsymbol{p}_i|\boldsymbol{1})] \right), \tag{9}$$

where y_{ij} are individual 0-1 target labels, $\hat{p}_{ij}$ are components of the predicted Dirichlet mean, S_i is the predicted Dirichlet precision, $\tilde{\boldsymbol{\alpha}}_i$ is the predicted Dirichlet concentration parameter, $\mathbf{1}$ is a vector of ones and λ_t is a annealing coefficient for optimization. The first part of their loss is responsible for fitting the training data and can thus be compared to the maximum likelihood objective in Section 2. The ML objective can be derived from the categorical log-likelihood via type-I maximum likelihood, whereas their objective is derived by minimizing the sum of squares via type-II maximum likelihood. The second part of their loss resembles the fKL in our approach. However, they only evaluate the KL divergence for the training data and explicitly consider the uniform Dirichlet distribution with concentration $\mathbf{1}$. Therefore, their KL divergence regularization term is a special case of our proposed regularization with the measurement set being the training data and the prior being the uniform Dirichlet distribution with precision K. For both parts, a major difference between evidential deep learning and our approach is the realization of the predictive Dirichlet distribution. Evidential deep learning directly predicts Dirichlet concentration parameters, whereas we use M predictions to estimate a Dirichlet distribution via maximum likelihood.

Experimental Comparison We compare our MAP and MAP fVI models to Belief Matching and Prior Networks, which both demonstrated scalability to ResNet models. To reproduce their results, we used the official open-source implementations [2] [3].

Figure 5 illustrates the test accuracy, log-likelihood, and expected calibration error for the corrupted CIFAR10 image classification task (see Section 3). We trained the models using the same procedure and hyperparameters described in Section E. The Belief Matching model corresponds closely to the MAP fVI model, as both the objectives and models are similar. Unfortunately, we were not able to reproduce the Prior Networks performance described in the paper (Malinin and Gales, 2019), neither with their listed hyperparameters (Table 4, (Malinin and Gales, 2019)) or the hyperparameters used in Appendix E. It is uncertain whether this is due to the model, implementation bugs or unrecorded hyperparameters [4].

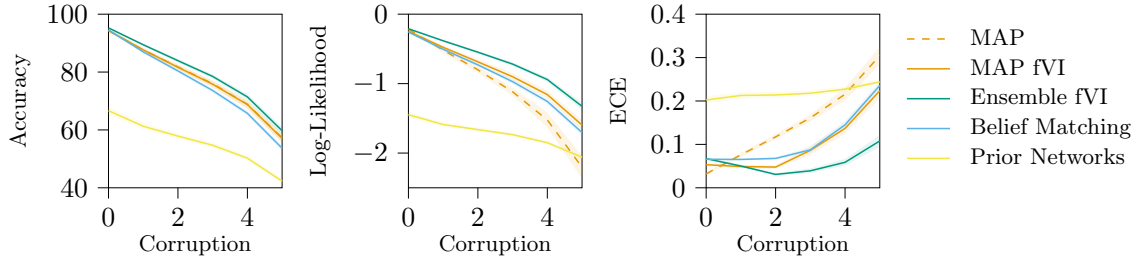

Figure 5: Comparing Prior Networks and Belief Matching against our MAP, MAP fVI and Ensemble fVI models on the corrupted CIFAR10 task. MAP fVI and Belief Matching achieve comparable performance, while our best model, Ensemble fVI, outperforms Belief Matching by a considerable amount.

2. github.com/tjoo512/belief-matching-framework
3. github.com/KaosEngineer/PriorNetworks
4. The authors did not respond to personal correspondence regarding this matter.

Appendix B. Deep Stochastic Classifiers as Implicit Stochastic Processes

An implicit stochastic process (Ma et al., 2019) is an infinite set of random variables $\boldsymbol{f}$, such that any finite subset $\boldsymbol{f}_{\boldsymbol{x}_{1:L}} = \{\boldsymbol{f}_{\boldsymbol{x}_1}, \boldsymbol{f}_{\boldsymbol{x}_2}, ..., \boldsymbol{f}_{\boldsymbol{x}_L}\}$ with $L \in \mathbb{N}$ has a joint distribution which is implicitly defined as

$$\boldsymbol{w} \sim p(\boldsymbol{w}), \qquad \boldsymbol{f}_{\boldsymbol{x}_l} = \boldsymbol{g}(\boldsymbol{x}_l, \boldsymbol{w}), \qquad \forall\, \boldsymbol{x}_l \in \mathcal{X}, \qquad 1 \leq l \leq L,$$

where the classifiers which we consider, such as BNNs and other stochastic neural networks, are instantiated as a feedforward or convolutional neural networks with stochastic weights, such that $\boldsymbol{g}(\boldsymbol{x}_l, \boldsymbol{w}) = \boldsymbol{\sigma}\left(\boldsymbol{\phi}\left(\boldsymbol{x}; \boldsymbol{w}\right)\right)$ in our case. In practice, the implicit stochastic process interpretation of BNNs and stochastic models entails that we consider the weight distribution $p(\boldsymbol{w})$ in a parameterized form $q_{\boldsymbol{\theta}}(\boldsymbol{w})$ with parameters $\boldsymbol{\theta}$ which we wish to optimize. The actual form and meaning of $\boldsymbol{\theta}$ depends on the specific neural network architecture, encoded through $\boldsymbol{\phi}$. Table 1 lists mathematical expressions to describe $\boldsymbol{\theta}$ for various models. Different model-specific parameterizations $q_{\boldsymbol{\theta}}(\boldsymbol{w})$ induce the same generic variational posterior over functions $q(\boldsymbol{f}|\boldsymbol{\theta})$ which allows us to implement function-space regularization independent of the specific model.

Formally, the stochastic process $\boldsymbol{f}$ is defined on the sample space Ω with an index set $\mathcal{X}$ defined by the data type, such that $\boldsymbol{f} : \mathcal{X} \times \Omega \to \Delta^{K-1}$, where Δ^{K-1} is the state space, which is the $K-1$ simplex. A random variable $\boldsymbol{f}(\boldsymbol{x}) : \Omega \to \Delta^{K-1}$ can be defined for each $\boldsymbol{x} \in \mathcal{X}$ and we write $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{f}_{\boldsymbol{x}}$. Kolmogorov's extension theorem (Tao, 2013) guarantees the existence of a stochastic process $\boldsymbol{f}$ if for each $L \in \mathbb{N}$ the finite marginal joint distributions $p_{1:L}(\boldsymbol{f}_{\boldsymbol{x}_{1:L}})$, where $\boldsymbol{f}_{\boldsymbol{x}_{1:L}} = \{\boldsymbol{f}_{\boldsymbol{x}_1}, \ldots, \boldsymbol{f}_{\boldsymbol{x}_L}\}$, satisfy exchangeability and consistency.

Exchangeability For any permutation π of $1, \ldots, L$, $p_{\pi(\boldsymbol{x}_{1:L})}(\boldsymbol{f}_{\pi(x_{1:L})}) = p_{\boldsymbol{x}_{1:L}}(\boldsymbol{f}_{\boldsymbol{x}_{1:L}})$. This requires that the process behavior is invariant to the order of inputs. For a feedforward neural network, this is satisfied because the respective predictions do not change if the order of inputs changes.

Consistency For any $1 \leq L' < L$, $p_{1:L'}(\boldsymbol{f}_{\boldsymbol{x}_{1:L'}}) = \int p_{1:L}(\boldsymbol{f}_{\boldsymbol{x}_{1:L}})\, \mathrm{d}\boldsymbol{f}_{\boldsymbol{x}_{L'+1:L}}$. This requires that future evaluations are independent of past evaluations. For a feedforward neural network, this is satisfied because predictions do not depend on previous predictions.

Table 1: Summary of stochastic (and deterministic) representations of weights $\boldsymbol{w}$ which correspond to popular BNN approaches, applying either to the whole network or per layer. Note $\boldsymbol{\Sigma}_\theta$ is typically factorized in practice. While we use Radial BNNs rather than MFVI, we include it for completeness. Rank-1 references specifically the Gaussian realization.

Model	Parameterization for $\boldsymbol{w} \sim q_{\boldsymbol{\theta}}(\cdot)$	Scope
MAP	$\delta(\boldsymbol{w} - \boldsymbol{w}_{\boldsymbol{\theta}})$	Network
MFVI	$\mathcal{N}(\boldsymbol{\mu}_{\boldsymbol{\theta}}, \boldsymbol{\Sigma}_{\boldsymbol{\theta}})$	Layer
Radial	$\boldsymbol{w} \sim \boldsymbol{\mu}_\theta + \sqrt{\boldsymbol{\Sigma}_\theta} \odot \hat{\boldsymbol{\epsilon}} \cdot \lvert r \rvert,\ \hat{\boldsymbol{\epsilon}} = \boldsymbol{\epsilon}/\lvert\boldsymbol{\epsilon}\rvert,\ \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}),\ r \sim \mathcal{N}(0, 1)$	Layer
MC Dropout	$\boldsymbol{w} \sim \bar{\boldsymbol{w}}_{\boldsymbol{\theta}} \odot \boldsymbol{b},\ \ b_i \sim \text{Bernoulli}(p_{\text{dropout}})$	Layer
Ensemble	$\frac{1}{M}\sum_m \delta(\boldsymbol{w} - \boldsymbol{w}_{\boldsymbol{\theta}}^m)$	Network
Rank-1	$\frac{1}{M}\sum_m \delta(\boldsymbol{w} - \boldsymbol{w}_{\boldsymbol{\theta}}^m),\ \boldsymbol{w}_{\boldsymbol{\theta}}^m \sim \bar{\boldsymbol{w}}_{\boldsymbol{\theta}} \odot \delta\boldsymbol{w}_{\boldsymbol{\theta}}^m,\ \delta\boldsymbol{w}_{\boldsymbol{\theta}}^m \sim \mathcal{N}(\boldsymbol{\mu}_\theta^m, \boldsymbol{v}_\theta^m \boldsymbol{u}_\theta^{m\top})$	Layer

Appendix C. Optimization and Prior Specification

Here, we provide additional methodological details which were omitted in Section 2.

Optimization We optimize $\boldsymbol{\theta}$ using backpropagation on the fELBO objective. For some models, such as deep ensembles, this is standard gradient descent, while for others, such as MFVI, the reparameterization trick is required. In cases where only a single sample is available ($M = 1$), such as MAP models or MC Dropout with single forward pass, we set the precision $z_{\boldsymbol{x}}$ to the size of the training data. When computing gradients, we assume that $\boldsymbol{\alpha}_{\boldsymbol{x}}$ does not depend on $\boldsymbol{\theta}$. This serves the practical purpose of pruning the Dirichlet MLE from the computation graph, speeding up computation and evoking expectation maximization-style inference. In Appendix G, we connect this approximation to the pathwise gradient estimator (Roeder et al., 2017), which can in fact be lower variance than the total gradient and lead to faster convergence in terms of computation time. In terms of mini-batching, like Sun et al. (2019), we divide both the batched expected log-likelihood and the fKL by the mini-batch size for numerical stability. Consequently, the fKL weight in the total ELBO depends on the mini-batch size, which is theoretically undesirable, but, in practice, KL divergence scaling in (weight-space) variational inference is a topic of active debate (Wenzel et al., 2020; Aitchison, 2021) and is frequently scaled or annealed for numerical reasons (Dusenberry et al., 2020).

Prior Specification We require the Dirichlet function prior $p(\boldsymbol{f})$ to be defined as a regular K-dimensional Dirichlet distribution $p(\boldsymbol{f}_{\boldsymbol{x}})$ at each input location $\boldsymbol{x}$. For most experiments in this paper, we choose $p(\boldsymbol{f}_{\boldsymbol{x}}) = \mathrm{Dir}(\cdot \mid \boldsymbol{\beta})$ with $\boldsymbol{\beta} = \mathbf{1}$, which is a constant uniform Dirichlet distribution with precision K. One might criticize that this constant uniform prior is factorized and does not encode any correlations between input locations. However, the posterior will still be correlated among input locations due to the neural network. As we learn a variational posterior over functions by adapting the implicit neural network weights, the neural network function induces smoothness in the variational posterior despite the factorized prior. A similar scenario arises in conventional weight-space variational inference with factorized Gaussian priors (MFVI): The weights are also not correlated by the prior yet the neural network function enables learning. In practice, it is often difficult to define correlated priors in domains with high-dimensional inputs, such as images. In a toy problem, we show that it is also possible to use more sophisticated priors based on, for example, GPs or random forests. Nonetheless, the constant uniform prior is simple, scalable, intuitive to understand, yet effective (see Section 3).

