A Compact Representation for Bayesian Neural Networks By Removing Permutation Symmetry

Tim Z. Xiao[1,2] **Weiyang Liu**[3,4] **Robert Bamler**[1]
zhenzhong.xiao@uni-tuebingen.de wl396@cam.ac.uk robert.bamler@uni-tuebingen.de
[1]University of Tübingen [2]IMPRS-IS [3]University of Cambridge
[4]Max Planck Institute for Intelligent Systems, Tübingen

Abstract

Bayesian neural networks (BNNs) are a principled approach to modeling predictive uncertainties in deep learning, which are important in safety-critical applications. Since exact Bayesian inference over the weights in a BNN is intractable, various approximate inference methods exist, among which sampling methods such as Hamiltonian Monte Carlo (HMC) are often considered the gold standard. While HMC provides high-quality samples, it lacks interpretable summary statistics because its sample mean and variance is meaningless in neural networks due to permutation symmetry. In this paper, we first show that the role of permutations can be meaningfully quantified by a number of transpositions metric. We then show that the recently proposed rebasin method [1] allows us to summarize HMC samples into a compact representation that provides a meaningful explicit uncertainty estimate for each weight in a neural network, thus unifying sampling methods with variational inference. We show that this compact representation allows us to compare trained BNNs directly in weight space across sampling methods and variational inference, and to efficiently prune neural networks trained without explicit Bayesian frameworks by exploiting uncertainty estimates from HMC.

1 Introduction and Background

When training a neural network on a data set $\mathcal{D}$, one minimizes a loss function $\mathcal{L}(\mathbf{W}, \mathcal{D})$ over weights and biases (collectively referred to as "weights" and denoted as boldface $\mathbf{W}$ in the following). Yet, when comparing two trained networks (e.g., to choose a training algorithm or learning rate), one rarely compares the trained weights directly. Instead, one compares various performance metrics of the two trained networks on a held-out data set. Direct comparisons in weight space are difficult, in part because standard distance metrics (e.g., the euclidean distance) are not meaningful in weight space due to a permutation symmetry of neural networks [7, 3, 4, 1]: consider two consecutive layers of a neural network, which represent a function $x \mapsto \sigma_2(W_2\sigma_1(W_1 x + b_1) + b_2)$, where $W_{1,2}$, $b_{1,2}$, and $\sigma_{1,2}$ are weight matrices, bias vectors, and (componentwise applied) activation functions, respectively. It is easy to see that this function does not change if we consider an arbitrary permutation matrix P and replace the weights and biases by $W_1' := PW_1$, $b_1' := Pb_1$, and $W_2' := W_2P^{-1}$. Thus, euclidean distances in weight space like $||W_1 - W_1'||_2^2$ are not meaningful.

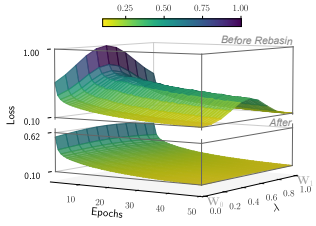

Figure 1: Training dynamics for models with $\mathbf{W}_0$ and $\mathbf{W}_1$, and their interpolations $\mathbf{W}_\lambda$.

Recent works [1, 4] propose and analyze a method called rebasin, which efficiently finds a permutation that brings the weights $\mathbf{W}_1$ of one neural network as close as possible to the weights $\mathbf{W}_0$ of an independently trained reference network with the same architecture. The authors show that ap-

Preprint.

