Data augmentation in Bayesian neural networks and the cold posterior effect

Abstract

Bayesian neural networks that incorporate data augmentation implicitly use a "randomly perturbed log-likelihood [which] does not have a clean interpretation as a valid likelihood function" (Izmailov et al. 2021). Here, we provide several approaches to developing principled Bayesian neural networks incorporating data augmentation. We introduce a "finite orbit" setting which allows valid likelihoods to be computed exactly, and for the more usual "full orbit" setting we derive multi-sample bounds tighter than those used previously for Bayesian neural networks with data augmentation. These models cast light on the origin of the cold posterior effect. In particular, we find that the cold posterior effect persists even in these principled models incorporating data augmentation. This suggests that the cold posterior effect cannot be dismissed as an artifact of data augmentation using incorrect likelihoods.

1 INTRODUCTION

The cold posterior effect [CPE; Wenzel et al., 2020] is the surprising observation that performance in neural networks is not optimal when we use the usual Bayesian posterior [Kolmogorov, 1950, Savage, 1954, Jaynes, 2003],

$$\mathrm{P}\left(\mathbf{w}|\mathbf{y},\mathbf{X}\right) \propto \mathrm{P}\left(\mathbf{w}\right)\mathrm{P}\left(\mathbf{y}|\mathbf{w},\mathbf{X}\right) \tag{1}$$

where $\mathbf{w}$ are the neural network weights, $\mathbf{X}$ is all inputs (typically images), and $\mathbf{y}$ is all outputs (typically class labels). Instead, we get better performance when using a "cold" posterior, i.e. the posterior taken to the power of $1/T$ where $T < 1$,

$$\mathrm{Q}\left(\mathbf{w}\right) \propto \left(\mathrm{P}\left(\mathbf{w}\right)\mathrm{P}\left(\mathbf{y}|\mathbf{w},\mathbf{X}\right)\right)^{1/T}. \tag{2}$$

The origin of the CPE is by now highly contentious, with three leading potential explanations [Noci et al., 2021]. The first hypothesis is that the process of data curation for popular datasets such as CIFAR-10 and ImageNet [Krizhevsky et al., 2009, Deng et al., 2009] involves multiple annotators agreeing upon the label for each image. In that case, there are in effect multiple labels for each image, which inflates the likelihood (but not the prior) term causing a "cooler" posterior [Adlam et al., 2020, Aitchison, 2020]. Second, the prior is always misspecified, and prior misspecification is known to induce cold posterior-like effects in specific (non-neural network) models [Grünwald, 2012, Grünwald et al., 2017, Adlam et al., 2020], which might give an explanation for the CPE in neural networks [Wenzel et al., 2020, Fortuin et al., 2021b]. However, Fortuin et al. [2021b] showed that better priors do not always reduce the size of the CPE, but can actually increase it. In particular, they found that incorporating spatial correlations in convolutional filters improved the performance of a ResNet trained on CIFAR-10, but also increased the magnitude of the CPE. The third possible explanation is that the CPE is an artifact of data augmentation [DA; Wenzel et al., 2020, Izmailov et al., 2021], as DA gives a "randomly perturbed log-likelihood [which] does not have a clean interpretation as a valid likelihood function" [Izmailov et al., 2021]. This is supported by observations in which the CPE only exists with DA, and disappears without DA [Wenzel et al., 2020, Fortuin et al., 2021b, Izmailov et al., 2021]. Of course it is quite possible that practical CPEs arise from a complex combination of these causes [Aitchison, 2020, Noci et al., 2021].

In spite of this controversy, recent work on the CPE agrees that it is important to investigate integrating DA with Bayesian neural networks (BNNs), and to examine the interaction with the CPE. From Noci et al. [2021]: "It remains an interesting open problem how to properly account for data augmentation in a Bayesian sense." And from Izmailov et al. [2021]: "Data augmentation cannot be naively incorporated in the Bayesian neural network model." and "We leave incorporating data augmentation ... as an exciting direction of future work."

Perhaps the most common understanding of the interaction

Submitted to the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022). **To be used for reviewing only**.

between the CPE and DA in BNNs is that DA increases the effective dataset size. From Izmailov et al. [2021]: "intuitively, data augmentation increases the amount of data observed by the model, and should lead to higher posterior contraction". From Osawa et al. [2019]: "DA increases the effective sample size". From, Noci et al. [2021]: "while data augmentation may increase the amount of data seen by the model, that increase is certainly not equal to the number of times each data point is augmented (after all, augmented data is not independent from the original data)."

In this work, we seek to understand whether the commonly used, but invalid DA likelihood can cause the CPE. Our contributions are as follows.

1. We give a formal argument that the notion that DA increases the effective dataset size is flawed (Sec. 3.1).
2. We derive a set of multi-sample lower bounds on the log-likelihood of a BNN incorporating DA (Sec. 3.2). These bounds are tighter than existing single-sample estimators for BNNs [e.g. Wenzel et al., 2020] and can be applied to a broad class of likelihood functions [unlike Van der Wilk et al., 2018].
3. We introduce a "finite orbit"[1] setting with a small number of admissible augmentations which allows us to compute *exact* log-likelihoods (Sec. 3.3).
4. We empirically evaluate the performance of our multi-sample bounds in both SGD training and BNN inference for image classification tasks (Sec. 4). In the latter case we explore the impact of both the bounds and the exact finite orbit log-likelihood on the CPE.
5. We find that the CPE persists even when using these principled DA likelihood bounds. This falsifies the hypothesis that the CPE is an artifact of loose bounds on the log-likelihood given by previous single-sample estimators.

We finish with some discussion summarising our findings and their reflection on the CPE (Sec. 6). Our conclusion in this work is that the CPE is not an artifact resulting from DA giving "randomly perturbed log-likelihood"s [Izmailov et al., 2021].

Note that in the remainder of the paper, we will follow Izmailov et al. [2021] in regarding models with loose, single-sample bounds as "unprincipled" (from Izmailov et al. [2021], the "randomly perturbed log-likelihood does not have a clean interpretation as a valid likelihood function"). In contrast, we term models using our exact log-likelihoods or our multi-sample bounds as being "principled".

[1] We employ the term "orbit" from group theory and function invariance [Kondor, 2008], even though our augmentations do not always form groups. In this work, it refers to the support of augmentation distribution $\mathrm{P}(\mathbf{x}'|\mathbf{x})$.

2 BACKGROUND

2.1 DATA AUGMENTATION

In supervised learning, we are interested in learning some unknown functional relationship from example input-output pairs $(\mathbf{x}_i, y_i), i = 1, \ldots, N$. Often, we have information about some form of invariance, i.e. the knowledge that the function does not change its output for certain transformations of the input. These might occasionally be true invariances, such as the identity of a molecule being invariant to rotations. But in most settings, these are so-called "soft" invariances or "insensitivities" [van der Wilk et al., 2018]. For instance, the class label for an image should not change due to small translations/crops of that image (but might change if we radically alter the image). The most basic form of DA takes advantage of this information by transforming, or augmenting, the inputs and copying the output value, to create additional input-output pairs which are then included in training. Often, the amount of additional "augmented data" can be unbounded, for example when allowable transformations are specified in a continuous range, e.g. rotations. This simple procedure has been very successful in improving performance in a wide variety of machine learning methods [Loosli et al., 2007, Krizhevsky et al., 2012, Bishop, 2006], and recent work has analysed the effect of data augmentation on invariances in the learned functions [Dao et al., 2019, Chen et al., 2020, Lyle et al., 2020].