Appendix D. Image Classification under Adversarial Attacks

Despite the success of CNNs in computer vision, adversarial attacks are one of the biggest risks when it comes to practical applications (Akhtar and Mian, 2018). We evaluate the robustness of fVI compared to standard weight-space prior approaches on the CIFAR10 and CIFAR100 data using the fast gradient sign method (FGSM) (Goodfellow et al., 2014). Figure 6 compares the accuracy and the log-likelihood of the test data with increasing amounts of perturbation, ranging from $\epsilon = 0$ (no attack) to $\epsilon = 0.3$. Although both weight-space and function-space models lose their classification accuracy when the FGSM attack is introduced, the fVI models only suffer small decreases in log-likelihood, whereas the weight-space LLH performance drops significantly. We also observe the accuracy vs robustness trade-off in the

fVI models. We attribute this behavior to the quality of the uncertainty quantification at the decision boundary. While both approaches have brittle boundaries due to the nature of CNNs, the predictive uncertainty at these decision boundaries is richer for fVI.

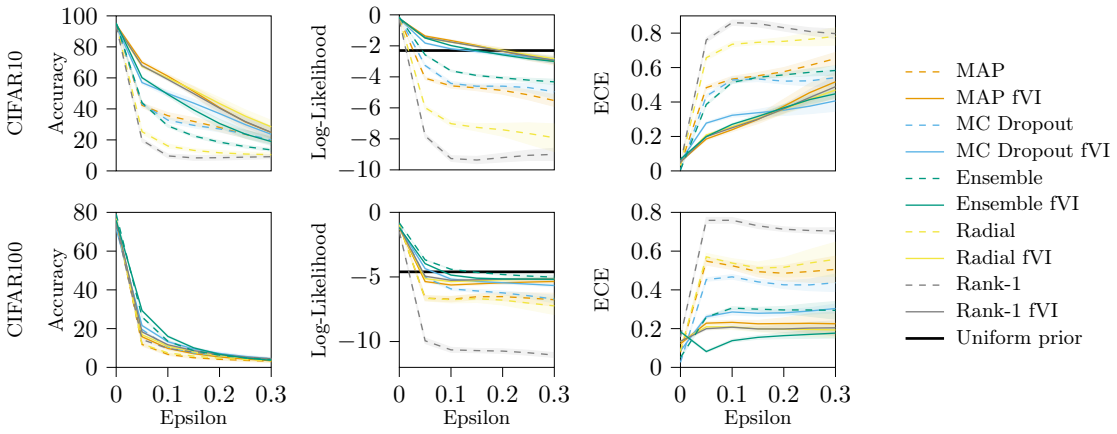

Figure 6: Metrics for adversarial examples on CIFAR10 (top) and CIFAR100 (bottom). All models use a ResNet-18 architecture. For CIFAR10, there are significant benefits of fVI over weight-space approaches across all metrics. For CIFAR100, the fVI benefits are still evident, but the higher label dimensionality results in stronger regularization from the uniform prior. As a result, the weight-space ensembles achieve slightly better performance over all epsilons.

Appendix E. Implementation Details and Computational Complexity

We implemented all models using the `PyTorch` library (Paszke et al., 2019) and all experiments were conducted using a i5-6600K CPU, a GTX1070 GPU and a GTX2080 GPU with less than 300 hours of total runtime.

The Two Moons data was generated by the `make_moons` function from the `scikit-learn` library using 100 samples, 0.2 noise and random state 456, the `PyTorch` manual seed was set to 123. For this toy problem, all models were MLPs with two hidden layers consisting of 25 hidden units each, bias terms enabled and ReLU activation. For the Dropout models, the dropout rate was set to 0.2 and for the Ensemble models, we used 10 members per ensemble. All models were trained for 1000 epochs at a learning rate of 0.005 using the Adam optimizer (Kingma and Ba, 2015) with default parameters. The measurement set for the KL divergence was the visible 2D input plane, discretized at steps of 0.05. Since there was no mini-batch training, the KL term was not scaled according to Section 2. This toy experiment is the only exception in this regard. For the constant uniform prior, $\boldsymbol{\beta}$ was set to $(1, 1)$. The GP prior and the random forest prior were implemented using their respective `scikit-learn` implementations by taking their categorical predictions as a Dirichlet mean and using a Dirichlet precision $z = K = 2$ to match the precision of the uniform prior. The GP used the RBF kernel with optimized hyperparameters and the random forest used 20 trees, the 'entropy' criterion and a maximum depth of 10. The random seeds were set to 123 for both GP and random forest.

For Rotated MNIST, we used all 60000 images of shape 28x28x1 reshaped to 784 from the train split with pixel values normalized to $[-1, 1]$ and no other pre-processing or data augmentation. All 10000 images from the test set were used during evaluation, rotated by a fixed degree, ranging from 0° to 180° in 10° steps, resulting in a total of 190000 test images. The MNIST (LeCun et al., 2010) data is available under the terms of the Creative Commons Attribution-Share Alike 3.0 license. All models were MLPs with two hidden layers consisting of 50 units each, bias terms enabled and ReLU activation. For the Dropout models, the dropout rate was set to 0.2 and for the Ensemble models, we used 10 members per ensemble. All models were trained for 30 epochs at a learning rate of 0.001 using a mini-batch size of 256. The measurement set for the KL divergence was the training data itself, except for the measurement set comparison section, where the different measurement sets are stated explicitly. Results were obtained using 10 random seeds.

For corrupted CIFAR10 and CIFAR100, we used all 50000 images of shape 32x32x3 from the regular train splits. Following (He et al., 2016), we normalized pixel values using the empirical mean and standard deviation, and employed data augmentation during training by first selecting random crops of size 32x32x3 after adding 4 pixels of zero padding to each side and then randomly flipping 50% of the images horizontally. All 10000 images from the regular test set were used during evaluation plus their corrupted versions (Hendrycks and Dietterich, 2019) with 19 different corruptions and 5 levels of severity, resulting in a total of 960000 test images. The CIFAR10 and CIFAR100 (Krizhevsky, 2009) data is available under the terms of the MIT License and the corrupted CIFAR10 and corrupted CIFAR100 (Hendrycks and Dietterich, 2019) data is available under the terms of the Apache License 2.0. All models were CNNs following the ResNet-18 architecture (He et al., 2016), designed for CIFAR images, rather than ImageNet.

Adopting (He et al., 2016), we trained with a batch size of 128 and used the SGD optimizer with momentum (0.9) for 200 epochs and scaled the learning rate by 0.1 at epochs 100 and 150. For the MAP, MC Dropout and Ensemble models without fVI, we used 0.0005 weight decay. For MC Dropout models (Gal and Ghahramani, 2016), the dropout rate was set to 0.2. For Ensemble (Lakshminarayanan et al., 2017) and Ensemble fVI, we used 5 members per ensemble.

For Radial BNNs (Farquhar et al., 2020) and Radial fVI, we implemented weight priors for all convolutional weights but not for the final linear layer. The standard deviation σ was parameterized using $\sigma = \log(1 + \exp(\rho))$ and ρ was initialized to -5 while the means were initialized using the `PyTorch` default initialization scheme for CNNs. For Radial BNNs without fVI, we used a closed-form Gaussian weight KL divergence with a Gaussian prior with a mean of 0 and a standard deviation of 0.1. For Radial fVI, we used our fKL instead of the weight-space KL.

For Rank-1 BNNs (Dusenberry et al., 2020) and Rank-1 fVI, we used 4 ensemble members and 250 training epochs instead of 200 due to slow convergence and scaled the learning rate by 0.1 at epochs 150 and 200. During training, we used implicit batch ensembling (Wen et al., 2020), whereas during prediction, we created explicit ensemble predictions by replicating the input. We placed Rank-1 Gaussian distributions over all convolutional weights but not over the final linear layer. The standard deviation σ was parameterized using $\sigma = \log(1 + \exp(\rho))$ and ρ was initialized to -3 while the means were initialized to 1. For Rank-1 BNNs without fVI, following (Dusenberry et al., 2020), the Rank-1 priors were Gaussian with a mean of 1

and a standard deviation of 0.1, and weight decay of 0.0001 was used. We did not use KL annealing epochs. For Rank-1 fVI, we used our fKL instead of the weight-space KL. For all fVI models, the measurement set for the KL divergence during fVI training was always the training data itself. Results were obtained using 10 random seeds.

When scaling to higher dimensional classification tasks, specifically $K \geq 100$, we observed numerical issues with the fELBO objective when using the uniform Dirichlet predictive prior. In higher dimensions, this prior would provide greater regularization. This is because the magnitude of the categorical likelihood does not change with dimensionality, as it is the log probability of the label class. Conversely, the KL between two Dirichlet densities requires summing over the parameters, so the magnitude naturally increases with K. To alleviate this over-regularization, we adopt the strategy of (Joo et al., 2020) and apply additional scaling to the KL term in the fELBO. This scaling can be shown to be numerically equivalent to a certain prior, i.e. $\boldsymbol{\beta}$ (Section 3.4, (Joo et al., 2020)). Therefore, optimizing this scaling is a form of model selection. For our CIFAR100 experiments, we simply chose a scaling such that that the fKL magnitude was close to the CIFAR10 values. We found this to be about 0.1, which matches a 10x scaling suggested by the Dirichlet KL due to the summation terms.

To ensure numerical stability, we defined a minimum and maximum precision for the posterior Dirichlet estimation: $z_{\min} = K$ and $z_{\max} = N$, where K is the number of classes and N is the number of training examples. For the MAP models, we skipped the Dirichlet MLE and set $z = z_{\max}$ because $M = 1$. Similarly, we used $M = 1$ for the MC Dropout and Radial BNN models during training and also set $z = z_{\max}$, although we set $M = 10$ during evaluation. For the Ensemble models, M was always the number of members in the ensemble and for the Rank-1 models, we replicated the input M times during evaluation, which results in M distinct predictions. Furthermore, we applied a small amount of label smoothing ${f_{\boldsymbol{x}}}_k^{(m)} \approx (1-\gamma){f_{\boldsymbol{x}}}_k^{(m)} + \gamma\frac{1}{K}$ throughout all steps of the KL divergence estimation, where γ was set to 10^{-4} for our experiments.

Minka's quasi-Newton maximum likelihood Dirichlet precision estimator (Minka, 2000), which we used for our implementation, translated to our notation, is given by

$$\frac{1}{z} = \frac{1}{z} + \frac{1}{z^2}\frac{\Delta_1}{\Delta_2}, \qquad \bar{\alpha}_k = \frac{1}{M}\sum_{m=1}^{M} {f_{\boldsymbol{x}}}_k^{(m)}, \qquad \breve{\alpha}_k = \frac{1}{M}\sum_{m=1}^{M} \log {f_{\boldsymbol{x}}}_k^{(m)}, \tag{10}$$

$$\Delta_1 = M\left(\psi_0(z) - \sum_{k=1}^{K}\bar{\alpha}_k\psi_0(z\bar{\alpha}_k) + \sum_{k=1}^{K}\bar{\alpha}_k\breve{\alpha}_k\right), \tag{11}$$

$$\Delta_2 = M\left(\psi_1(z) - \sum_{k=1}^{K}\bar{\alpha}_k^2\psi_1(z\bar{\alpha}_k)\right), \tag{12}$$

where ψ_0 is the digamma function and ψ_1 is the trigamma function.

We initialized the algorithm with an approximate maximum likelihood solution using Stirling's approximation to the gamma function Γ (Minka, 2000),

$$z^{(0)} = \frac{K-1}{-2\sum_{k=1}^{K}\bar{\alpha}_k(\log\breve{\alpha}_k - \log\bar{\alpha}_k)}. \tag{13}$$

We stopped the algorithm once the change per step is less than 10^{-5}. Counting the number of iterations until convergence for a trained Ensemble fVI model with $M = 10$

ensemble members and 10000 MNIST test examples, the mean was 3.0796, the 95th quantile was 3, the 99th quantile was 15 and the maximum was 1172. Note that the number of iterations until convergence in vectorized mini-batch computation is equal to the maximum number of iterations until convergence of the items in the mini-batch.