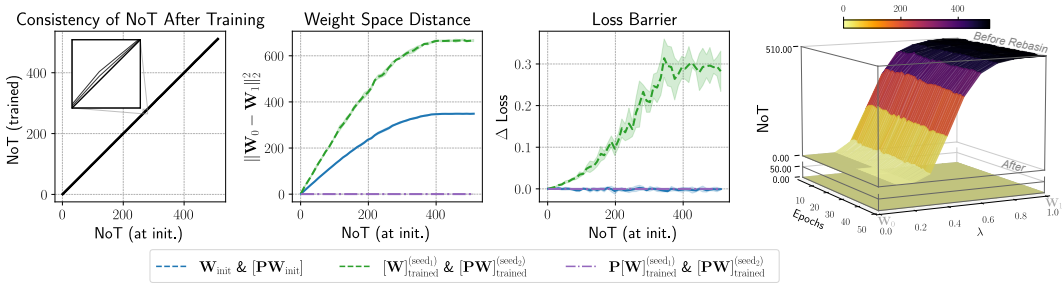

Figure 2: Left three: effect of permuting initial weights by different Number of Transpositions (NoT) on NoT after training, weight-space distance, and loss barrier (shaded regions: $\pm 1\sigma$ over 5 runs). Right: NoT changes monotonically along the interpolation $\mathbf{W}_\lambda$ between two models $\mathbf{W}_0$ and $\mathbf{W}_1$.

plying this permutation essentially removes the loss barrier [5] when linearly interpolating between the weights of the two neural networks (see Figure 1 for before and after rebasin, where λ controls the interpolation $\mathbf{W}_\lambda := (\lambda - 1)\mathbf{W}_0 + \lambda \mathbf{W}_1$). The authors conjecture that the loss landscape is quasi-convex up to permutations. In this paper, we build on these findings in three ways

1. We propose to quantify rebasin permutations by their number of transpositions (NoT), see below. We show empirically that NoT is a valuable metric for analyzing weight-space geometry as it is remarkably stable over training and correlates strongly with the loss barrier.
2. We argue that a quasi-convex loss landscape is particularly useful for Bayesian neural networks (BNNs), which aim to find (an approximation of) the posterior distribution $p(\mathbf{W} \mid \mathcal{D})$ over weights $\mathbf{W}$ that are consistent with the data $\mathcal{D}$. We show empirically that rebasin allows us to summarize the approximate posterior of sampling based inference methods like Hamiltonian Monte Carlo (HMC) in the same compact representation that variational inference (VI) uses, thus enabling direct comparisons across inference methods.
3. We show that the proposed unifying compact representation of BNNs is interpretable. For example, unusual for sampling methods in BNNs, we obtain meaningful explicit uncertainty estimates in weight space from HMC. We show that we can use these uncertainty estimates from HMC to efficiently prune a neural network trained with deep ensembles [12].

The paper is structured as follows: we introduce and empirically analyzes the NoT metric (item 1 above) in Section 2, discuss BNNs (items 2 and 3 above) in Section 3, and conclude in Section 4.

2 Quantifying Permutations in Weight Space by Number of Transpositions

We briefly introduce and analyze a metric to quantify the "magnitude" of permutations obtained from rebasin [1], which proved useful for building up intuition and for debugging implementations of experiments discussed in Section 3. Readers only interested in BNNs may opt to skip this part.

We propose to measure the magnitude of a permutation by its number of transpositions (NoT), i.e., the minimal number of pairwise swaps whose consecutive execution results in the given permutation. It is a well-known result from algebra [9] that this is always possible, and that NoT can be calculated efficiently by first factorizing a permutation into non-overlapping cycles and then expressing each cycle of length k as a product of $(k-1)$ transpositions. For example, the permutation P that maps $1 \mapsto 4$, $2 \mapsto 1$, $3 \mapsto 5$, $4 \mapsto 2$, and $5 \mapsto 3$ can be written as $P = (1\ 4\ 2)\,(3\ 5) = (1\ 4)\,(4\ 2)\,(3\ 5)$, where $(a_1, a_2, \ldots, a_k)$ denotes a cycle $a_1 \mapsto a_2 \mapsto \cdots \mapsto a_k \mapsto a_1$. Thus, $\mathrm{NoT}(P) = 3$.

We find empirically that NoT is a meaningful metric for analyzing weight space geometry. We consider a neural network for classification of MNIST digits with a single hidden layer and a total of 512 hidden activations that can be permuted. We trained a network with randomly initialized weights $\mathbf{W}_{\text{init}}$ and a set of networks whose weights were initialized as $P\mathbf{W}_{\text{init}}$ where P is a random permutation with $\mathrm{NoT}(P)$ ranging over all values from zero to 511. After training both neural networks, we obtained a permutation P' by matching the two trained networks using rebasin.