2.2 BAYESIAN INFERENCE

Bayesian inference allows us to infer a distribution over neural network weights, which incorporates uncertainty induced by having finite data. Bayes prescribes a strict procedure for updating beliefs about unknown quantities in light of observed data. The model is specified by a prior on the weights $\mathrm{P}(\mathbf{w})$ and a log-likelihood, $\sum_{i=1}^{N} \mathcal{L}^i(y_i; \mathbf{w})$. Thus, the log-posterior is given by

$$\log \mathrm{P}\left(\mathbf{w}|\mathbf{X}, \mathbf{y}\right) = \log \mathrm{P}\left(\mathbf{w}\right) + \sum_{i=1}^{N} \mathcal{L}^i(\mathbf{w}) + \mathrm{const}\,. \quad (3)$$

We define the no augmentation log-likelihood as

$$\mathcal{L}^i_{\text{noaug}}(y_i; \mathbf{w}) = \log \mathrm{P}\left(y_i | g(\mathbf{x}_i; \mathbf{w})\right). \quad (4)$$

where $g(\cdot; \mathbf{w})$ is the neural network.

3 METHODS

3.1 ARGUMENTS AGAINST PREVIOUS APPROACHES TO DA

Many authors have claimed that DA increases the effective dataset size [Noci et al., 2021, Osawa et al., 2019, Izmailov

et al., 2021]. Here we argue that this view leads to problems within the framework of probabilistic modelling. We can see this in the form of the resulting log-likelihood. For K augmented inputs, $\mathbf{x}'_{i;k}$, we can write the log-likelihood for a single underlying image as,

$$\mathcal{L}^i_{\text{add}}(y_i; \mathbf{w}) = \sum_{k=1}^{K} \log \mathrm{P}\left(y_i \middle| g(\mathbf{x}'_{i;k}; \mathbf{w})\right). \tag{5}$$

For continuous transformations such as rotations, there are an infinite number of possible augmentations, $K = \infty$, so $\mathcal{L}_{\text{add}}$ would result in the prior being ignored during inference. While this result seems strange, if the outputs for all augmentations $\mathbf{x}'_{i;k}$ were independently labelled, we would indeed have an infinitely large (conditionally) IID dataset and ignoring the prior would be the right answer. However, in practice, the unaugmented input $\mathbf{x}_i$ is labelled by an annotator and the result y_i is assumed to apply to all augmentations $\mathbf{x}'_{i;k}$. As such, the labels for different augmentations of the same input are not independent, and an approach (such as this one) which assumes they are cannot be valid.

A method which avoids having to specify the augmented dataset size is to average the log-likelihood over augmentations

$$\mathcal{L}^i_{\text{loss}}(y_i; \mathbf{w}) = \mathbb{E}\left[\log \mathrm{P}\left(y_i | g(\mathbf{x}'_i; \mathbf{w})\right)\right]. \tag{6}$$

Indeed, most implementations which use DA when training BNNs target this log-likelihood, at least implicitly. They do so by taking a pre-existing inference algorithm and replacing the original input, $\mathbf{x}_i$, with a random augmentation, $\mathbf{x}'_i$. This approach is convenient, as a single sample from the augmentation distribution can provide an unbiased estimate $\hat{\mathcal{L}}_{\text{loss}} = \log \mathrm{P}\left(y_i | g(\mathbf{x}'_i; \mathbf{w})\right)$. Importantly though, a valid likelihood should arise from a valid distribution over labels, and should therefore normalize if we sum over labels. For instance, without augmentation,

$$1 = \textstyle\sum_{y_i=1}^{Y} \exp\left(\mathcal{L}^i_{\text{noaug}}(y_i; \mathbf{w})\right). \tag{7}$$

However, if we try to interpret $\mathcal{L}^i_{\text{loss}}(y_i; \mathbf{w})$ as a log-likelihood we find that it does not normalize to 1,

$$1 \neq \sum_{y_i=1}^{Y} \exp\left(\mathcal{L}^i_{\text{loss}}(y_i; \mathbf{w})\right). \tag{8}$$

and therefore $\mathcal{L}^i_{\text{loss}}(y_i; \mathbf{w})$ cannot be the log of a valid probability distribution. Note that we might try to get a valid likelihood by including a normalizer. The problem is that this normalizer would need to depend on $\mathbf{w}$, and thus would need to be included in the log-likelihood, and of course no normalizer terms appear in the loss Eq. (6).

3.2 TIGHT LOWER BOUNDS ON THE LOG-LIKELIHOOD OF PRINCIPLED DA MODELS

To incorporate data augmentation into likelihoods, we define the probabilities for each class as being averages over augmentations. We can choose to either average logits (equal to the neural network outputs, $\mathbf{f}(\cdot; \mathbf{w})$) or predictive probabilities ($\operatorname{softmax} \mathbf{f}(\cdot; \mathbf{w})$),

$$\mathbf{p}_{\text{inv}}(\mathbf{x}_i; \mathbf{w}) = \mathbb{E}\left[\operatorname{softmax} \mathbf{f}(\mathbf{x}'_i; \mathbf{w})\right], \tag{9}$$

$$\mathbf{f}_{\text{inv}}(\mathbf{x}_i; \mathbf{w}) = \mathbb{E}\left[\mathbf{f}(\mathbf{x}'_i; \mathbf{w})\right]. \tag{10}$$

where we take expectations over $\mathrm{P}(\mathbf{x}'_i | \mathbf{x}_i)$. Remember that $\mathbf{f}(\mathbf{x}'_i; \mathbf{w})$ is the (vector-valued) neural network output for an augmented input, which is used as the logits in classification, so $\mathbf{f}_{\text{inv}}(\mathbf{x}_i; \mathbf{w})$ is the outputs averaged over all augmentations of the same underlying image. Likewise, $\mathbf{p}_{\text{inv}}(\mathbf{x}_i; \mathbf{w})$ is the vector of probabilities given by averaging the predicted probabilities over augmentations. These are denoted "inv" for invariant, because averaging over augmentations can give invariances in $\mathbf{f}_{\text{inv}}(\mathbf{x}_i; \mathbf{w})$ and $\mathbf{p}_{\text{inv}}(\mathbf{x}_i; \mathbf{w})$ that are not present in the underlying neural network, $\mathbf{f}(\mathbf{x}_i; \mathbf{w})$. The resulting (usually intractable) log-likelihoods are

$$\begin{aligned}\mathcal{L}^i_{\text{prob}}(y_i; \mathbf{w}) &= \log \mathrm{P}_{\text{prob}}(y_i | \mathbf{x}_i, \mathbf{w}) \\ &= \log \mathbb{E}\left[\operatorname{softmax}_{y_i} \mathbf{f}(\mathbf{x}'_i; \mathbf{w})\right], \end{aligned} \tag{11}$$

$$\begin{aligned}\mathcal{L}^i_{\text{logits}}(y_i; \mathbf{w}) &= \log \mathrm{P}_{\text{logits}}(y_i | \mathbf{x}_i, \mathbf{w}) \\ &= \log \operatorname{softmax}_{y_i} \mathbb{E}\left[\mathbf{f}(\mathbf{x}'_i; \mathbf{w})\right]. \end{aligned} \tag{12}$$