Although the computational complexity of the underlying deep learning model depends on the model architecture, data input size, number of parameters, etc., for the following comparison, we assume that a single forward pass through the model takes $\mathcal{O}(1)$, i.e. a constant amount of time, because the weight-space and function-space objectives share the same model. With a mini-batch size of B, computation of the standard ML objective takes $\mathcal{O}(BMK)$ time per mini-batch iteration. Assuming a constant number of quasi-Newton steps, the Dirichlet precision estimation takes $\mathcal{O}(SK + M)$ time for an measurement set of size S. Computing the fKL for an measurement set of size S takes $\mathcal{O}(SMK)$ time. If the training data is used as measurement set the forward pass through the model can be shared between the log-likelihood and fKL calculation, resulting in an overall asymptotic time complexity of $\mathcal{O}(BMK)$ per mini-batch iteration. In case of a different measurement set, the asymptotic time complexity becomes $\mathcal{O}((B+S)MK)$.

Algorithm 1 Function-Space Regularization for Deep Bayesian Classification

Require: training data $\mathcal{D} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n)\}_{n=1}^N$, implicit stochastic process $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{w})$, variational posterior $q_{\boldsymbol{\theta}}(\boldsymbol{w})$, function-space prior $p(\boldsymbol{f}_{\boldsymbol{x}}) = \mathrm{Dir}(\beta_{\boldsymbol{x}})$

while not converged **do**
  Sample mini-batch $\{(\boldsymbol{x}_l, \boldsymbol{y}_l)\}_{l=1}^L \subset \mathcal{D}$
  Sample M predictions per mini-batch item: $\boldsymbol{w}^{(m)} \sim q_{\boldsymbol{\theta}}(\boldsymbol{w})$, $\boldsymbol{f}_l^{(m)}(\boldsymbol{x}_l^{(m)}) = g_{\boldsymbol{\theta}}(\boldsymbol{x}_l^{(m)}, \boldsymbol{w}^{(m)})$
  Compute the expected log-likelihood $\mathcal{L}_\phi(\mathcal{D}) \approx \frac{1}{LM} \sum_{l,m=1}^{L,M} \mathrm{Cat}(\boldsymbol{y}_l, \boldsymbol{f}_l^{(m)})$
  Sample measurement set $\{\boldsymbol{x}_s\}_{s=1}^S \subset \mathcal{X}$
  Sample M predictions per measurement item: $\boldsymbol{w}^{(m)} \sim q_{\boldsymbol{\theta}}(\boldsymbol{w})$, $f_s^{(m)}(\boldsymbol{x}_s^{(m)}) = g_{\boldsymbol{\theta}}(\boldsymbol{x}_s^{(m)}, \boldsymbol{w}^{(m)})$
  Estimate $\alpha_{\boldsymbol{x}_s}$ using Newton method from samples $\{f_{\boldsymbol{x}_s}^{(s)}\}_{s=1}^S$
  Estimate factorized fKL: $D_{\mathrm{KL}}[q_{\boldsymbol{\theta}} \,||\, p] = \frac{1}{SM} \sum_{s,m=1}^{S,M} (\log q_{\boldsymbol{\theta}}(\boldsymbol{f}_s^{(m)}) - \log p(\boldsymbol{f}_s^{(m)}))$
  Gradient decent of fELBO approximation $\mathcal{L}(\boldsymbol{\theta}) \approx \mathcal{L}_{\boldsymbol{\theta}}(\mathcal{D}) + \frac{1}{L} D_{\mathrm{KL}}[q_{\boldsymbol{\theta}} \,||\, p]$, using reparameterization gradients or otherwise, depending on $q_{\boldsymbol{\theta}}(\boldsymbol{w})$ (Table 1).
end while

Appendix F. Ablation Studies

Samples During Training In Section 2, a Dirichlet estimation procedure was proposed using M samples. In the single sample case $M = 1$, motivated by MAP models, a crude approximation was proposed to approximate the precision with the number of training data samples. During training, the $M = 1$ approximation was also used for MC Dropout, Radial and Rank-1 BNNs, akin to their respective weight-space variational inference procedures. To assess the consequence of this approximation, we repeated the CIFAR10 corruption experiment for MC dropout with $M = 5$, matching the Ensemble models. Figure 7 shows

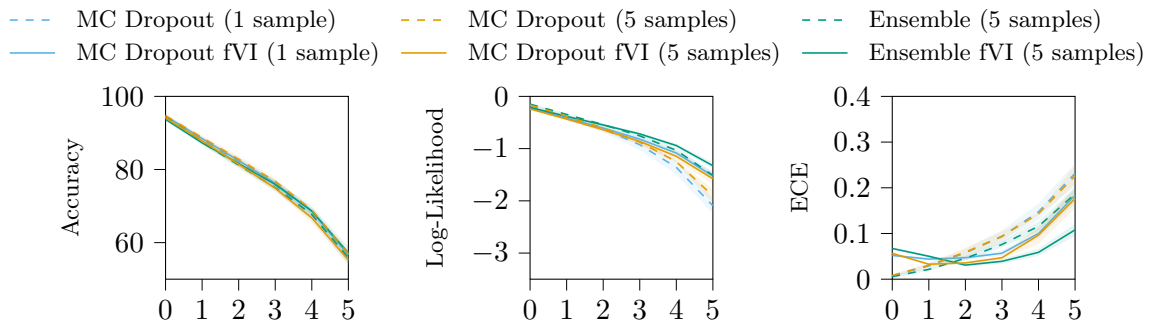

Figure 7: Reproduction of CIFAR10 corruption results in Figure 4, including MC Dropout results with 5 predictive samples during training. For 5 sample MC Dropout, 3 random seeds were used rather than 10 due to the additional training time.

that the 5 sample MC Dropout performance is closer to the 1 sample MC Dropout performance than the Ensemble. This result indicates that the model, rather than M during training, has greater impact. The similarity in performance between 1 and 5 sample MC Dropout suggests that the 1 sample approximation is reasonable.

Scaling Issues with High Label Dimensionality The CIFAR100 experiments revealed an issue with the fELBO objective that caused underfitting for the larger label dimension. To examine why, recall that the categorical likelihood is $\log f_{\boldsymbol{x}k}$ when $y_k = 1$. Therefore, the dimensionality of $\boldsymbol{y}$ does not directly influence the value. Conversely, the KL divergence between two Dirichlet densities $D_{\mathrm{KL}}(p_1||p_2)$ does incorporate the label dimensionality K (Rauber et al., 2008),

$$D_{\mathrm{KL}}(p_1||p_2) = \log\Gamma(z^{(1)}) - \sum_{k=1}^{K}\log\Gamma(\alpha_k^{(1)}) - \log\Gamma(z^{(2)}) + \sum_{k=1}^{K}\log\Gamma(\alpha_k^{(2)}) + \sum_{k=1}^{K}(\alpha_k^{(1)} - \alpha_k^{(2)})(\Psi(\alpha_k^{(1)}) - \Psi(\alpha_k^{(2)})).$$

To counteract this linear increase due to the summation terms, we can assess a heuristic annealing scale factor on the fKL during training of $1/K = \lambda$,

$$\mathcal{L}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{f}\sim q(\cdot|\boldsymbol{\theta})}\left[\log p(\mathcal{D}|\boldsymbol{f})\right] - \lambda D_{\mathrm{KL}}[q(\boldsymbol{f}|\boldsymbol{\theta}) \,||\, p(\boldsymbol{f})].$$

To investigate this relationship between the Dirichlet KL divergence and the number of classes of the classification, we conducted a toy experiment in a hypercube $[-1, 1]^D$ with fixed number of input dimensions D and increasing number of classes K. The classes were created by using each dimension as the decision boundary, i.e. $\boldsymbol{x}_d = 0$, and leveraging all permutations to create up to $K = 2^D$ classes, where D was set to 8. The training data, measurement set for fKL, and test data, consisting of 1000 data points each, were all sampled uniformly at random from the hypercube. We used a MAP model with 2-layer MLP architecture with 25 hidden units each, bias terms enabled, and ReLU activation functions. We trained for 3000 epochs using Adam optimizer with a learning rate of 0.005 and default parameters otherwise.

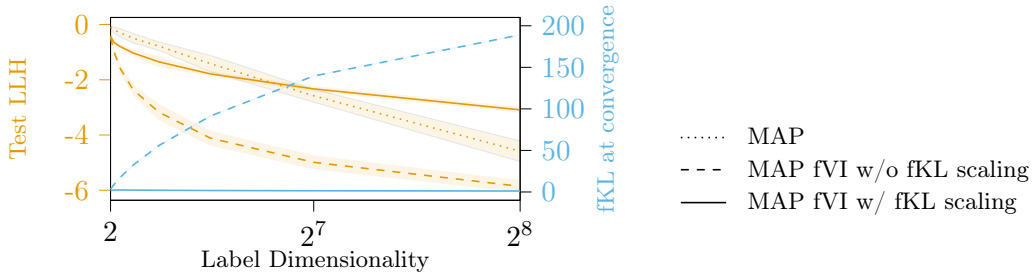

Figure 8: A toy classification example on a hypercube to illustrate the scaling issues associated with the Dirichlet density. The vanilla MAP performance acts as a baseline, and demonstrates the typical range of log-likelihood values for this task across increasing label dimensionality. For fVI, the Dirichlet fKL reports a significantly larger range that is x100 the log-likelihood range. This value imbalance affects the fELBO objective, resulting in significant underfitting. Applying a heuristic to scale the fKL term, keeping the fKL invariant across label dimensionality avoids the underfitting phemonema. Plot reports mean and 2 standard deviations over 10 seeds.

Figure 8 illustrates the model's test log-likelihood and fKL after training. The regular MAP model represents a decent baseline with linear decrease in performance as the label dimensionality increases. In contrast, the test log-likelihood of the MAP fVI model without KL scaling decreases exponentially while the fKL increases approximately linearly. Applying the above proposed scaling to the fKL during training aids the optimization, keeps the final fKL at convergence consistently low and significantly improves the model's test log-likelihood.

Changing Prior Dirichlet Parameters The uniform Dirichlet distribution with concentration parameters $\beta_k = 1$ is a natural choice for an uninformed prior over the simplex. However, potentially interesting cases to consider are priors where all β_k are set to another value which is greater or smaller than 1. While the Dirichlet mean remains the same, $\beta_k > 1$ corresponds to greater confidence that the class probabilities are uniformly distributed and $\beta_k < 1$ prefers dominance of any particular class. It was also hypothesized that scaling β_k could yield results comparable to scaling the fKL as discussed in the previous subsection. To test this hypothesis, we repeated the hypercube experiment from the previous subsection with the MAP fVI model without fKL scaling while using different β_k as prior parameters.

Figure 9 show the test log-likelihood of the MAP fVI model without fKL scaling after training with Dirichlet priors using varying prior concentration parameters β_k. However, there are no significant differences when using different β_k and no particular β_k achieves test log-likelihoods which would be comparable to the improvements due to fKL scaling discussed in the previous subsection.

To further investigate the Dirichlet prior with different concentration parameters, we repeated the visualizable Two Moons toy problem using the MC Dropout fVI model with different β_k. Figure 10 depicts the predicted class probabilities of the toy problem with $K = 2$ classes. For $\beta_k < 1$, the areas of confident prediction enlarge but quickly fall back

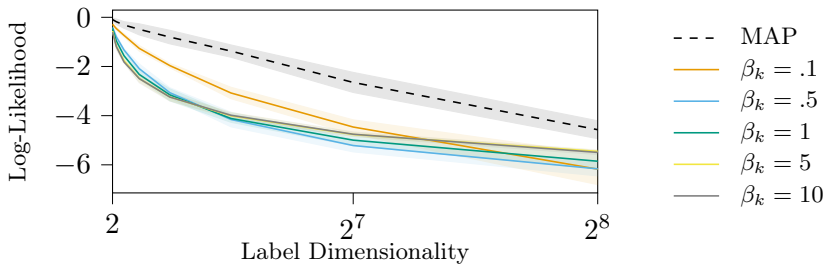

Figure 9: Test log-likelihood values for the hypercube toy problem with prior parameters β_k larger or smaller than 1.