As a first high-level check, Figure 2 (left) shows $\mathrm{NoT}(P')$ as a function of $\mathrm{NoT}(P)$ for three random $\mathbf{W}_{\text{init}}$. Even though the permuted and unpermuted networks were trained with different ran-

dom seeds for the sampling of minibatches, NoT remains almost exactly unaffected by training. Further, Figure 2 (center two) show that both the euclidean distance in weight space and the barrier of the loss function after training correlate strongly with NoT. Here, the barrier is defined as in [5] as $\max_{\lambda\in[0,1]} \mathcal{L}(\mathbf{W}_\lambda, \mathcal{D}) - \frac{1}{2}\big(\mathcal{L}(\mathbf{W}_0, \mathcal{D}) + \mathcal{L}(\mathbf{W}_1, \mathcal{D})\big)$, where $\mathbf{W}_\lambda := (\lambda - 1)\mathbf{W}_0 + \lambda\mathbf{W}_1$, and $\mathbf{W}_0$ and $\mathbf{W}_1$ are the weights of the two trained models. Finally, Figure 2 (right) analyzes $\mathrm{NoT}(\tilde{P})$ both over training epochs and along the linear interpolation $\lambda \in [0, 1]$, where $\tilde{P}$ is obtained by matching $\mathbf{W}_\lambda$ to $\mathbf{W}_0$ using rebasin. We observe that the $\mathrm{NoT}(\tilde{P})$ remains flat in the vicinity of either trained model, and changes smoothly and monotonically in between.

From these observations, we conclude that, while comparing neural network weights by euclidean distance alone is not meaningful (see Section 1), we can meaningfully quantify weight-space distances by a pair $\big(||\mathbf{W}_0 - P\mathbf{W}_1||_2^2, \mathrm{NoT}(P)\big)$ where the permutation P matches $\mathbf{W}_1$ to $\mathbf{W}_0$ by rebasin.

3 A Unifying Compact Representation for Bayesian Neural Networks

Building on the argument [14, 11, 18] that removing permutation degrees of freedom is particularly useful for *Bayesian* Neural Networks (BNNs), we propose a framework to combine the strengths of two classes of inference algorithms in BNNs. Training a BNN amounts to finding (an approximation of) the so-called posterior distribution $p(\mathbf{W} \mid \mathcal{D})$ of all weights $\mathbf{W}$ that are consistent with the training data $\mathcal{D}$, and thus involves more than a single set of weights. We show below that being able to meaningfully compare weights makes approximate posteriors of BNNs more interpretable.

The exact posterior distribution is $p(\mathbf{W} \mid \mathcal{D}) = p(\mathbf{W})p(\mathcal{D} \mid \mathbf{W})/p(\mathcal{D})$, where $p(\mathbf{W})$ is a prior distribution that acts like a regularizer (often an isotropic Gaussian), $p(\mathcal{D} \mid \mathbf{W}) = \exp[-\mathcal{L}(\mathbf{W}, \mathcal{D})]$, and $p(\mathcal{D}) = \int p(\mathbf{W})p(\mathcal{D} \mid \mathbf{W})\,\mathrm{d}\mathbf{W}$. The exact posterior is usually intractable in BNNs, but various efficient approximation methods have been developed. We group them into two categories:

(a) **Parametric methods**, such as variational inference (VI; [2]) and Laplace approximation [15, 10] approximate the posterior $p(\mathbf{W} \mid \mathcal{D})$ explicitly with a simpler distribution $q(\mathbf{W})$, e.g., a fully factorized normal distribution with fitted means and variances;

(b) **Sampling methods**, such as Hamiltonian Monte Carlo (HMC; [13]), stochastic gradient Langevin dynamics [17], and MCDropout [6] draw a set of K samples $\{\mathbf{W}^{(k)}\}_{k=1}^K$ directly from $p(\mathbf{W} \mid \mathcal{D})$ without explicit representing their distribution; deep ensembles [12] is sometimes also considered in this context despite not following the Bayesian framework.