These likelihoods were originally proposed in [Wenzel et al., 2020] for averaging probabilities and [van der Wilk et al., 2018] for averaging logits. However, they are intractable, as it is not (usually) possible to evaluate the expectation under all data augmentations. Instead, we need to choose an estimator or bound on these quantities. [Wenzel et al., 2020] suggested a loose single sample bound for averaging probabilities, and [van der Wilk et al., 2018] suggested an unbiased estimator that is restricted to Gaussian likelihoods in regression. In contrast, we show that we can get tight, intuitive and easy to evaluate, multi-sample bounds analogous to those in [Burda et al., 2015],

$$\begin{aligned}\hat{\mathcal{L}}^i_{\text{prob},K}(y_i; \mathbf{w}) &= \log\left(\tfrac{1}{K}\textstyle\sum_{k=1}^{K} \operatorname{softmax}_{y_i} \mathbf{f}(\mathbf{x}'_{i;k}; \mathbf{w})\right), \\ \hat{\mathcal{L}}^i_{\text{logits},K}(y_i; \mathbf{w}) &= \log \operatorname{softmax}_{y_i}\left(\tfrac{1}{K}\textstyle\sum_{k=1}^{K} \mathbf{f}(\mathbf{x}'_{i;k}; \mathbf{w})\right). \end{aligned} \tag{13}$$

To prove the lower bound for averaging probabilities, we first rewrite the expectation inside the logarithm of (Eq. 11) as the expectation of its average, over K identically distributed random variables, $\mathbf{x}'_{i;k}$. We then take an approach familiar from variational inference [Jordan et al., 1999] by applying Jensen's inequality to the (concave) logarithm

function.

$$\begin{aligned}\mathcal{L}^i_{\text{prob}}\left(y_i;\mathbf{w}\right) &= \log \mathbb{E}\left[\tfrac{1}{K}\textstyle\sum_{k=1}^K \text{softmax}_{y_i}\,\mathbf{f}(\mathbf{x}'_{i;k};\mathbf{w})\right] \\ &\geq \mathbb{E}\left[\log \tfrac{1}{K}\textstyle\sum_{k=1}^K \text{softmax}_{y_i}\,\mathbf{f}(\mathbf{x}'_{i;k};\mathbf{w})\right] \\ &= \mathbb{E}\left[\hat{\mathcal{L}}^i_{\text{prob},K}\left(y_i;\mathbf{w}\right)\right]. \end{aligned} \tag{14}$$

For averaging logits, we follow a similar method, noting that $\log \text{softmax}_{y_i}$ is a concave function [Boyd et al., 2004] taking a vector of logits and returning a scalar log-probability for class y_i. As such, we can again apply Jensen's inequality,

$$\begin{aligned}\mathcal{L}^i_{\text{logits}}\left(y_i;\mathbf{w}\right) &= \log \text{softmax}_{y_i}\,\mathbb{E}\left[\tfrac{1}{K}\textstyle\sum_{k=1}^K \mathbf{f}(\mathbf{x}'_{i;k};\mathbf{w})\right] \\ &\geq \mathbb{E}\left[\log \text{softmax}_{y_i}\,\tfrac{1}{K}\textstyle\sum_{k=1}^K \mathbf{f}(\mathbf{x}'_{i;k};\mathbf{w})\right] \\ &= \mathbb{E}\left[\hat{\mathcal{L}}^i_{\text{logits},K}\left(y_i;\mathbf{w}\right)\right]. \end{aligned} \tag{15}$$

Finally, note that these objectives naturally correspond to the notions of averaging logits or averaging probabilities, which could be motivated using non-probabilistic considerations. Importantly, we do not claim the notion of averaging probabilities or averaging logits for different augmentations as a contribution in itself. We only claim as a contribution the notion that averaging probabilities or logits provide lower-bounds on principled log-likelihoods including data-augmentation, implying they can be used in a principled Bayesian setting, and they are not ruled out despite having some degree of stochasticity.

Increasing K reduces the variance and tightens the bounds which eventually become exact as $K \to \infty$ [Burda et al., 2015].

$$\mathbb{E}\left[\hat{\mathcal{L}}^i_{\text{logits},K}\left(y_i;\mathbf{w}\right)\right] \leq \mathbb{E}\left[\mathcal{L}^i_{\text{logits},K+1}\left(y_i;\mathbf{w}\right)\right] \tag{16}$$

$$\mathcal{L}^i_{\text{logits}}\left(y_i;\mathbf{w}\right) = \lim_{K\to\infty} \hat{\mathcal{L}}^i_{\text{logits},K}\left(y_i;\mathbf{w}\right) \tag{17}$$

$$\mathbb{E}\left[\hat{\mathcal{L}}^i_{\text{prob},K}\left(y_i;\mathbf{w}\right)\right] \leq \mathbb{E}\left[\mathcal{L}^i_{\text{prob},K+1}\left(y_i;\mathbf{w}\right)\right] \tag{18}$$

$$\mathcal{L}^i_{\text{prob}}\left(y_i;\mathbf{w}\right) = \lim_{K\to\infty} \hat{\mathcal{L}}^i_{\text{prob},K}\left(y_i;\mathbf{w}\right) \tag{19}$$

However, larger K introduces greater computational cost. We therefore consider what value of K is likely to be sensible, by plotting the bound against K. We indeed found that the bound increases with K up to around 10, when it saturates (Fig. 1). While these differences might seem small when evaluated purely at test-time, they seem to cause much larger differences when integrated into training (Figs. 2 and 4). In contrast, in VI, practitioners frequently use a single-sample bound. However, VI incorporates a highly effective variance reduction strategy that is absent in our setting: an optimized variational approximate posterior (see Appendix B). In principle, similar variance reduction strategies exist in our setting, but would involve learning a separate variance-reducing augmentation distribution for each image, which is clearly impractical. Indeed, in our setting, $K = 1$ represents

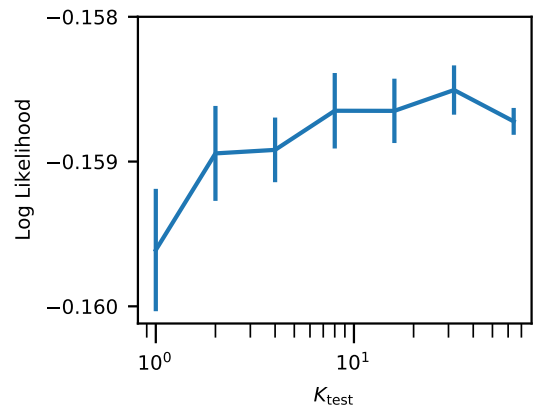


Figure 1: The effect of K_{test} on the log-likelihood bound for a test batch (size 512) of CIFAR-10. Values shown for ResNet20 BNN trained and tested with $\hat{\mathcal{L}}_{\text{prob},K}$ ($K_{\text{train}} = 8$ and $T = 0.001$). Error bars cover two standard errors above/below mean for DA sampling with different seeds. Sixty seeds used for $K_{\text{test}} = \{1, 2\}$, thirty for $K_{\text{test}} = 4$ and five for all other K_{test}.

such a crude approximation that it collapses the differences between averaging probabilities, logits, and losses,

$$\hat{\mathcal{L}}^i_{\text{logits};1}\left(y_i;\mathbf{w}\right) = \hat{\mathcal{L}}^i_{\text{prob};1}\left(y_i;\mathbf{w}\right) = \hat{\mathcal{L}}^i_{\text{loss}}\left(y_i;\mathbf{w}\right) \tag{20}$$

which are all equal to $\log \text{softmax}_{y_i}\,\mathbf{f}(\mathbf{x}'_{i;1};\mathbf{w})$.