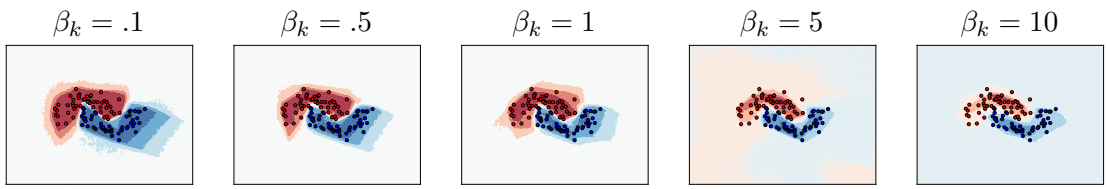

Figure 10: Reproduction of the Two Moons toy problem (Figure 1) with varying uniform prior precision.

to uniformity, whereas for $\beta_k > 1$, the confident predictions or more locally concentrated, slowly tapering towards uniformity.

Appendix G. Approximate Gradient Computation

As stated in the main paper, we do not compute the total derivative of the approximate fKL divergence

$$D_{\mathrm{KL}}[q||p] \approx \frac{1}{M} \sum_{l,m=1}^{L,M} \left(\log q \left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)} \middle| \boldsymbol{\theta} \right) - \log p \left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)} \right) \right), \tag{14}$$

$$= \frac{1}{M} \sum_{l,m=1}^{L,M} \left(\log q \left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}(\boldsymbol{\theta}) \middle| \boldsymbol{\alpha} \left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(1:M)}(\boldsymbol{\theta}) \right) \right) - \log p \left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}(\boldsymbol{\theta}) \right) \right), \tag{15}$$

but only a partial one. Note that we have introduced the symbols $\boldsymbol{\alpha} \left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(1:M)}(\boldsymbol{\theta}) \right)$ and $\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}(\boldsymbol{\theta})$ to highlight the dependence of the Dirichlet posterior estimates on the M implicit functions $\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}(\boldsymbol{\theta}),\ m \in [1, M]$ and the dependence of the individual implicit functions on the network

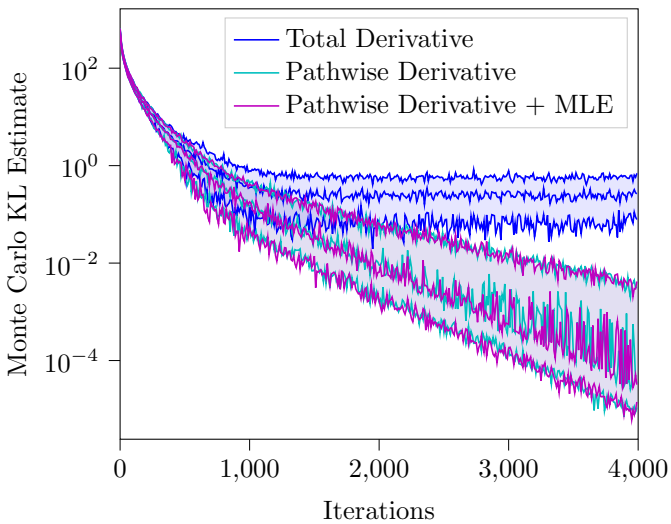

Figure 11: Comparing total and pathwise gradients for fitting a 100-dimensional Dirichlet using a Monte Carlo KL estimate, inspired by Figure 1 of (Roeder et al., 2017). The total gradient has visibly higher variance when viewed in log-space, leading the premature convergence. Moreover, using an MLE fit from 5 samples for the pathwise estimator has no visible difference to the quality of the gradient estimate. Results are over 50 seeds, showing the 10th, 50th and 90th percentiles.

parameters $\boldsymbol{\theta}$. The total derivative of above approximate KL divergence corresponds to

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} D_{\mathrm{KL}}[q||p] &\approx \frac{1}{M}\sum_{l,m=1}^{L,M}\left(\nabla_{\boldsymbol{\theta}}\log q\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}(\boldsymbol{\theta})\middle|\boldsymbol{\alpha}\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(1:M)}(\boldsymbol{\theta})\right)\right) - \nabla_{\boldsymbol{\theta}}\log p\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}(\boldsymbol{\theta})\right)\right)\\ &= \frac{1}{M}\sum_{l,m=1}^{L,M}\partial_{\boldsymbol{f}}\left(\log q\left(\boldsymbol{f}\middle|\boldsymbol{\alpha}\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(1:M)}(\boldsymbol{\theta})\right)\right) - \log p\left(\boldsymbol{f}\right)\right)\Big|_{\boldsymbol{f}=\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}(\boldsymbol{\theta})}\nabla_{\boldsymbol{\theta}}\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}(\boldsymbol{\theta})\\ &\quad + \frac{1}{M}\sum_{l,m=1}^{L,M}\partial_{\boldsymbol{\alpha}}\log q\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}(\boldsymbol{\theta})\middle|\boldsymbol{\alpha}\right)\Big|_{\boldsymbol{\alpha}=\boldsymbol{\alpha}\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(1:M)}(\boldsymbol{\theta})\right)}\nabla_{\boldsymbol{\theta}}\boldsymbol{\alpha}\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(1:M)}(\boldsymbol{\theta})\right). \qquad (16)\end{aligned}$$

The partial derivative with which we optimize the fKL divergence in our algorithm omits the blue term in (16). This simplifies the computation graph, as $\boldsymbol{\alpha}\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(1:M)}(\boldsymbol{\theta})\right)$ is a maximum-likelihood estimate computed from the implicit function samples $\boldsymbol{f}_{\boldsymbol{s}_l}^{(m)}(\boldsymbol{\theta})$. Computing $\nabla_{\boldsymbol{\theta}}\boldsymbol{\alpha}\left(\boldsymbol{f}_{\boldsymbol{s}_l}^{(1:M)}(\boldsymbol{\theta})\right)$ requires to compute the gradient of a maximum-likelihood estimate (MLE) $\boldsymbol{\alpha}$ w.r.t. the implicit functions $\boldsymbol{f}$. Given that we use an iterative scheme to compute an approximate MLE, this would require us to differentiate through each iteration of the MLE computation.

We now want to provide evidence that using this partial derivative of the approximate fKL divergence is still a reasonable choice. As argued in (Roeder et al., 2017), terms of the form $\mathbb{E}_{p(x|\boldsymbol{\theta})}\left[\partial_{\boldsymbol{\theta}}\log p(x|\boldsymbol{\theta})\right] = 0$ tend to introduce high variance into the gradient of variational inference objectives due to the Monte Carlo expectation approximation. Therefore, omitting

that term from the gradient estimate can actually benefit the convergence of gradient-based variational inference methods in certain cases. We visualize this phenomena in Figure 11 for fitting a high-dimensional Dirichlet, in which the gradient-based optimization of a KL divergence objective using the total derivative prematurely converges due to high variance, while an optimization that ignores the variance-inducing terms does not face this problem. Moreover, the MLE fit of the variational density using 5 samples has no visible effect of the gradient estimation quality, despite fitting a 100-dimensional distribution. This result indicates that as long as the predictive distribution is approximately Dirichlet, which is a central assumption of this approach, the gradient assumption is reasonable.

However, our method does not exactly match the pathwise gradient of Roeder et al. (2017). The ignored term of the total approximate fKL divergence derivative (16) resembles the variance-inducing term, as the M implicit functions $\boldsymbol{f}^{(m)}(\boldsymbol{\theta})$ evaluated at the different elements $\boldsymbol{s}$ of the measurement set $\mathcal{S}$ approximate the expectation over $q\left(\boldsymbol{f}|\boldsymbol{\alpha}\right)$

$$\frac{1}{M}\sum_{m=1}^{M}\partial_{\boldsymbol{\alpha}}\log q\left(\boldsymbol{f}_{\boldsymbol{s}}^{(m)}(\boldsymbol{\theta})\middle|\boldsymbol{\alpha}\right)\approx\mathbb{E}_{q(\boldsymbol{f}|\boldsymbol{\alpha})}\left[\partial_{\boldsymbol{\alpha}}\log q\left(\boldsymbol{f}|\boldsymbol{\alpha}\right)\right]=0. \tag{17}$$

However, the implicit function samples $\boldsymbol{f}_{\boldsymbol{s}}^{(m)}(\boldsymbol{\theta})$ are clearly not i.i.d. samples as there is a tight correlation between the parameter $\boldsymbol{\alpha}$ of the Dirichlet distribution q and the implicit function samples. Moreover, our ensemble model does not use reparameterized gradients, optimizing a set of network weight 'particles' instead. Therefore, we also compared the optimization of the approximate fKL divergence objective using the partial- and total derivative on a particle-based variational representation of the Dirichlet. To compute the gradient of the MLE w.r.t. the implicit function samples required by the total derivative, we first compute the MLE $\boldsymbol{\alpha}$ by solving the underlying convex optimization problem (Minka, 2000) using the `CVXOpt` library (Andersen et al.). We then leverage the implicit function theorem (Dontchev and Rockafellar, 2009) to compute the gradient of $\boldsymbol{\alpha}$ w.r.t. the samples, leveraging that the gradient of the likelihood function vanishes for the MLE

$$\mathbf{0}=\nabla_{\boldsymbol{f}^{(1:M)}}\log\left(p(\boldsymbol{f}^{(1:M)}|\boldsymbol{\alpha})\right).$$

The results in Figure 12 indeed highlight that in the setting of this paper, the total derivative leads to a faster descent along the fKL divergence objective per gradient step when optimizing particles. However, we see that the partial derivatives also minimize the fKL divergence. Furthermore, due to the lower computational overhead of the partial derivatives, this optimization is carried out in significantly less time.

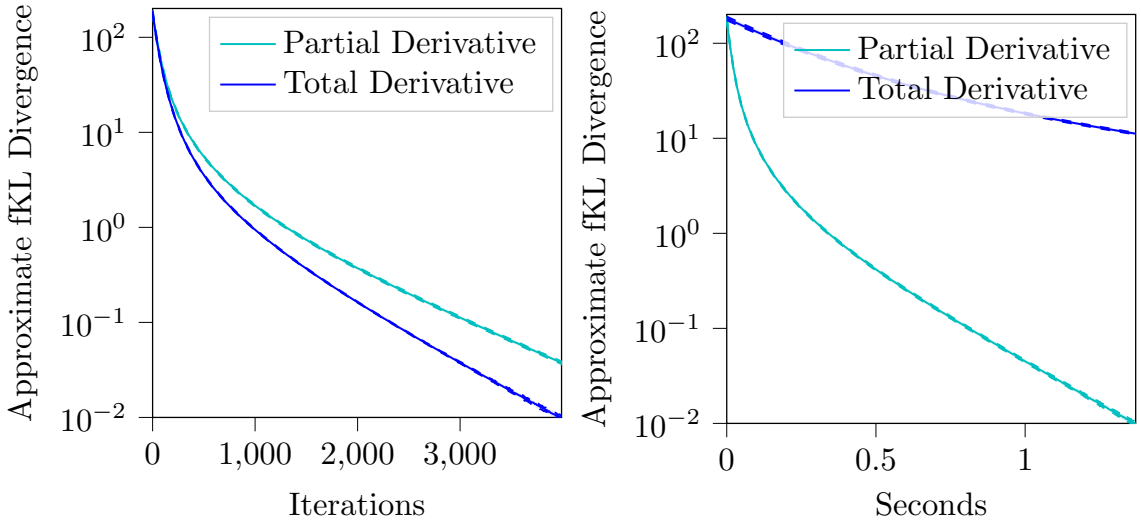

Figure 12: A comparison of the total- and partial derivative for minimizing approximate fKL divergence (15) of an empirical distribution $q(\boldsymbol{f}) \approx \frac{1}{M}\sum_{m=1}^{M} \delta_{\boldsymbol{f}^{(m)}}(\boldsymbol{f}),\ \boldsymbol{f}^{(m)} \sim q(\boldsymbol{f})$ represented by $M = 5$ samples to a 100-dimensional Dirichlet prior p w.r.t. the samples $\boldsymbol{f}^{(m)}$. The curves show the fKL divergence over gradient steps (left) and time (right) using the total and partial derivative. The partial derivative (that ignores the blue term in (16)) leads to a slower decline in fKL divergence per gradient step. However, given the much cheaper computational cost of the partial derivative, it leads to a faster optimization of the fKL divergence w.r.t. time. Results are computed over 50 seeds, showing the 10th, 50th and 90th percentiles. For each seed, the samples $\boldsymbol{f}^{(m)}$ were intialized differently.