As sampling methods (b) lack explicit uncertainty information, one might be tempted to fit samples with a parametric distribution, e.g., a Gaussian $q_\mathrm{d}(\mathbf{W}) = \mathcal{N}(\boldsymbol{\mu}_\mathrm{d}, \mathrm{diag}(\boldsymbol{\sigma}_\mathrm{d}^2))$, where $\boldsymbol{\mu}_\mathrm{d}$ and $\boldsymbol{\sigma}_\mathrm{d}^2$ are the sample mean and variance, and the subscript 'd' is for 'direct'. However, we show in Section 3.1 that q_d is a poor posterior approximation, likely because the posterior is multimodal due to the permutation symmetry. However, the quasi-convexity conjecture [1, 4] suggests that the posterior is unimodal once we remove the permutation degrees of freedom. Therefore, we propose to summarize samples from, e.g., HMC, with a diagonal Gaussian $q_\mathrm{r}(\mathbf{W}) = \mathcal{N}(\boldsymbol{\mu}_\mathrm{r}, \mathrm{diag}(\boldsymbol{\sigma}_\mathrm{r}^2))$ where $\boldsymbol{\mu}_\mathrm{r}$ and $\boldsymbol{\sigma}_\mathrm{r}^2$ are the sample mean and variance after using rebasin [1] to match each sample to an arbitrary shared reference sample. We show in Section 3.1 that q_r approximates the posterior well despite its compactness, and in Section 3.2 that having a unifying compact representation allows us to combine the respective strengths of different inference methods, thus going beyond the findings in [18].

All experiments were done with a simple fully connected network for MNIST classification with a single hidden layer of size 512. We compare q_d and q_r across HMC, deep ensemble, and VI.

3.1 Compact Representation as an Approximate Posterior

Before we use $q_\mathrm{r}(\mathbf{W})$ to compare models directly in weight space, we first evaluate whether it provides a good approximation of the posterior despite radically reducing the amount of information available in the samples. Evaluations of BNNs are typically done by comparing the predictive distribution $p(\boldsymbol{y}^* \mid \boldsymbol{x}^*, \mathcal{D}) = \int p(\boldsymbol{y}^* \mid \boldsymbol{x}^*, \mathbf{W})p(\mathbf{W} \mid \mathcal{D})\,\mathrm{d}\mathbf{W}$ on a test set $\mathcal{D}_\mathrm{test}$, where $\mathbf{x}^*$ is the input, $\mathbf{y}^*$ is the prediction, and we assume that $p(\boldsymbol{y}^* \mid \boldsymbol{x}^*, \mathbf{W})$ is described by the neural network. There are two popular metrics to compare predictive distributions between a method p and HMC (which is often considered the most precise approximation of a BNN posterior [8]): *agreement* and

Table 1: Performance of different BNNs (q_d: before rebasin; q_r: after rebasin) on their agreement (Eq. (1)) and total variation (TV; Eq. (2)) to HMC samples, and on their test set accuracy.

	HMC			Ensemble			VI
	Sample	$q_\mathrm{d}(\mathbf{W})$	$q_\mathrm{r}(\mathbf{W})$	Sample	$q_\mathrm{d}(\mathbf{W})$	$q_\mathrm{r}(\mathbf{W})$	$q(\mathbf{W})$
(↑) Agreement with HMC samples	1.	0.1212	0.8249	0.9931	0.5239	0.9868	0.9885
(↓) TV to HMC samples	0.	0.8641	0.6570	0.0229	0.7210	0.0495	0.0235
Test Accuracy (%) of Samples	98.43	11.11	82.34	98.66	52.25	97.72	98.11
Test Accuracy (%) of $\boldsymbol{\mu}_\mathrm{d}$ and $\boldsymbol{\mu}_\mathrm{r}$	N/A	28.06	92.25	N/A	86.40	97.97	98.04

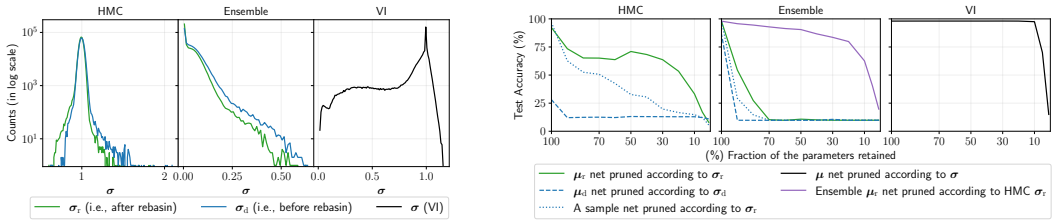

Figure 3: Left: histograms of the standard deviation $\boldsymbol{\sigma}$ of weights before ($\boldsymbol{\sigma}_\mathrm{d}$) and after ($\boldsymbol{\sigma}_\mathrm{r}$) rebasin. Right: test accuracy vs. various levels of weight pruning (retaining only weights with lowest $\boldsymbol{\sigma}$).