3.3 FINITE ORBIT

Finally, all of the above is for the usual "full orbit" setting, where there is a distribution over a very large, or even infinite number of possible augmentations. The full orbit setting necessitates the use of the bound in (Eq. 13), and allows us to use different numbers of samples at test and training time, K_{test} and K_{train} respectively. Remarkably, if we consider an alternative "finite orbit" by restricting the augmentations to a small subset, we can *exactly* evaluate the log-likelihood. In the finite orbit setting, the distribution over augmented images, $\mathbf{x}'_i$, conditioned on the underlying unaugmented image, $\mathbf{x}_i$, can be written as,

$$\text{P}\left(\mathbf{x}'_i|\mathbf{x}_i\right) = \tfrac{1}{K}\textstyle\sum_{k=1}^K \delta\left(\mathbf{x}'_i - a_k(\mathbf{x}_i)\right), \tag{21}$$

where δ is the Dirac-delta, and a_k is a function that applies the kth fixed augmentation. In this setting, it is possible to exactly compute $\mathcal{L}^i_{\text{logits}}(y_i;\mathbf{w})$ and $\mathcal{L}^i_{\text{prob}}(y_i;\mathbf{w})$ by summing over the K augmentations. This allows us to empirically explore how exact log-likelihood computation influences the CPE, comparing it with the bounds in the full-orbit setting (Eq. 13) in Sec. 4.2. When implementing finite orbit augmentation in practice, we choose the K fixed augmentations by sampling them before training. The finite orbit setting uses the same augmentations, and therefore the same number of augmentations, at test and train time: $K_{\text{train}} = K_{\text{test}} = K$.

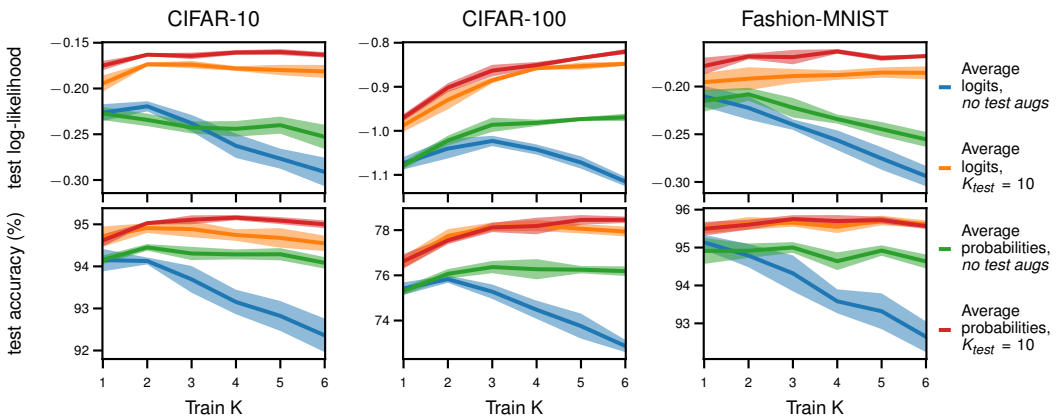


Figure 2: Comparison of averaging logits and probabilities for different values of K_{train}, and using $K_{\text{test}} = 10$ vs. using no test-time augmentations. Here, we use ResNet18 with SGD (i.e. no Bayesian inference). We use only full orbit to decouple K_{train} from K_{test}.

4 RESULTS

4.1 PRINCIPLED DA IN NON-BAYESIAN NETWORKS

We begin by comparing averaging logits and averaging probabilities in a non-Bayesian setting: SGD. Critically, higher values of K_{train} imply a larger computational cost per epoch, as each image is replicated and augmented K_{train} times before going through the network. When assessing the benefit of averaging probabilities/logits over standard DA for SGD training, we must therefore control for computational cost. We do this by training for $200/K_{\text{train}}$ epochs. Note that $K_{\text{train}} = 1$ with no test-time augmentation (i.e. green and blue in Fig. 2) corresponds to the standard DA approach for both averaging logits and averaging probabilities (Eq. 20). In this experiment, we consider only full orbit, which unlike finite orbit allows us to decouple K_{train} and K_{test}.

We trained ResNet18[2] on CIFAR-10, CIFAR-100 [Krizhevsky et al., 2009][3] and FashionMNIST [Xiao et al., 2017][4] with a learning rate of 0.1, decayed to 0.01 three quarters of the way through training. We apply the same two augmentation transformations as Wenzel et al. [2020], Fortuin et al. [2021b], Noci et al. [2021]: 1. a random crop with padding of four pixels on all borders and 2. a random horizontal flip with probability 0.5. The training runs took around 12 GPU-days on Nvidia 2080s.[5]

In agreement with past work [Lyle et al., 2020], we found that averaging over augmentations at test-time (red and orange) is better than using the test image without augmentation (green and blue), with $K_{\text{train}} = 1$ corresponding to the standard DA procedure. In addition, we show that improved performance with multiple test-time augmentations continues to hold for larger values of K_{train}. Thus, if sufficient compute is available at test-time, averaging across augmentations gives an easy method to improve the performance of a pre-trained network.

Importantly, we see some performance gains with higher values of K_{train} if we focus on the case with test augmentations, though they are somewhat inconsistent across datasets. We see strong improvements for the hardest dataset (CIFAR-100), and smaller improvements that saturate at $K_{\text{train}} = 2$ for CIFAR-10. For FashionMNIST, the picture is more mixed. We suspect this is because we used a DA strategy tuned for CIFAR-10 and CIFAR-100, rather than FashionMNIST.

In addition, averaging probabilities seems to give somewhat better performance than averaging logits: compare averaging probabilities vs. logits both with test-time augmentation (red vs. orange) and without test-time augmentation (green vs. blue). The performance differences are consistent in both comparisons, though smaller when test-time augmentation is applied.