Appendix H. Experimental Results

Numerical values of all our experimental results are reported here.

Table 2: Accuracies for the rotated MNIST experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

MNIST	Angle °						
Accuracy ↑	0	10	20	30	40	50	60
MAP	96.82 ± 0.06	94.19 ± 0.15	87.17 ± 0.43	71.60 ± 0.76	53.11 ± 0.76	$\mathbf{36.03 \pm 0.52}$	$\mathbf{25.18 \pm 0.46}$
MAP fVI	$\mathbf{97.09 \pm 0.05}$	$\mathbf{95.01 \pm 0.13}$	$\mathbf{88.51 \pm 0.32}$	$\mathbf{73.53 \pm 0.51}$	$\mathbf{54.32 \pm 0.57}$	36.03 ± 0.40	24.56 ± 0.35
MC Dropout	$\mathbf{96.23 \pm 0.05}$	93.63 ± 0.16	$\mathbf{86.77 \pm 0.16}$	$\mathbf{71.06 \pm 0.28}$	$\mathbf{52.46 \pm 0.48}$	$\mathbf{35.57 \pm 0.56}$	25.28 ± 0.54
MC Dropout fVI	96.10 ± 0.08	$\mathbf{93.70 \pm 0.17}$	86.61 ± 0.42	71.04 ± 0.74	52.20 ± 0.71	34.94 ± 0.55	$\mathbf{25.38 \pm 0.33}$
Ensemble	97.97 ± 0.02	96.36 ± 0.04	91.02 ± 0.18	76.98 ± 0.25	57.96 ± 0.29	39.52 ± 0.27	27.94 ± 0.32
Ensemble fVI	$\mathbf{98.09 \pm 0.02}$	$\mathbf{96.71 \pm 0.02}$	$\mathbf{91.95 \pm 0.13}$	$\mathbf{78.87 \pm 0.23}$	$\mathbf{60.08 \pm 0.31}$	$\mathbf{40.81 \pm 0.22}$	$\mathbf{28.20 \pm 0.29}$

Table 3: Accuracies for the rotated MNIST experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

MNIST Accuracy ↑	Angle ° 70	80	90	100	110	120
MAP	**18.60 ± 0.39**	**15.01 ± 0.34**	12.81 ± 0.46	11.77 ± 0.52	11.66 ± 0.49	14.09 ± 0.33
MAP fVI	17.91 ± 0.42	14.72 ± 0.47	**13.33 ± 0.48**	**13.02 ± 0.40**	**13.33 ± 0.28**	**15.38 ± 0.21**
MC Dropout	20.26 ± 0.48	17.22 ± 0.47	15.11 ± 0.43	13.41 ± 0.38	12.67 ± 0.35	14.32 ± 0.23
MC Dropout fVI	**20.99 ± 0.39**	**18.22 ± 0.34**	**16.26 ± 0.39**	**14.82 ± 0.46**	**13.69 ± 0.38**	**14.44 ± 0.29**
Ensemble	19.90 ± 0.31	16.11 ± 0.32	13.62 ± 0.31	12.09 ± 0.26	12.31 ± 0.22	14.92 ± 0.21
Ensemble fVI	**20.08 ± 0.38**	**16.43 ± 0.37**	**14.28 ± 0.30**	**13.07 ± 0.21**	**13.19 ± 0.16**	**15.43 ± 0.12**

Table 4: Accuracies for the rotated MNIST experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

MNIST Accuracy ↑	Angle ° 130	140	150	160	170	180
MAP	16.92 ± 0.43	19.05 ± 0.56	22.36 ± 0.55	25.06 ± 0.64	26.95 ± 0.58	28.58 ± 0.57
MAP fVI	**17.94 ± 0.29**	**19.82 ± 0.43**	**22.45 ± 0.54**	**25.26 ± 0.46**	**27.42 ± 0.43**	**28.93 ± 0.42**
MC Dropout	**17.35 ± 0.34**	20.11 ± 0.49	22.82 ± 0.43	24.37 ± 0.39	26.03 ± 0.43	28.14 ± 0.43
MC Dropout fVI	17.12 ± 0.35	**20.25 ± 0.40**	**23.25 ± 0.34**	**25.24 ± 0.27**	**26.86 ± 0.18**	**28.95 ± 0.29**
Ensemble	18.02 ± 0.32	20.83 ± 0.29	24.49 ± 0.26	27.61 ± 0.20	30.03 ± 0.24	31.27 ± 0.32
Ensemble fVI	**18.47 ± 0.19**	**21.49 ± 0.20**	**25.21 ± 0.21**	**28.54 ± 0.24**	**31.31 ± 0.29**	**32.51 ± 0.31**

Table 5: Log-likelihoods for the rotated MNIST experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

MNIST Log-Likelihood ↑	Angle ° 0	10	20	30	40	50	60
MAP	**−0.11 ± 0.00**	**−0.20 ± 0.00**	−0.48 ± 0.02	−1.20 ± 0.03	−2.36 ± 0.04	−3.80 ± 0.04	−5.05 ± 0.05
MAP fVI	−0.13 ± 0.00	−0.20 ± 0.00	**−0.39 ± 0.01**	**−0.84 ± 0.02**	**−1.53 ± 0.02**	**−2.34 ± 0.03**	**−3.02 ± 0.03**
MC Dropout	**−0.15 ± 0.00**	**−0.23 ± 0.00**	**−0.43 ± 0.00**	**−0.88 ± 0.01**	−1.50 ± 0.02	−2.28 ± 0.03	−2.97 ± 0.04
MC Dropout fVI	−0.20 ± 0.00	−0.28 ± 0.01	−0.48 ± 0.01	−0.91 ± 0.02	**−1.46 ± 0.02**	**−2.06 ± 0.02**	**−2.50 ± 0.02**
Ensemble	**−0.07 ± 0.00**	**−0.12 ± 0.00**	**−0.29 ± 0.00**	−0.73 ± 0.01	−1.50 ± 0.02	−2.56 ± 0.03	−3.60 ± 0.04
Ensemble fVI	−0.11 ± 0.00	−0.17 ± 0.00	−0.32 ± 0.00	**−0.67 ± 0.01**	**−1.20 ± 0.01**	**−1.85 ± 0.01**	**−2.43 ± 0.01**

Table 6: Log-likelihoods for the rotated MNIST experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

MNIST Log-Likelihood ↑	Angle ° 70	80	90	100	110	120
MAP	−6.05 ± 0.06	−6.79 ± 0.07	−7.22 ± 0.09	−7.34 ± 0.09	−7.27 ± 0.10	−6.95 ± 0.09
MAP fVI	**−3.54 ± 0.06**	**−3.94 ± 0.07**	**−4.14 ± 0.08**	**−4.18 ± 0.08**	**−4.16 ± 0.07**	**−4.05 ± 0.06**
MC Dropout	−3.47 ± 0.04	−3.88 ± 0.04	−4.17 ± 0.05	−4.28 ± 0.05	−4.32 ± 0.05	−4.15 ± 0.05
MC Dropout fVI	**−2.81 ± 0.02**	**−3.04 ± 0.03**	**−3.20 ± 0.04**	**−3.25 ± 0.04**	**−3.28 ± 0.04**	**−3.17 ± 0.03**
Ensemble	−4.52 ± 0.04	−5.31 ± 0.04	−5.77 ± 0.05	−5.90 ± 0.06	−5.86 ± 0.06	−5.68 ± 0.06
Ensemble fVI	**−2.89 ± 0.02**	**−3.27 ± 0.02**	**−3.49 ± 0.02**	**−3.57 ± 0.02**	**−3.57 ± 0.02**	**−3.47 ± 0.02**

Table 7: Log-likelihoods for the rotated MNIST experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

MNIST	Angle °					
Log-Likelihood ↑	130	140	150	160	170	180
MAP	-6.57 ± 0.09	-6.36 ± 0.08	-6.20 ± 0.07	-6.16 ± 0.06	-6.13 ± 0.05	-6.00 ± 0.07
MAP fVI	$\mathbf{-3.87 \pm 0.05}$	$\mathbf{-3.78 \pm 0.04}$	$\mathbf{-3.68 \pm 0.04}$	$\mathbf{-3.63 \pm 0.03}$	$\mathbf{-3.61 \pm 0.04}$	$\mathbf{-3.50 \pm 0.04}$
MC Dropout	-3.93 ± 0.05	-3.85 ± 0.06	-3.87 ± 0.05	-4.04 ± 0.04	-4.15 ± 0.04	-4.19 ± 0.04
MC Dropout fVI	$\mathbf{-3.03 \pm 0.02}$	$\mathbf{-2.94 \pm 0.02}$	$\mathbf{-2.91 \pm 0.02}$	$\mathbf{-2.97 \pm 0.02}$	$\mathbf{-3.00 \pm 0.01}$	$\mathbf{-2.98 \pm 0.01}$
Ensemble	-5.37 ± 0.05	-5.24 ± 0.04	-5.18 ± 0.02	-5.31 ± 0.01	-5.37 ± 0.02	-5.25 ± 0.02
Ensemble fVI	$\mathbf{-3.29 \pm 0.02}$	$\mathbf{-3.22 \pm 0.01}$	$\mathbf{-3.17 \pm 0.01}$	$\mathbf{-3.21 \pm 0.02}$	$\mathbf{-3.23 \pm 0.02}$	$\mathbf{-3.16 \pm 0.02}$

Table 8: Expected calibration errors for the rotated MNIST experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

MNIST	Angle °						
ECE ↓	0	10	20	30	40	50	60
MAP	$\mathbf{0.01 \pm 0.00}$	$\mathbf{0.02 \pm 0.00}$	0.06 ± 0.00	0.17 ± 0.01	0.32 ± 0.01	0.46 ± 0.00	0.56 ± 0.00
MAP fVI	0.05 ± 0.00	0.05 ± 0.00	$\mathbf{0.04 \pm 0.00}$	$\mathbf{0.03 \pm 0.00}$	$\mathbf{0.15 \pm 0.01}$	$\mathbf{0.28 \pm 0.01}$	$\mathbf{0.38 \pm 0.01}$
MC Dropout	$\mathbf{0.04 \pm 0.00}$	$\mathbf{0.05 \pm 0.00}$	$\mathbf{0.05 \pm 0.00}$	$\mathbf{0.02 \pm 0.00}$	0.13 ± 0.01	0.25 ± 0.01	0.33 ± 0.01
MC Dropout fVI	0.09 ± 0.00	0.10 ± 0.00	0.10 ± 0.00	0.04 ± 0.01	$\mathbf{0.07 \pm 0.01}$	$\mathbf{0.19 \pm 0.01}$	$\mathbf{0.27 \pm 0.01}$
Ensemble	$\mathbf{0.01 \pm 0.00}$	$\mathbf{0.02 \pm 0.00}$	$\mathbf{0.01 \pm 0.00}$	$\mathbf{0.04 \pm 0.00}$	0.15 ± 0.00	0.27 ± 0.00	0.36 ± 0.00
Ensemble fVI	0.06 ± 0.00	0.08 ± 0.00	0.10 ± 0.00	0.08 ± 0.00	$\mathbf{0.02 \pm 0.00}$	$\mathbf{0.14 \pm 0.00}$	$\mathbf{0.23 \pm 0.00}$

Table 9: Expected calibration errors for the rotated MNIST experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