total variation (TV) [19] (in the following, $I[\,\cdot\,]$ is the indicator function),

$$\mathrm{Agree.}(p, p_\mathrm{HMC}) = \frac{1}{|\mathcal{D}_\mathrm{test}|} \sum_{\boldsymbol{x}^* \in \mathcal{D}_\mathrm{test}} I\Big[\arg\max_{\boldsymbol{y}^*} p(\boldsymbol{y}^* \mid \boldsymbol{x}^*, \mathcal{D}) = \arg\max_{\boldsymbol{y}^*} p_\mathrm{HMC}(\boldsymbol{y}^* \mid \boldsymbol{x}_i^*, \mathcal{D})\Big]; \quad (1)$$

$$\mathrm{TV}(p, p_\mathrm{HMC}) = \frac{1}{|\mathcal{D}_\mathrm{test}|} \sum_{\boldsymbol{x}^* \in \mathcal{D}_\mathrm{test}} \frac{1}{2} \sum_{\boldsymbol{y}^*} \big|p(\boldsymbol{y}^* \mid \boldsymbol{x}_i^*, \mathcal{D}) - p_\mathrm{HMC}(\boldsymbol{y}^* \mid \boldsymbol{x}_i^*, \mathcal{D})\big|. \quad (2)$$

Results in Table 1 show that: (i) $q_\mathrm{r}(\mathbf{W})$ has much better performance than $q_\mathrm{d}(\mathbf{W})$ for both HMC and ensemble; thus, rebasin is crucial for obtaining a accurate compact representation; (ii) ensemble outperforms HMC in $q_\mathrm{r}(\mathbf{W})$, which could indicate issues of the activation matching algorithm [1] when applied between networks with different loss levels, e.g., within HMC samples; and (iii) $q_\mathrm{r}(\mathbf{W})$ provides a parameter efficient representation for ensemble with competitive performance.

3.2 Comparing and Merging BNNs in Weight Space

Figure 3 (left) shows histograms of the variances $\boldsymbol{\sigma}_\mathrm{d}$ and $\boldsymbol{\sigma}_\mathrm{r}$ for HMC and ensemble, and the variances fitted by VI. We observe that (i) permuting all samples into the same basin reduces variances overall, as expected; and (ii) the two Bayesian methods (HMC and VI) have lots of weights with a variance close to one, which is not affected by rebasin in HMC. This indicates mode collapse since we used a standard Gaussian prior, i.e., the Bayesian methods identify these weights as unnecessary.

The proposed unifying compact representation allows us to merge different BNNs by stitching the means $\boldsymbol{\mu}_\mathrm{r}$ from one model with the variances $\boldsymbol{\sigma}_\mathrm{r}^2$ from another. Figure 3 (right) shows test accuracies of neural networks after pruning weights with high $\boldsymbol{\sigma}_\mathrm{r}^2$ (a simplified variant of the compression method in [20, 16]). The purple curve uses weights $\boldsymbol{\mu}_\mathrm{r}$ from ensemble but $\boldsymbol{\sigma}_\mathrm{r}^2$ from HMC, thus combining the predictive strength of ensemble with the accurate uncertainty estimates of HMC. It significantly outperforms both variants that use only the ensemble or only HMC (green curves).

4 Conclusion

When doing Bayesian inference in BNNs, it is straightforward to go from parametric based to sampling based inference, e.g., one can easily draw samples from a variational distribution. But permutation symmetry makes it difficult to go in the reverse direction. In this work, we use the recently proposed rebasin method to remove the permutation symmetry. We propose a unifying compact representation for Bayesian inference in BNNs, which allows us to go from sampling based inference to parametric based inference, and to combine the respective strengths of different inference methods.