Indeed, performance falls quite dramatically as K_{train} increases for averaging logits without test-time augmentation (blue). This is an indication that averaging probabilities and logits might actually behave quite differently. To understand how these differences might arise, consider the effect of averaging on the NN function itself. Both schemes can be justified by using averaging to increase invariance to the augmentation transformations (Sec. 3.2). Averaging probabilities, however, also forces the NN function itself to become invariant. If different augmentations produce different predictions, then the resulting averaged class probabilities will be more uncertain, which is penalized by the likelihood on

[2] `github.com/kuangliu/pytorch-cifar`; MIT Licensed

[3] `cs.toronto.edu/~kriz/cifar.html`

[4] `github.com/zalandoresearch/fashion-mnist`; MIT Licensed

[5] Code available: `anonymous.4open.science/r/Augmentations-1E35/`

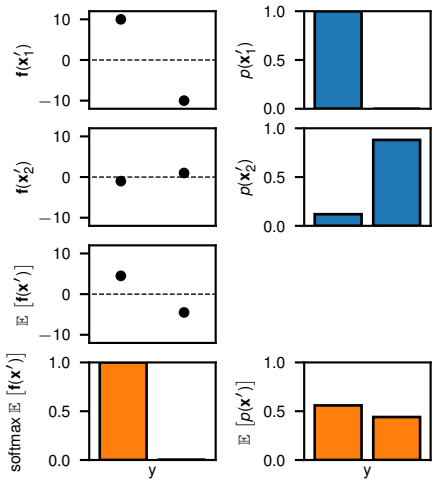


Figure 3: Example effect of averaging logits against averaging probabilities. $\mathbf{x}'_1$ and $\mathbf{x}'_2$ are two augmentations of the same image, $\mathbf{f}(\mathbf{x}'_1)$ and $\mathbf{f}(\mathbf{x}'_2)$ are logits outputted by a NN, and $p(\mathbf{x}'_1)$ and $p(\mathbf{x}'_2)$ are the probabilities corresponding to these logits. The prediction derived from the averaged logits is much more certain than the average of the individual probabilities.

the training points. This effect is much weaker when averaging logits. Consider an extreme example, as illustrated in Fig. 3. It is a two-class classification problem with two augmentations, $\mathbf{x}'_1$ and $\mathbf{x}'_2$, of the same image with logits, $\mathbf{f}(\mathbf{x}'_1) = (10, -10)$ and $\mathbf{f}(\mathbf{x}'_2) = (-1, 1)$. Averaging logits gives us $\mathbb{E}\left[\mathbf{f}(\mathbf{x}')\right] = (4.5, -4.5)$, and applying the softmax, we very confidently predict the first class. In contrast, if we use averaging probabilities, then the first augmentation almost certainly predicts the first class $p(\mathbf{x}'_1) \approx (1, 0)$ and the second augmentation almost certainly predicts the second class, $p(\mathbf{x}'_2) \approx (0, 1)$, so when we average them we obtain $\mathbb{E}\left[p(\mathbf{x}')\right] \approx (0.5, 0.5)$, which indicates a high degree of uncertainty.

4.2 BAYESIAN NEURAL NETWORKS AND THE COLD POSTERIOR EFFECT

Next, we ask a very different question: how is the CPE influenced when DA is incorporated into the model in a principled way? To this end, we use a different experimental setup. In particular, we take the code[6] and networks from Fortuin et al. [2021b,a] and mirror their experimental setup for CIFAR-10 as closely as possible. This code combines a cyclical learning rate schedule [Zhang et al., 2019], a gradient-guided Monte Carlo (GGMC) scheme [Garriga-Alonso and Fortuin, 2021], and the preconditioning and convergence diagnostics from Wenzel et al. [2020]. The DA transformations are the same as those described in Sec. 4.1. Following Fortuin et al. [2021b], we ran 60 cycles with 50 epochs in each cycle. We recorded one sample at the end of each of the last five epochs of a cycle, giving 300 samples total. Importantly, to allow for running many sampling epochs in these experiments, we follow Fortuin et al. [2021b] in using the ResNet20 architecture from Wenzel et al. [2020], which has far fewer channels than the ResNet18 used in Sec. 4.1 (i.e. 32 channels for the first block up to 128 in the last block compared to 64 channels up to 512 [He et al., 2016a]). As such, SGD with this network performs poorly compared with that in Sec. 4.1 (ResNet20, CIFAR-10 accuracy $\sim 92\%$ [Wenzel et al., 2020] vs. ResNet18, CIFAR-10 accuracy $\sim 95\%$ [He et al., 2016b]). The experiments took around 60 GPU-days on Nvidia RTX6000s[7].

The results are presented in Fig. 4. We replicate the finding that the CPE is largely absent without DA (dashed black line), and is present in the standard setup with DA at training time ($K_{\text{train}} = 1$) but without augmentation at test time (solid black). Further, we show that the CPE persists with principled DA likelihoods: averaging logits with full orbit (purple, top row), and averaging probabilities with finite and full orbits (green). The best method overall appears to be averaging probabilities with a full orbit (dark green line, top row) at $T = 0.001$, though averaging logits (dark purple lines) is better at $T = 1$.

Surprisingly, the CPE is absent in one particular setting: averaging logits with a finite orbit (purple line, bottom row). However, the relevance of this is unclear, as it is clearly the worst performing of all DA approaches by quite some margin. Indeed, remember that the arguments for the optimality of Bayesian inference apply only in the case that the model is well-specified [Kolmogorov, 1950, Savage, 1954, Jaynes, 2003]. However, the poor performance of averaging logits with a finite orbit indicates that it is likely to be the wrong model, while other settings are likely to be closer to the true model. In that case, the presence or absence of the CPE in the wrong model (averaging logits with a finite orbit) is immaterial to our understanding of the CPE in the right model. Note that this argument could not be made if there was a model without the CPE with performance equal to or better than the other models (see Sec. 6 for further discussion).

The CPE was originally discovered in Wenzel et al. [2020] when assessing test accuracy and log-likelihood — they did not consider other measures of distribution calibration like expected calibration error (ECE). Indeed, later work on the CPE found that measures such as ECE are far more complex and usually do not agree with test accuracy and log-likelihood [Fortuin et al., 2021a]. It is therefore difficult to interpret the differences between test log-likelihod and ECE, especially if we remember that test log-likelihood is

[6] `github.com/ratschlab/bnn_priors`; MIT Licensed

[7] Code available: `anonymous.4open.science/r/projects2020_BNN-priors-B869/experiments/bayes_data_aug/README.md`

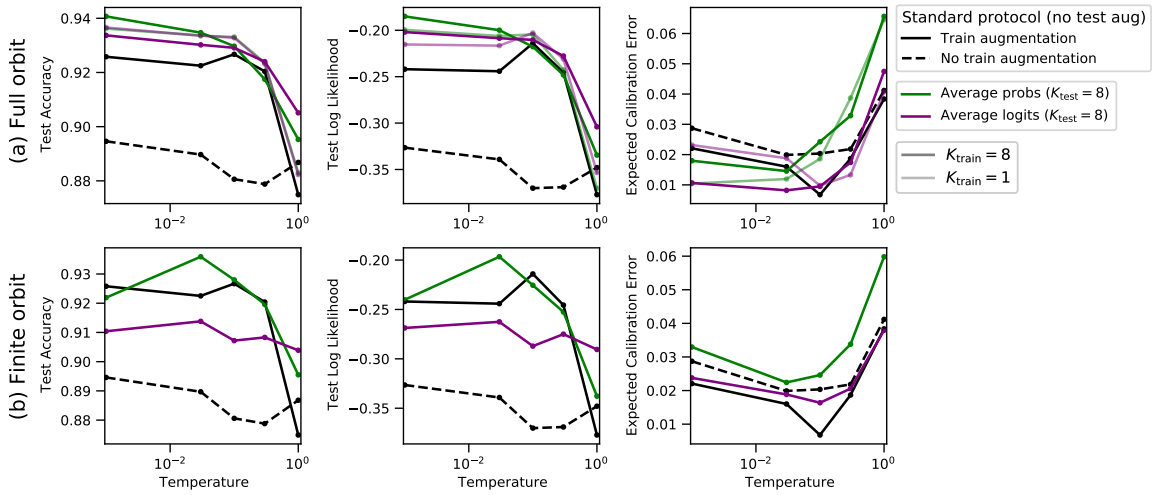


Figure 4: The cold posterior effect for different DA setups when running GGMC with ResNet20 on CIFAR-10. Without DA, there is a minimal CPE. Most other configurations show significant improvement for $T < 1$, with the exception of averaging the logits over a finite orbit. Averages computed with $K_{\text{train}} = 8$ and $K_{\text{test}} = 8$.