MNIST	Angle °					
ECE ↓	70	80	90	100	110	120
MAP	0.62 ± 0.00	0.65 ± 0.01	0.68 ± 0.01	0.69 ± 0.01	0.69 ± 0.01	0.67 ± 0.01
MAP fVI	$\mathbf{0.44 \pm 0.01}$	$\mathbf{0.48 \pm 0.01}$	$\mathbf{0.50 \pm 0.01}$	$\mathbf{0.50 \pm 0.01}$	$\mathbf{0.51 \pm 0.01}$	$\mathbf{0.50 \pm 0.00}$
MC Dropout	0.39 ± 0.01	0.43 ± 0.01	0.46 ± 0.01	0.48 ± 0.01	0.50 ± 0.01	0.48 ± 0.01
MC Dropout fVI	$\mathbf{0.32 \pm 0.01}$	$\mathbf{0.36 \pm 0.01}$	$\mathbf{0.38 \pm 0.01}$	$\mathbf{0.40 \pm 0.01}$	$\mathbf{0.42 \pm 0.01}$	$\mathbf{0.40 \pm 0.01}$
Ensemble	0.43 ± 0.00	0.46 ± 0.00	0.49 ± 0.00	0.51 ± 0.01	0.52 ± 0.00	0.50 ± 0.00
Ensemble fVI	$\mathbf{0.31 \pm 0.00}$	$\mathbf{0.35 \pm 0.00}$	$\mathbf{0.38 \pm 0.00}$	$\mathbf{0.40 \pm 0.00}$	$\mathbf{0.41 \pm 0.00}$	$\mathbf{0.40 \pm 0.00}$

Table 10: Expected calibration errors for the rotated MNIST experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

MNIST	Angle °					
ECE ↓	130	140	150	160	170	180
MAP	0.65 ± 0.01	0.63 ± 0.01	0.60 ± 0.01	0.58 ± 0.01	0.57 ± 0.01	0.56 ± 0.01
MAP fVI	$\mathbf{0.48 \pm 0.00}$	$\mathbf{0.47 \pm 0.01}$	$\mathbf{0.45 \pm 0.01}$	$\mathbf{0.43 \pm 0.01}$	$\mathbf{0.41 \pm 0.01}$	$\mathbf{0.40 \pm 0.01}$
MC Dropout	0.45 ± 0.01	0.42 ± 0.01	0.41 ± 0.01	0.41 ± 0.01	0.41 ± 0.01	0.40 ± 0.00
MC Dropout fVI	$\mathbf{0.37 \pm 0.00}$	$\mathbf{0.35 \pm 0.00}$	$\mathbf{0.33 \pm 0.00}$	$\mathbf{0.34 \pm 0.00}$	$\mathbf{0.33 \pm 0.00}$	$\mathbf{0.32 \pm 0.00}$
Ensemble	0.48 ± 0.00	0.46 ± 0.00	0.44 ± 0.00	0.43 ± 0.00	0.42 ± 0.00	0.42 ± 0.00
Ensemble fVI	$\mathbf{0.38 \pm 0.00}$	$\mathbf{0.35 \pm 0.00}$	$\mad{0.33 \pm 0.00}$	$\mad{0.32 \pm 0.00}$	$\mad{0.31 \pm 0.00}$	$\mad{0.30 \pm 0.00}$

Table 11: Accuracies for the corrupted CIFAR10 experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR10 Accuracy ↑	Corruption Severity 0	1	2	3	4	5
MAP	94.32 ± 0.05	87.65 ± 0.08	$\mathbf{81.76 \pm 0.09}$	$\mathbf{75.97 \pm 0.13}$	$\mathbf{68.86 \pm 0.20}$	$\mathbf{57.28 \pm 0.21}$
MAP fVI	$\mathbf{94.40 \pm 0.08}$	$\mathbf{87.66 \pm 0.06}$	81.66 ± 0.10	75.81 ± 0.16	68.77 ± 0.17	57.06 ± 0.18
MC Dropout	$\mathbf{94.32 \pm 0.04}$	$\mathbf{88.21 \pm 0.07}$	$\mathbf{82.13 \pm 0.13}$	$\mathbf{75.81 \pm 0.19}$	$\mathbf{67.59 \pm 0.21}$	$\mathbf{55.72 \pm 0.26}$
MC Dropout fVI	93.38 ± 0.03	87.01 ± 0.09	80.64 ± 0.14	74.36 ± 0.18	66.33 ± 0.18	54.87 ± 0.19
Ensemble	$\mathbf{95.30 \pm 0.04}$	89.37 ± 0.03	83.77 ± 0.06	78.21 ± 0.07	71.05 ± 0.11	59.33 ± 0.16
Ensemble fVI	95.26 ± 0.03	$\mathbf{89.44 \pm 0.03}$	$\mathbf{83.94 \pm 0.07}$	$\mathbf{78.49 \pm 0.08}$	$\mathbf{71.42 \pm 0.12}$	$\mathbf{59.73 \pm 0.16}$
Radial	$\mathbf{95.05 \pm 0.04}$	$\mathbf{87.98 \pm 0.07}$	$\mathbf{81.82 \pm 0.10}$	75.93 ± 0.12	$\mathbf{68.93 \pm 0.16}$	57.24 ± 0.21
Radial fVI	93.73 ± 0.03	87.43 ± 0.07	81.75 ± 0.14	$\mathbf{76.09 \pm 0.21}$	68.84 ± 0.29	$\mathbf{57.42 \pm 0.35}$
Rank1	93.68 ± 0.05	87.60 ± 0.05	82.32 ± 0.08	$\mathbf{76.90 \pm 0.08}$	$\mathbf{69.92 \pm 0.13}$	$\mathbf{58.53 \pm 0.17}$
Rank1 fVI	$\mathbf{93.91 \pm 0.04}$	$\mathbf{87.75 \pm 0.05}$	$\mathbf{82.38 \pm 0.09}$	76.77 ± 0.14	69.69 ± 0.17	58.23 ± 0.19
Subnetwork	91.00 ± 0.00	83.00 ± 1.00	77.00 ± 0.00	68.00 ± 1.00	64.00 ± 1.00	59.00 ± 0.00
Belief Matching	94.52 ± 0.03	86.98 ± 0.12	80.39 ± 0.21	73.62 ± 0.29	65.74 ± 0.35	53.67 ± 0.36
Prior Networks	66.65 ± 0.61	61.28 ± 0.43	57.87 ± 0.37	54.71 ± 0.35	50.24 ± 0.33	42.29 ± 0.31

Table 12: Log-likelihoods for the corrupted CIFAR10 experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR10 Log-Likelihood ↑	Corruption Severity 0	1	2	3	4	5
MAP	$\mathbf{-0.22 \pm 0.00}$	-0.52 ± 0.00	-0.80 ± 0.01	-1.12 ± 0.01	-1.52 ± 0.01	-2.20 ± 0.02
MAP fVI	-0.25 ± 0.00	$\mathbf{-0.48 \pm 0.00}$	$\mathbf{-0.69 \pm 0.00}$	$\mathbf{-0.90 \pm 0.01}$	$\mathbf{-1.16 \pm 0.01}$	$\mathbf{-1.60 \pm 0.01}$
MC Dropout	$\mathbf{-0.17 \pm 0.00}$	$\mathbf{-0.39 \pm 0.00}$	$\mathbf{-0.63 \pm 0.01}$	-0.93 ± 0.01	-1.36 ± 0.02	-2.09 ± 0.02
MC Dropout fVI	-0.25 ± 0.00	-0.44 ± 0.00	-0.64 ± 0.01	$\mathbf{-0.85 \pm 0.01}$	$\mathbf{-1.12 \pm 0.01}$	$\mathbf{-1.56 \pm 0.01}$
Ensemble	$\mathbf{-0.15 \pm 0.00}$	$\mathbf{-0.35 \pm 0.00}$	$\mathbf{-0.54 \pm 0.00}$	-0.76 ± 0.00	-1.03 ± 0.01	-1.51 ± 0.01
Ensemble fVI	-0.21 ± 0.00	-0.38 ± 0.00	-0.55 ± 0.00	$\mathbf{-0.72 \pm 0.00}$	$\mathbf{-0.94 \pm 0.00}$	$\mathbf{-1.33 \pm 0.01}$
Radial	$\mathbf{-0.21 \pm 0.00}$	-0.58 ± 0.00	-0.93 ± 0.01	-1.32 ± 0.01	-1.79 ± 0.01	-2.61 ± 0.02
Radial fVI	-0.28 ± 0.00	$\mathbf{-0.49 \pm 0.00}$	$\mathbf{-0.69 \pm 0.01}$	$\mathbf{-0.90 \pm 0.01}$	$\mathbf{-1.17 \pm 0.01}$	$\mathbf{-1.61 \pm 0.02}$
Rank1	-0.33 ± 0.00	-0.71 ± 0.01	-1.06 ± 0.01	-1.46 ± 0.01	-2.02 ± 0.02	-3.00 ± 0.03
Rank1 fVI	$\mathbf{-0.27 \pm 0.00}$	$\mathbf{-0.47 \pm 0.00}$	$\mathbf{-0.65 \pm 0.00}$	$\mathbf{-0.85 \pm 0.01}$	$\mathbf{-1.10 \pm 0.01}$	$\mathbf{-1.53 \pm 0.01}$
Subnetwork	-0.27 ± 0.00	-0.51 ± 0.01	-0.73 ± 0.01	-1.06 ± 0.02	-1.25 ± 0.03	-1.47 ± 0.03
Belief Matching	-0.26 ± 0.00	-0.51 ± 0.00	-0.73 ± 0.01	-0.97 ± 0.01	-1.26 ± 0.01	-1.70 ± 0.02
Prior Networks	-1.45 ± 0.02	-1.59 ± 0.01	-1.66 ± 0.01	-1.73 ± 0.01	-1.85 ± 0.01	-2.06 ± 0.01

Table 13: Expected calibration errors for the corrupted CIFAR10 experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR10	Corruption Severity					
ECE ↓	0	1	2	3	4	5
MAP	**0.03 ± 0.00**	0.08 ± 0.00	0.12 ± 0.00	0.16 ± 0.00	0.22 ± 0.00	0.30 ± 0.00
MAP fVI	0.05 ± 0.00	**0.05 ± 0.00**	**0.05 ± 0.00**	**0.09 ± 0.00**	**0.14 ± 0.00**	**0.22 ± 0.00**
MC Dropout	**0.01 ± 0.00**	**0.03 ± 0.00**	0.06 ± 0.00	0.09 ± 0.00	0.15 ± 0.00	0.23 ± 0.00
MC Dropout fVI	0.06 ± 0.00	0.03 ± 0.00	**0.03 ± 0.00**	**0.05 ± 0.00**	**0.09 ± 0.00**	**0.17 ± 0.00**
Ensemble	**0.01 ± 0.00**	**0.02 ± 0.00**	0.05 ± 0.00	0.08 ± 0.00	0.12 ± 0.00	0.19 ± 0.00
Ensemble fVI	0.07 ± 0.00	0.05 ± 0.00	**0.03 ± 0.00**	**0.04 ± 0.00**	**0.06 ± 0.00**	**0.11 ± 0.00**
Radial	**0.03 ± 0.00**	0.08 ± 0.00	0.13 ± 0.00	0.17 ± 0.00	0.23 ± 0.00	0.32 ± 0.00
Radial fVI	0.05 ± 0.00	**0.05 ± 0.00**	**0.05 ± 0.00**	**0.09 ± 0.00**	**0.14 ± 0.00**	**0.23 ± 0.00**
Rank1	**0.04 ± 0.00**	0.09 ± 0.00	0.13 ± 0.00	0.17 ± 0.00	0.23 ± 0.00	0.32 ± 0.00
Rank1 fVI	0.05 ± 0.00	**0.05 ± 0.00**	**0.04 ± 0.00**	**0.07 ± 0.00**	**0.12 ± 0.00**	**0.20 ± 0.00**
Subnetwork	0.01 ± 0.00	0.03 ± 0.00	0.06 ± 0.00	0.11 ± 0.01	0.13 ± 0.01	0.16 ± 0.01
Belief Matching	0.07 ± 0.00	0.07 ± 0.00	0.07 ± 0.00	0.09 ± 0.00	0.14 ± 0.00	0.24 ± 0.00
Prior Networks	0.20 ± 0.00	0.21 ± 0.00	0.21 ± 0.00	0.22 ± 0.00	0.23 ± 0.00	0.24 ± 0.00