Acknowledgments and Disclosure of Funding

The authors would like to thank Yingzhen Li, Takeru Miyato, Johannes Zenn, Nicolò Zottino and Andi Zhang for helpful discussions. Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC number 2064/1 – Project number 390727645. This work was supported by the German Federal Ministry of Education and Research (BMBF): Tübingen AI Center, FKZ: 01IS18039A. Robert Bamler acknowledges funding by the German Research Foundation (DFG) for project 448588364 of the Emmy Noether Programme. The authors thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting Tim Z. Xiao.

References

[1] Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. *arXiv preprint arXiv:2209.04836*, 2022. 1, 2, 3, 4

[2] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR, 2015. 3

[3] Johanni Brea, Berfin Simsek, Bernd Illing, and Wulfram Gerstner. Weight-space symmetry in deep networks gives rise to permutation saddles, connected by equal-loss valleys across the loss landscape. *arXiv preprint arXiv:1907.02911*, 2019. 1

[4] Rahim Entezari, Hanie Sedghi, Olga Saukh, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. *arXiv preprint arXiv:2110.06296*, 2021. 1, 3

[5] Jonathan Frankle, Gintare Karolina Dziugaite, Daniel Roy, and Michael Carbin. Linear mode connectivity and the lottery ticket hypothesis. In *International Conference on Machine Learning*, pages 3259–3269. PMLR, 2020. 2, 3

[6] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016. 3

[7] Robert Hecht-Nielsen. On the algebraic structure of feedforward network weight spaces. In *Advanced Neural Computers*, pages 129–135. Elsevier, 1990. 1

[8] Pavel Izmailov, Sharad Vikram, Matthew D Hoffman, and Andrew Gordon Gordon Wilson. What are bayesian neural network posteriors really like? In *International conference on machine learning*, pages 4629–4640. PMLR, 2021. 3

[9] Anthony W Knapp. *Basic algebra*. Springer Science & Business Media, 2007. 2

[10] Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being bayesian, even just a bit, fixes overconfidence in relu networks. In *International conference on machine learning*, pages 5436–5446. PMLR, 2020. 3

[11] Richard Kurle, Tim Januschowski, Jan Gasthaus, and Yuyang Bernie Wang. On symmetries in variational bayesian neural nets. 2021. 3

[12] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017. 2, 3

[13] Radford M Neal et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011. 3

[14] Arya A Pourzanjani, Richard M Jiang, and Linda R Petzold. Improving the identifiability of neural networks for bayesian inference. In *NIPS Workshop on Bayesian Deep Learning*, volume 4, page 31, 2017. 3

[15] Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018. 3

[16] Zipei Tan and Robert Bamler. Post-Training Neural Network Compression With Variational Bayesian Quantization. In *Advances in Neural Information Processing Systems*, Workshop on Challenges in Deploying and Monitoring Machine Learning Systems, 2022. 4

[17] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688, 2011. 3

[18] Jonas Gregor Wiese, Lisa Wimmer, Theodore Papamarkou, Bernd Bischl, Stephan Günnemann, and David Rügamer. Towards efficient mcmc sampling in bayesian neural networks by exploiting symmetry. *arXiv preprint arXiv:2304.02902*, 2023. 3

[19] Andrew Gordon Wilson, Pavel Izmailov, Matthew D Hoffman, Yarin Gal, Yingzhen Li, Melanie F Pradier, Sharad Vikram, Andrew Foong, Sanae Lotfi, and Sebastian Farquhar. Evaluating approximate inference in bayesian deep learning. In *NeurIPS 2021 Competitions and Demonstrations Track*, pages 113–124. PMLR, 2022. 4

[20] Yibo Yang, Robert Bamler, and Stephan Mandt. Variational bayesian quantization. In *International Conference on Machine Learning*, pages 10670–10680. PMLR, 2020. 4

A Experiment Setup

We use a fully connected network with a single hidden layer of size 512 for all models in MNIST classification setting. More specifically, for deep ensemble, we use 5 randomly initialized members, trained with Adam for 50 epochs using a maximum a posteriori (MAP) with standard Gaussian prior. For VI, the network has double the size of parameters since we need to model both the mean and variance. It is also trained with Adam for 50 epochs. When evaluating their predictive distribution, for both ensemble and VI, we use 100 samples in Monte Carlo integration.

For HMC, we use 600 epochs for burn-in, then we save a sample for every 10 epochs. Each epoch uses 500 leapfrog steps. In total, we generated 1000 HMC samples for the evaluations in the paper.