itself a proper scoring rule [Gneiting and Raftery, 2007], and therefore captures one possible notion of calibration. In particular, test log-likelihood heavily penalizes an event assessed as low probability actually happening, e.g. if our classifier predicts a probability of 0.001%, while the actually happens even 0.1% of the time. In contrast, ECE considers the absolute difference in probability, so it far more heavily penalises e.g. a predicted probability of 40% while the event actually happens 60% of the time. Needless to say, the most appropriate measure of calibration will depend heavily on the domain, with log-likelihood being more appropriate for low-probability but high risk events. In our case, averaging probabilities (green) achieves the greatest log-likelihood scores, standard DA (solid black) achieves the lowest ECE.

The usefulness of our results is contingent on understanding whether we are indeed accurately approximating the posterior. To check this, we computed the kinetic temperature [Leimkuhler and Matthews, 2015], which estimates the temperature of a given parameter in the Langevin dynamics simulation from the norm of its momentum. In expectation, the kinetic temperature estimator should be equal to the desired temperature, T. The results (Appendix C) indicate that all the samplers run at their desired temperature, a result that is consistent with accurate posterior sampling.

As discussed in Sec. 3, increasing K tightens our log-likelihood bounds, but incurs greater computational cost. It is natural to question which value of K is a good trade-off. We explore how the log-likelihood of test data under a trained model varies with K_{test}. As expected, the results (Fig. 1) show the log-likelihood increases with K_{test}, with even $K = 2$ being a significant improvement over $K = 1$ (standard DA). However, the curve plateaus, suggesting that for CIFAR-10, there is little benefit of using $K > 8$.

5 RELATED WORK

Past work introduced noisy-input generative models which average probabilities [Wenzel et al., 2020]. However, this work did not consider the tighter multi-sample bounds developed here, or the finite orbit setting which allows us to evaluate the exact likelihood. This left open the possibility raised by Izmailov et al. [2021] that the CPE was an artifact of standard DA resulting in an invalid likelihood. In contrast, we considered exact likelihoods in the finite orbit setting, and tighter multi-sample lower bounds in the full orbit setting. Further, the invariant function perspective allowed us to derive a log-likelihood bound for averaging logits, not considered by Wenzel et al. [2020]. As the CPE persists when using our principled DA models, we can exclude the possibility that the CPE is an artifact of DA giving a "randomly perturbed log-likelihood". Other work has introduced a log-likelihood estimator for averaging GP logits using the invariance principle [van der Wilk et al., 2018]. However, the method only works for a quadratic log-likelihood and thus necessitates Pólya-Gamma approximations for classification. Further, the work did not consider BNNs or the connection to the CPE.

There is a small but growing body of work that considers averaging over multiple augmentations at training time [Hoffer et al., 2019, Berman et al., 2019, Choi et al., 2019, Benton et al., 2020, Lyle et al., 2020, Touvron et al., 2021, Fort et al., 2021]. However, this work was not done within a Bayesian framework (e.g. by using stochastic gradient Langevin dynamics (SGLD) or a similar inference algorithm), did not show that averaging across multiple training augmentations gives a multi-sample bound on the log-likelihood of a principled model, did not consider the finite-orbit setting where the log-likelihood can be computed exactly, and did not

consider the interaction with the CPE. In addition, much of this work uses averaging losses [Hoffer et al., 2019, Berman et al., 2019, Choi et al., 2019, Benton et al., 2020, Touvron et al., 2021, Fort et al., 2021] which is equivalent to using a loose single-sample bound on the log-likelihoods. While Lyle et al. [2020] show that feature averaging during training can improve generalization, our work is, to the best of our knowledge, the first to average predicted probabilities at training time. Finally, the idea of averaging at test-time is more common and has been practiced for longer [e.g. Krizhevsky et al., 2012, Simonyan and Zisserman, 2014, He et al., 2015, Szegedy et al., 2015, Foster et al., 2020].

A considerable body of past work on BNNs uses DA, both with variational inference [Blundell et al., 2015] and SGLD [e.g. Zhang et al., 2018, 2019, Osawa et al., 2019, Fortuin et al., 2021b, Immer et al., 2021]. However, as discussed in Sec. 2 (Background), these methods simply substitute non-augmented for augmented data and thus do not use a valid log-likelihood. In contrast, we incorporated DA into the probabilistic generative model, and thus are able to give valid log-likelihoods based on averaging logits or averaging probabilities in the classification case.

6 CONCLUSION

We have shown how DA can be properly incorporated into a generative model suitable for inference in BNNs, by deriving a lower-bound on the log-likelihood of a DA-invariant function. Empirically, we have seen that the CPE persists even when using our principled DA formulation, and in agreement with past work [Wenzel et al., 2020, Fortuin et al., 2021b, Izmailov et al., 2021], we show that the CPE disappears without DA.

What do these results imply for the origin of the CPE? Our models in principle have a clean log-likelihood which can be evaluated exactly in the finite orbit setting, or which we estimate using tightened multi-sample bounds in the full orbit setting. This falsifies the hypothesis, that the CPE is an artifact arising from DA giving a "randomly perturbed log-likelihood [which] does not have a clean interpretation as a valid likelihood function".

Indeed, it is worth stepping back and considering the original motivation for studying the CPE, namely that if we have the correct model, then Bayesian inference with $T = 1$ should give optimal performance [Kolmogorov, 1950, Savage, 1954, Jaynes, 2003, Wenzel et al., 2020]. Critically, we need the right model for us to expect optimal performance at $T = 1$. We now have two classes of model, with DA and without DA, so which is right(er)? Given the significant and widely recognised performance benefits of DA, it seems very likely that the "right" model would include some form of DA. If the model with DA is right(er), and that model displays the CPE, then the CPE still demands an explanation, and the presence or absence of the CPE in the wrong model without DA is immaterial. As such, the presence of the CPE in models with DA remains an important problem, and is likely to be caused by one of the two other explanations discussed in Sec. 1 (Introduction): either data curation [Aitchison, 2020] or prior misspecification [Wenzel et al., 2020, Fortuin et al., 2021b]. Indeed, we would tentatively suggest the opposite of Izmailov et al. [2021]: that it is in reality the *lack* of a CPE without DA that is an artifact of using the wrong model (i.e. without DA).

Finally, note that the CPE is not always observed, e.g. in language classification [Izmailov et al., 2021]. This is absolutely expected as the data-curation explanation of Aitchison [2020] only implies CPE in fairly restricted settings; i.e. *only* in the case of reasonably accurate approximate posterior inference, such as SGLD, in a BNN where the data has been curated by excluding datapoints with an ambiguous class-label. Thus, Aitchison [2020] does not lead us to expect the CPE e.g. in latent variable models, in regression settings (where you typically do not curate data), or in hybrid models where we perform Bayesian inference over only a small subset of parameters.