Table 14: Accuracies for the CIFAR10 adversarial attack experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR10	Adversarial Attack Epsilon						
Accuracy ↑	0.00	0.05	0.10	0.15	0.20	0.25	0.30
MAP	94.32 ± 0.05	42.35 ± 0.23	35.61 ± 0.31	31.93 ± 0.38	28.04 ± 0.41	23.60 ± 0.40	19.65 ± 0.34
MAP fVI	**94.40 ± 0.08**	**70.04 ± 0.17**	**61.10 ± 0.20**	**51.32 ± 0.47**	**40.86 ± 0.68**	**31.60 ± 0.77**	**24.80 ± 0.63**
MC Dropout	**94.32 ± 0.02**	43.30 ± 0.15	32.76 ± 0.24	29.39 ± 0.25	27.08 ± 0.31	23.83 ± 0.38	19.99 ± 0.44
MC Dropout fVI	93.43 ± 0.04	**56.99 ± 0.16**	**49.91 ± 0.29**	**43.73 ± 0.39**	**36.97 ± 0.57**	**29.86 ± 0.68**	**23.63 ± 0.68**
Ensemble	**95.30 ± 0.04**	43.99 ± 0.10	29.06 ± 0.13	22.55 ± 0.13	18.58 ± 0.15	15.68 ± 0.15	13.54 ± 0.15
Ensemble fVI	95.26 ± 0.03	**60.12 ± 0.10**	**49.34 ± 0.21**	**39.38 ± 0.36**	**30.59 ± 0.42**	**23.73 ± 0.43**	**18.92 ± 0.39**
Radial	**94.84 ± 0.04**	27.92 ± 0.17	18.50 ± 0.18	14.98 ± 0.22	12.69 ± 0.25	11.28 ± 0.22	10.47 ± 0.20
Radial fVI	93.72 ± 0.03	**67.33 ± 0.13**	**59.92 ± 0.25**	**52.07 ± 0.40**	**43.46 ± 0.60**	**35.32 ± 0.64**	**28.44 ± 0.66**
Rank1	93.55 ± 0.05	19.65 ± 0.21	9.66 ± 0.18	8.34 ± 0.16	8.47 ± 0.17	8.77 ± 0.20	9.11 ± 0.19
Rank1 fVI	**93.86 ± 0.04**	**67.89 ± 0.14**	**58.99 ± 0.12**	**49.88 ± 0.24**	**40.48 ± 0.36**	**31.82 ± 0.51**	**24.89 ± 0.57**

Table 15: Log-likelihoods for the CIFAR10 adversarial attack experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR10	Adversarial Attack Epsilon						
Log-Likelihood ↑	0.00	0.05	0.10	0.15	0.20	0.25	0.30
MAP	$\mathbf{-0.22 \pm 0.00}$	-4.09 ± 0.01	-4.58 ± 0.02	-4.71 ± 0.03	-4.87 ± 0.04	-5.16 ± 0.05	-5.53 ± 0.06
MAP fVI	-0.25 ± 0.00	$\mathbf{-1.36 \pm 0.01}$	$\mathbf{-1.66 \pm 0.01}$	$\mathbf{-1.97 \pm 0.02}$	$\mathbf{-2.34 \pm 0.03}$	$\mathbf{-2.71 \pm 0.04}$	$\mathbf{-3.01 \pm 0.04}$
MC Dropout	$\mathbf{-0.17 \pm 0.00}$	-3.26 ± 0.01	-4.42 ± 0.02	-4.61 ± 0.02	-4.60 ± 0.02	-4.69 ± 0.05	-4.96 ± 0.07
MC Dropout fVI	-0.25 ± 0.00	$\mathbf{-1.82 \pm 0.01}$	$\mathbf{-2.19 \pm 0.01}$	$\mathbf{-2.37 \pm 0.02}$	$\mathbf{-2.52 \pm 0.02}$	$\mathbf{-2.70 \pm 0.03}$	$\mathbf{-2.90 \pm 0.04}$
Ensemble	$\mathbf{-0.15 \pm 0.00}$	-2.58 ± 0.01	-3.61 ± 0.01	-3.91 ± 0.01	-4.07 ± 0.01	-4.21 ± 0.02	-4.32 ± 0.03
Ensemble fVI	-0.21 ± 0.00	$\mathbf{-1.49 \pm 0.00}$	$\mathbf{-1.98 \pm 0.01}$	$\mathbf{-2.30 \pm 0.01}$	$\mathbf{-2.59 \pm 0.01}$	$\mathbf{-2.82 \pm 0.02}$	$\mathbf{-3.00 \pm 0.02}$
Radial	$\mathbf{-0.21 \pm 0.00}$	-5.52 ± 0.01	-6.49 ± 0.02	-6.78 ± 0.04	-7.01 ± 0.05	-7.30 ± 0.07	-7.56 ± 0.09
Radial fVI	-0.28 ± 0.00	$\mathbf{-1.50 \pm 0.01}$	$\mathbf{-1.76 \pm 0.01}$	$\mathbf{-2.01 \pm 0.01}$	$\mathbf{-2.29 \pm 0.02}$	$\mathbf{-2.57 \pm 0.03}$	$\mathbf{-2.83 \pm 0.03}$
Rank1	-0.33 ± 0.00	-7.84 ± 0.02	-9.30 ± 0.03	-9.39 ± 0.03	-9.23 ± 0.04	-9.08 ± 0.06	-9.00 ± 0.07
Rank1 fVI	$\mathbf{-0.27 \pm 0.00}$	$\mathbf{-1.46 \pm 0.01}$	$\mathbf{-1.75 \pm 0.00}$	$\mathbf{-2.04 \pm 0.01}$	$\mathbf{-2.34 \pm 0.02}$	$\mathbf{-2.66 \pm 0.03}$	$\mathbf{-2.93 \pm 0.04}$

Table 16: Expected calibration errors for the CIFAR10 adversarial attack experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR10	Adversarial Attack Epsilon						
ECE ↓	0.00	0.05	0.10	0.15	0.20	0.25	0.30
MAP	$\mathbf{0.03 \pm 0.00}$	0.48 ± 0.00	0.53 ± 0.00	0.55 ± 0.00	0.58 ± 0.00	0.61 ± 0.00	0.65 ± 0.01
MAP fVI	0.05 ± 0.00	$\mathbf{0.19 \pm 0.00}$	$\mathbf{0.24 \pm 0.00}$	$\mathbf{0.30 \pm 0.00}$	$\mathbf{0.38 \pm 0.01}$	$\mathbf{0.45 \pm 0.01}$	$\mathbf{0.52 \pm 0.01}$
MC Dropout	$\mathbf{0.01 \pm 0.00}$	0.43 ± 0.00	0.53 ± 0.00	0.53 ± 0.00	0.52 ± 0.00	0.52 ± 0.00	0.54 ± 0.01
MC Dropout fVI	0.06 ± 0.00	$\mathbf{0.28 \pm 0.00}$	$\mathbf{0.32 \pm 0.00}$	$\mathbf{0.34 \pm 0.00}$	$\mathbf{0.35 \pm 0.00}$	$\mathbf{0.38 \pm 0.01}$	$\mathbf{0.41 \pm 0.01}$
Ensemble	$\mathbf{0.01 \pm 0.00}$	0.39 ± 0.00	0.51 ± 0.00	0.54 ± 0.00	0.56 ± 0.00	0.57 ± 0.00	0.58 ± 0.00
Ensemble fVI	0.07 ± 0.00	$\mathbf{0.20 \pm 0.00}$	$\mathbf{0.27 \pm 0.00}$	$\mathbf{0.32 \pm 0.00}$	$\mathbf{0.37 \pm 0.00}$	$\mathbf{0.41 \pm 0.00}$	$\mathbf{0.45 \pm 0.00}$
Radial	$\mathbf{0.03 \pm 0.00}$	0.62 ± 0.00	0.70 ± 0.00	0.71 ± 0.00	0.73 ± 0.00	0.74 ± 0.00	0.75 ± 0.01
Radial fVI	0.05 ± 0.00	$\mathbf{0.21 \pm 0.00}$	$\mathbf{0.26 \pm 0.00}$	$\mathbf{0.30 \pm 0.00}$	$\mathbf{0.36 \pm 0.00}$	$\mathbf{0.42 \pm 0.01}$	$\mathbf{0.47 \pm 0.01}$
Rank1	$\mathbf{0.04 \pm 0.00}$	0.76 ± 0.00	0.86 ± 0.00	0.86 ± 0.00	0.83 ± 0.00	0.81 ± 0.00	0.80 ± 0.00
Rank1 fVI	0.05 ± 0.00	$\mathbf{0.20 \pm 0.00}$	$\mathbf{0.25 \pm 0.00}$	$\mathbf{0.30 \pm 0.00}$	$\mathbf{0.36 \pm 0.00}$	$\mathbf{0.43 \pm 0.01}$	$\mathbf{0.49 \pm 0.01}$

Table 17: Accuracies for the corrupted CIFAR100 experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR100	Corruption Severity					
Accuracy ↑	0	1	2	3	4	5
MAP	$\mathbf{75.68 \pm 0.07}$	64.17 ± 0.06	55.41 ± 0.07	49.78 ± 0.07	43.11 ± 0.08	33.03 ± 0.08
MAP fVI	74.77 ± 0.09	$\mathbf{64.26 \pm 0.07}$	$\mathbf{55.82 \pm 0.09}$	$\mathbf{50.31 \pm 0.11}$	$\mathbf{43.56 \pm 0.12}$	$\mathbf{33.81 \pm 0.11}$
MC Dropout	$\mathbf{74.15 \pm 0.07}$	$\mathbf{63.23 \pm 0.06}$	$\mathbf{54.04 \pm 0.09}$	$\mathbf{48.33 \pm 0.08}$	$\mathbf{41.63 \pm 0.08}$	$\mathbf{32.02 \pm 0.09}$
MC Dropout fVI	71.53 ± 0.12	61.03 ± 0.10	51.94 ± 0.11	46.46 ± 0.10	39.88 ± 0.09	30.87 ± 0.10
Ensemble	$\mathbf{79.13 \pm 0.05}$	$\mathbf{68.00 \pm 0.05}$	$\mathbf{59.19 \pm 0.06}$	$\mathbf{53.42 \pm 0.07}$	$\mathbf{46.45 \pm 0.06}$	$\mathbf{35.69 \pm 0.06}$
Ensemble fVI	75.89 ± 0.06	66.38 ± 0.07	57.97 ± 0.09	52.23 ± 0.09	45.14 ± 0.09	35.19 ± 0.11
Radial	$\mathbf{76.40 \pm 0.08}$	63.76 ± 0.07	54.68 ± 0.06	49.02 ± 0.05	42.29 ± 0.07	31.89 ± 0.07
Radial fVI	75.29 ± 0.10	$\mathbf{64.84 \pm 0.11}$	$\mathbf{56.49 \pm 0.12}$	$\mathbf{50.96 \pm 0.11}$	$\mathbf{44.22 \pm 0.10}$	$\mathbf{34.43 \pm 0.09}$
Rank1	73.68 ± 0.10	63.48 ± 0.09	55.34 ± 0.12	49.92 ± 0.11	43.45 ± 0.10	33.87 ± 0.11
Rank1 fVI	$\mathbf{75.56 \pm 0.10}$	$\mathbf{65.49 \pm 0.08}$	$\mathbf{57.55 \pm 0.11}$	$\mathbf{52.24 \pm 0.10}$	$\mathbf{45.63 \pm 0.11}$	$\mathbf{35.73 \pm 0.11}$