References

Ben Adlam, Jasper Snoek, and Samuel L Smith. Cold posteriors and aleatoric uncertainty. *arXiv preprint arXiv:2008.00029*, 2020.

Laurence Aitchison. A statistical theory of cold posteriors in deep neural networks. *arXiv preprint arXiv:2008.05912*, 2020.

Gregory Benton, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. Learning invariances in neural networks. *arXiv preprint arXiv:2010.11882*, 2020.

Maxim Berman, Hervé Jégou, Andrea Vedaldi, Iasonas Kokkinos, and Matthijs Douze. Multigrain: a unified image embedding for classes and instances. *arXiv preprint arXiv:1902.05509*, 2019.

Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

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.

Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

Shuxiao Chen, Edgar Dobriban, and Jane H. Lee. A group-theoretic framework for data augmentation. *Journal of Machine Learning Research*, 21(245):1–71, 2020. URL http://jmlr.org/papers/v21/20-163.html.

Dami Choi, Alexandre Passos, Christopher J Shallue, and George E Dahl. Faster neural network training with data echoing. *arXiv preprint arXiv:1907.05550*, 2019.

Andreas C Damianou, Michalis K Titsias, and Neil Lawrence. Variational inference for latent variables and uncertain inputs in gaussian processes. 2016.

Tri Dao, Albert Gu, Alexander Ratner, Virginia Smith, Chris De Sa, and Christopher Re. A kernel theory of modern data augmentation. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 1528–1537. PMLR, 09–15 Jun 2019. URL http://proceedings.mlr.press/v97/dao19b.html.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Stanislav Fort, Andrew Brock, Razvan Pascanu, Soham De, and Samuel L. Smith. Drawing multiple augmentation samples per image during training efficiently decreases test error. *arXiv preprint arXiv:2105.13343*, 2021.

Vincent Fortuin, Adrià Garriga-Alonso, Mark van der Wilk, and Laurence Aitchison. BNNpriors: A library for Bayesian neural network inference with different prior distributions. *Software Impacts*, page 100079, 2021a.

Vincent Fortuin, Adrià Garriga-Alonso, Florian Wenzel, Gunnar Rätsch, Richard Turner, Mark van der Wilk, and Laurence Aitchison. Bayesian neural network priors revisited. *arXiv preprint arXiv:2102.06571*, 2021b.

Adam Foster, Rattana Pukdee, and Tom Rainforth. Improving transformation invariance in contrastive representation learning. *arXiv preprint arXiv:2010.09515*, 2020.

Adrià Garriga-Alonso and Vincent Fortuin. Exact Langevin dynamics with stochastic gradients. *arXiv preprint arXiv:2102.01691*, 2021.

Agathe Girard and Roderick Murray-Smith. Learning a gaussian process model with uncertain inputs. *Department of Computing Science, University of Glasgow, Tech. Rep. TR-2003-144*, 2003.

Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American statistical Association*, 102(477):359–378, 2007.

Peter Grünwald. The safe bayesian. In *International Conference on Algorithmic Learning Theory*, pages 169–183. Springer, 2012.

Peter Grünwald, Thijs Van Ommen, et al. Inconsistency of bayesian inference for misspecified linear models, and a proposal for repairing it. *Bayesian Analysis*, 12(4): 1069–1103, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016b.

Elad Hoffer, Tal Ben-Nun, Itay Hubara, Niv Giladi, Torsten Hoefler, and Daniel Soudry. Augment your batch: better training with larger batches. *arXiv preprint arXiv:1901.09335*, 2019.

Alexander Immer, Matthias Bauer, Vincent Fortuin, Gunnar Rätsch, and Mohammad Emtiyaz Khan. Scalable marginal likelihood estimation for model selection in deep learning. *arXiv preprint arXiv:2104.04975*, 2021.

Pavel Izmailov, Sharad Vikram, Matthew D. Hoffman, and Andrew Gordon Wilson. What are Bayesian neural network posteriors really like? *arXiv:2104.14421*, 2021.

Edwin T Jaynes. *Probability theory: The logic of science*. Cambridge university press, 2003.

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2): 183–233, 1999.

AN Kolmogorov. *Foundations of the theory of probability*. Courier Dover Publications, 1950.

Imre Risi Kondor. *Group theoretical methods in machine learning*. 2008.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

Ben Leimkuhler and Charles Matthews. The canonical distribution and stochastic differential equations. In *Molecular Dynamics*, pages 211–260. Springer, 2015.

Gaëlle Loosli, Stéphane Canu, and Léon Bottou. Training invariant support vector machines using selective sampling. *Large scale kernel machines*, 2, 2007.

Clare Lyle, Mark van der Wilk, Marta Kwiatkowska, Yarin Gal, and Benjamin Bloem-Reddy. On the benefits of invariance in neural networks. *arXiv preprint arXiv:2005.00178*, 2020.

Andrew McHutchon and Carl Rasmussen. Gaussian process training with input noise. *Advances in Neural Information Processing Systems*, 24:1341–1349, 2011.

Lorenzo Noci, Kevin Roth, Gregor Bachmann, Sebastian Nowozin, and Thomas Hofmann. Disentangling the roles of curation, data-augmentation and the prior in the cold posterior effect. *arXiv preprint arXiv:2106.06596*, 2021.

Kazuki Osawa, Siddharth Swaroop, Anirudh Jain, Runa Eschenhagen, Richard E Turner, Rio Yokota, and Mohammad Emtiyaz Khan. Practical deep learning with Bayesian principles. *arXiv preprint arXiv:1906.02506*, 2019.

Leonard J Savage. *The foundations of statistics*. Courier Corporation, 1954.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. *arXiv preprint arXiv:2103.17239*, 2021.

Mark van der Wilk, Matthias Bauer, ST John, and James Hensman. Learning invariances using the marginal likelihood. *arXiv preprint arXiv:1808.05563*, 2018.

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. Citeseer, 2011.

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? *arXiv preprint arXiv:2002.02405*, 2020.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Guodong Zhang, Shengyang Sun, David Duvenaud, and Roger Grosse. Noisy natural gradient as variational inference. In *International Conference on Machine Learning*, pages 5852–5861. PMLR, 2018.

Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical stochastic gradient MCMC for Bayesian deep learning. *arXiv preprint arXiv:1902.03932*, 2019.