Table 18: Log-likelihoods for the corrupted CIFAR100 experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR100	Corruption Severity					
Log-Likelihood ↑	0	1	2	3	4	5
MAP	**−1.00 ± 0.00**	**−1.59 ± 0.00**	−2.09 ± 0.01	−2.48 ± 0.01	−2.99 ± 0.01	−3.73 ± 0.01
MAP fVI	−1.20 ± 0.00	−1.69 ± 0.00	**−2.08 ± 0.01**	**−2.35 ± 0.01**	**−2.69 ± 0.01**	**−3.19 ± 0.01**
MC Dropout	**−0.97 ± 0.00**	**−1.51 ± 0.00**	**−2.02 ± 0.01**	−2.42 ± 0.01	−2.96 ± 0.01	−3.76 ± 0.02
MC Dropout fVI	−1.17 ± 0.00	−1.65 ± 0.01	−2.09 ± 0.01	**−2.39 ± 0.01**	**−2.79 ± 0.01**	**−3.35 ± 0.01**
Ensemble	**−0.81 ± 0.00**	**−1.30 ± 0.00**	**−1.70 ± 0.00**	**−1.98 ± 0.00**	**−2.37 ± 0.01**	**−2.93 ± 0.01**
Ensemble fVI	−1.18 ± 0.00	−1.61 ± 0.00	−1.98 ± 0.00	−2.24 ± 0.00	−2.58 ± 0.00	−3.06 ± 0.01
Radial	**−0.98 ± 0.00**	**−1.66 ± 0.01**	−2.21 ± 0.01	−2.65 ± 0.02	−3.21 ± 0.02	−4.09 ± 0.03
Radial fVI	−1.21 ± 0.00	−1.69 ± 0.00	**−2.08 ± 0.01**	**−2.34 ± 0.01**	**−2.67 ± 0.01**	**−3.16 ± 0.00**
Rank1	−1.48 ± 0.01	−2.29 ± 0.01	−3.01 ± 0.01	−3.59 ± 0.02	−4.38 ± 0.02	−5.57 ± 0.02
Rank1 fVI	**−1.17 ± 0.00**	**−1.64 ± 0.00**	**−2.00 ± 0.01**	**−2.25 ± 0.01**	**−2.58 ± 0.01**	**−3.06 ± 0.01**

Table 19: Expected calibration errors for the corrupted CIFAR100 experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR100	Corruption Severity					
ECE ↓	0	1	2	3	4	5
MAP	**0.08 ± 0.00**	0.12 ± 0.00	0.16 ± 0.00	0.20 ± 0.00	0.23 ± 0.00	0.30 ± 0.00
MAP fVI	0.12 ± 0.00	**0.11 ± 0.00**	**0.09 ± 0.00**	**0.06 ± 0.00**	**0.04 ± 0.00**	**0.04 ± 0.00**
MC Dropout	**0.02 ± 0.00**	0.06 ± 0.00	0.10 ± 0.00	0.13 ± 0.00	0.17 ± 0.00	0.24 ± 0.00
MC Dropout fVI	0.08 ± 0.00	**0.06 ± 0.00**	**0.02 ± 0.00**	**0.01 ± 0.00**	**0.05 ± 0.00**	**0.10 ± 0.00**
Ensemble	**0.05 ± 0.00**	**0.04 ± 0.00**	**0.02 ± 0.00**	**0.02 ± 0.00**	**0.04 ± 0.00**	0.10 ± 0.00
Ensemble fVI	0.18 ± 0.00	0.18 ± 0.00	0.16 ± 0.00	0.14 ± 0.00	0.10 ± 0.00	**0.04 ± 0.00**
Radial	**0.09 ± 0.00**	0.14 ± 0.00	0.19 ± 0.00	0.23 ± 0.00	0.27 ± 0.00	0.34 ± 0.00
Radial fVI	0.13 ± 0.00	**0.12 ± 0.00**	**0.10 ± 0.00**	**0.08 ± 0.00**	**0.05 ± 0.00**	**0.04 ± 0.00**
Rank1	0.16 ± 0.00	0.22 ± 0.00	0.28 ± 0.00	0.32 ± 0.00	0.37 ± 0.00	0.44 ± 0.00
Rank1 fVI	**0.13 ± 0.00**	**0.13 ± 0.00**	**0.11 ± 0.00**	**0.09 ± 0.00**	**0.06 ± 0.00**	**0.03 ± 0.00**

Table 20: Accuracies for the CIFAR100 adversarial attack experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR100	Adversarial Attack Epsilon						
Accuracy ↑	0.00	0.05	0.10	0.15	0.20	0.25	0.30
MAP	**75.68 ± 0.07**	11.84 ± 0.13	6.70 ± 0.08	4.98 ± 0.08	4.18 ± 0.07	3.62 ± 0.09	3.18 ± 0.09
MAP fVI	74.77 ± 0.09	**15.83 ± 0.11**	**9.79 ± 0.12**	**7.08 ± 0.09**	**5.38 ± 0.05**	**4.30 ± 0.08**	**3.46 ± 0.07**
MC Dropout	**74.05 ± 0.08**	19.55 ± 0.10	10.56 ± 0.08	7.45 ± 0.09	5.90 ± 0.09	4.91 ± 0.06	4.01 ± 0.07
MC Dropout fVI	71.61 ± 0.10	**21.60 ± 0.07**	**13.22 ± 0.09**	**9.52 ± 0.09**	**7.19 ± 0.12**	**5.62 ± 0.11**	**4.49 ± 0.11**
Ensemble	**79.13 ± 0.05**	26.04 ± 0.05	13.51 ± 0.09	8.64 ± 0.07	6.32 ± 0.06	**4.92 ± 0.07**	**3.97 ± 0.07**
Ensemble fVI	75.89 ± 0.06	**29.32 ± 0.06**	**16.00 ± 0.08**	**10.09 ± 0.07**	**6.77 ± 0.05**	4.82 ± 0.05	3.66 ± 0.06
Radial	**76.42 ± 0.08**	12.05 ± 0.08	7.36 ± 0.06	5.57 ± 0.05	4.52 ± 0.07	3.77 ± 0.07	3.19 ± 0.08
Radial fVI	75.29 ± 0.10	**16.85 ± 0.12**	**10.97 ± 0.09**	**7.91 ± 0.09**	**6.06 ± 0.10**	**4.83 ± 0.12**	**3.99 ± 0.10**
Rank1	73.76 ± 0.07	14.01 ± 0.09	9.54 ± 0.08	7.63 ± 0.12	6.18 ± 0.12	5.09 ± 0.14	4.20 ± 0.13
Rank1 fVI	**75.58 ± 0.09**	**18.34 ± 0.08**	**11.68 ± 0.07**	**8.65 ± 0.10**	**6.67 ± 0.09**	**5.29 ± 0.08**	**4.38 ± 0.08**

Table 21: Log-likelihoods for the CIFAR100 adversarial attack experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR100	Adversarial Attack Epsilon						
Log-Likelihood ↑	0.00	0.05	0.10	0.15	0.20	0.25	0.30
MAP	$\mathbf{-1.00 \pm 0.00}$	-6.64 ± 0.03	-6.72 ± 0.03	-6.53 ± 0.03	-6.52 ± 0.05	-6.64 ± 0.06	-6.76 ± 0.08
MAP fVI	-1.20 ± 0.00	$\mathbf{-5.36 \pm 0.01}$	$\mathbf{-5.62 \pm 0.02}$	$\mathbf{-5.53 \pm 0.02}$	$\mathbf{-5.45 \pm 0.02}$	$\mathbf{-5.40 \pm 0.02}$	$\mathbf{-5.36 \pm 0.02}$
MC Dropout	$\mathbf{-0.97 \pm 0.00}$	-5.02 ± 0.01	-5.93 ± 0.02	-6.08 ± 0.02	-6.23 ± 0.02	-6.46 ± 0.03	-6.71 ± 0.05
MC Dropout fVI	-1.17 ± 0.00	$\mathbf{-4.36 \pm 0.01}$	$\mathbf{-5.17 \pm 0.02}$	$\mathbf{-5.35 \pm 0.01}$	$\mathbf{-5.45 \pm 0.01}$	$\mathbf{-5.55 \pm 0.01}$	$\mathbf{-5.66 \pm 0.02}$
Ensemble	$\mathbf{-0.81 \pm 0.00}$	$\mathbf{-3.68 \pm 0.00}$	$\mathbf{-4.41 \pm 0.00}$	$\mathbf{-4.65 \pm 0.01}$	$\mathbf{-4.81 \pm 0.01}$	$\mathbf{-4.93 \pm 0.02}$	$\mathbf{-5.01 \pm 0.02}$
Ensemble fVI	-1.18 ± 0.00	-3.97 ± 0.00	-4.86 ± 0.01	-5.10 ± 0.01	-5.15 ± 0.01	-5.17 ± 0.01	-5.18 ± 0.01
Radial	$\mathbf{-0.98 \pm 0.00}$	-6.73 ± 0.02	-6.80 ± 0.03	-6.69 ± 0.03	-6.79 ± 0.05	-7.04 ± 0.07	-7.28 ± 0.10
Radial fVI	-1.21 ± 0.00	$\mathbf{-5.14 \pm 0.01}$	$\mathbf{-5.36 \pm 0.02}$	$\mathbf{-5.29 \pm 0.02}$	$\mathbf{-5.22 \pm 0.02}$	$\mathbf{-5.18 \pm 0.02}$	$\mathbf{-5.16 \pm 0.02}$
Rank1	-1.48 ± 0.00	-9.99 ± 0.02	-10.69 ± 0.02	-10.76 ± 0.02	-10.81 ± 0.04	-10.93 ± 0.05	-11.12 ± 0.07
Rank1 fVI	$\mathbf{-1.17 \pm 0.00}$	$\mathbf{-4.95 \pm 0.01}$	$\mathbf{-5.26 \pm 0.01}$	$\mathbf{-5.22 \pm 0.01}$	$\mathbf{-5.17 \pm 0.01}$	$\mathbf{-5.16 \pm 0.01}$	$\mathbf{-5.16 \pm 0.02}$

Table 22: Expected calibration errors for the CIFAR100 adversarial attack experiment. Means and standard errors over ten seeds. Best results within archetype in boldface, best results overall in blue.

CIFAR100	Adversarial Attack Epsilon						
ECE ↓	0.00	0.05	0.10	0.15	0.20	0.25	0.30
MAP	$\mathbf{0.08 \pm 0.00}$	0.55 ± 0.00	0.52 ± 0.00	0.49 ± 0.00	0.49 ± 0.01	0.49 ± 0.01	0.51 ± 0.01
MAP fVI	0.12 ± 0.00	$\mathbf{0.23 \pm 0.00}$	$\mathbf{0.23 \pm 0.00}$	$\mathbf{0.22 \pm 0.00}$	$\mathbf{0.23 \pm 0.00}$	$\mathbf{0.23 \pm 0.01}$	$\mathbf{0.23 \pm 0.01}$
MC Dropout	$\mathbf{0.03 \pm 0.00}$	0.45 ± 0.00	0.47 ± 0.00	0.44 ± 0.00	0.43 ± 0.00	0.43 ± 0.00	0.44 ± 0.01
MC Dropout fVI	0.09 ± 0.00	$\mathbf{0.26 \pm 0.00}$	$\mathbf{0.29 \pm 0.00}$	$\mathbf{0.28 \pm 0.00}$	$\mathbf{0.28 \pm 0.00}$	$\mathbf{0.29 \pm 0.00}$	$\mathbf{0.30 \pm 0.00}$
Ensemble	$\mathbf{0.05 \pm 0.00}$	0.26 ± 0.00	0.31 ± 0.00	0.30 ± 0.00	0.30 ± 0.00	0.30 ± 0.01	0.29 ± 0.01
Ensemble fVI	0.18 ± 0.00	$\mathbf{0.08 \pm 0.00}$	$\mathbf{0.14 \pm 0.00}$	$\mathbf{0.15 \pm 0.00}$	$\mathbf{0.16 \pm 0.00}$	$\mathbf{0.17 \pm 0.00}$	$\mathbf{0.18 \pm 0.00}$
Radial	$\mathbf{0.09 \pm 0.00}$	0.57 ± 0.00	0.54 ± 0.00	0.51 ± 0.00	0.51 ± 0.01	0.53 ± 0.01	0.55 ± 0.01
Radial fVI	0.13 ± 0.00	$\mathbf{0.21 \pm 0.00}$	$\mathbf{0.21 \pm 0.00}$	$\mathbf{0.20 \pm 0.00}$	$\mathbf{0.20 \pm 0.00}$	$\mathbf{0.20 \pm 0.01}$	$\mathbf{0.19 \pm 0.01}$
Rank1	0.16 ± 0.00	0.76 ± 0.00	0.76 ± 0.00	0.73 ± 0.00	0.71 ± 0.00	0.71 ± 0.00	0.70 ± 0.01
Rank1 fVI	$\mathbf{0.13 \pm 0.00}$	$\mathbf{0.20 \pm 0.00}$	$\mathbf{0.21 \pm 0.00}$	$\mathbf{0.20 \pm 0.00}$	$\mathbf{0.20 \pm 0.00}$	$\mathbf{0.20 \pm 0.00}$	$\mathbf{0.20 \pm 0.01}$