A AVERAGING LOSSES EMERGES WHEN USING DA IN VI AND SGLD

There are two particularly important algorithms for doing Bayesian inference in neural networks: stochastic gradient Langevin dynamics [SGLD; Welling and Teh, 2011] and variational inference [VI; Blundell et al., 2015]. In SGLD without DA, we draw samples from the posterior over weights by following gradient of the log-probability with added noise,

$$(\Delta \mathbf{w})_{\text{noaug}} = \frac{\epsilon}{2} \nabla_{\mathbf{w}} \left[\log \mathrm{P}\left(\mathbf{w}\right) + \sum_{i=1}^{N} \log \mathrm{P}_{\text{noaug}}\left(y_i | \mathbf{x}_i, \mathbf{w}\right) \right] + \sqrt{\epsilon}\, \boldsymbol{\eta} \tag{22}$$

where $\boldsymbol{\eta}$ is standard Gaussian IID noise, and for simplicity we give the expression for full-batch Langevin dynamics rather than minibatched SGLD (they do not differ for the purposes of reasoning about DA). Likewise the variational inference objective is,

$$\text{ELBO}_{\text{noaug}} = \mathbb{E}_{\mathrm{Q}(\mathbf{w})} \left[\log \mathrm{P}\left(\mathbf{w}\right) + \sum_{i=1}^{N} \log \mathrm{P}_{\text{noaug}}\left(y_i | \mathbf{x}_i, \mathbf{w}\right) - \log \mathrm{Q}\left(\mathbf{w}\right) \right] \tag{23}$$

where $\mathrm{Q}\left(\mathbf{w}\right)$ is the variational approximate posterior learned by optimizing this objective. To understand the effect of the standard approach to DA, we replace $\log \mathrm{P}_{\text{noaug}}\left(y_i | \mathbf{x}_i, \mathbf{w}\right)$ with $\mathcal{L}^i_{\text{loss}}(y_i; \mathbf{w})$. Then, we consider the expected update to the weights, averaging over the augmented images, $\mathbf{x}'_i$ conditioned on the underlying unaugmented images, $\mathbf{x}_i$,

$$\mathbb{E}\left[(\Delta \mathbf{w})_{\text{aug}} \right] = \frac{\epsilon}{2} \nabla_{\mathbf{w}} \left[\log \mathrm{P}\left(\mathbf{w}\right) + \sum_{i=1}^{N} \mathcal{L}^i_{\text{loss}}\left(y_i; \mathbf{w}\right) \right] + \sqrt{\epsilon}\, \boldsymbol{\eta}, \tag{24}$$

$$\text{ELBO}_{\text{aug}} = \mathbb{E}_{\mathrm{Q}(\mathbf{w})} \left[\log \mathrm{P}\left(\mathbf{w}\right) + \sum_{i=1}^{N} \mathcal{L}^i_{\text{loss}}\left(y_i; \mathbf{w}\right) - \log \mathrm{Q}\left(\mathbf{w}\right) \right]. \tag{25}$$

In both cases, this ultimately replaces $\log \mathrm{P}_{\text{noaug}}\left(y_i | \mathbf{x}_i, \mathbf{w}\right)$ with $\mathcal{L}^i_{\text{loss}}\left(y_i; \mathbf{w}\right)$, which as discussed in Sec. 3 is not a valid log-likelihood.

B THE APPROXIMATE POSTERIOR IN VI REDUCES VARIANCE

Here, we derive the ELBO using Jensen's inequality; we take x to be the data and z to be a latent variable. Our goal is to compute the model evidence, $\mathrm{P}\left(x\right)$, by integrating out z,

$$\mathrm{P}\left(x\right) = \int dz\, \mathrm{P}\left(x|z\right) \mathrm{P}\left(z\right) = \int dz\, \mathrm{P}\left(x, z\right) \tag{26}$$

where $\mathrm{P}\left(z\right)$ is the prior, $\mathrm{P}\left(x|z\right)$ is the likelihood and $\mathrm{P}\left(x, z\right)$ is the joint. We introduce an approximate posterior, $\mathrm{Q}\left(z\right)$, and rewrite the integral as an expectation over that approximate posterior and apply Jensen's inequality,

$$\log \mathrm{P}\left(x\right) = \log \int dz\, \mathrm{Q}\left(z\right) \frac{\mathrm{P}\left(x, z\right)}{\mathrm{Q}\left(z\right)} \tag{27}$$

$$= \log \mathbb{E}_{\mathrm{Q}(z)} \left[\frac{\mathrm{P}\left(x, z\right)}{\mathrm{Q}\left(z\right)} \right] \geq \mathbb{E}_{\mathrm{Q}(z)} \left[\log \frac{\mathrm{P}\left(x, z\right)}{\mathrm{Q}\left(z\right)} \right]. \tag{28}$$

Now it is evident that the tightness of the bound is controlled by the variance of $\mathrm{P}\left(x, z\right) / \mathrm{Q}\left(z\right)$. Critically, if $\mathrm{Q}\left(z\right)$ matches the true posterior,

$$\mathrm{Q}\left(z\right) = \mathrm{P}\left(z|x\right) \propto \mathrm{P}\left(x, z\right) \tag{29}$$

then $\mathrm{P}\left(x, z\right) / \mathrm{Q}\left(z\right)$ is constant (zero variance) and the bound is tight.

C KINETIC DIAGNOSTIC RESULTS

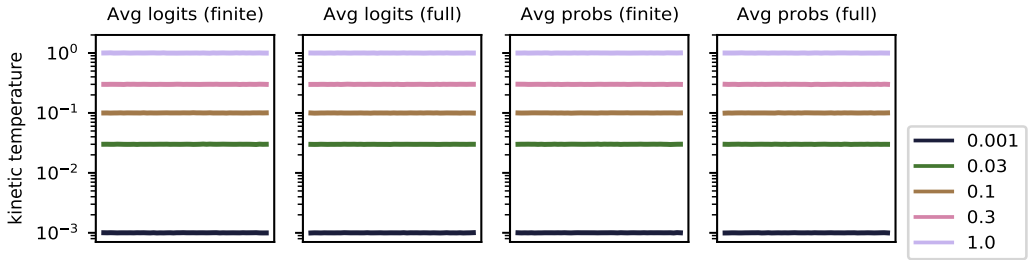


Figure 5: The evolution of the kinetic temperature diagnostic [Leimkuhler and Matthews, 2015] throughout inference. Good agreement between the diagnostic temperature and intended temperature (in legend) suggests accurate inference.

D GENERALIZATION OUTSIDE OF CLASSIFICATION

We may be interested in generalizing the averaging logits and averaging probabilities ideas outside classification. For averaging probabilities, we use,

$$\mathrm{P}\left(y_i|\mathbf{x}_i\right) = \int d\mathbf{x}_i' \, \mathrm{P}\left(y_i|\mathbf{x}_i'\right) \mathrm{P}\left(\mathbf{x}_i'|\mathbf{x}_i\right) = \mathbb{E}\left[\mathrm{P}\left(y_i|\mathbf{x}_i'\right)\right] \tag{30}$$

Intuitively, each augmentation forms one component of a (potentially infinite) mixture model over the outputs, y_i. Importantly, this expression makes no assumption about the support of distributions over y_i, so y_i could be a finite set (classification), real-valued (regression), or anything else (a string, a graph, etc.) Note that directly applying a multi-sample estimator to (the logarithm of) (Eq. 30) gives us a log-likelihood lower bound as in (Eq. 14).

To generalize averaging logits, consider a situation where a distribution over an arbitrary y_i is parameterized by a vector, $\mathbf{f}_i$ output by a neural network,

$$\mathrm{P}\left(y_i|\mathbf{x}\right) = \pi\left(y_i; \mathbf{f}_i\right). \tag{31}$$

In the standard case with no augmentation, we would take $\mathbf{f}_i = \mathbf{f}(\mathbf{x}_i; \mathbf{w})$, (where we take $\mathbf{f}_i$ as the specific vector for input i, and $\mathbf{f}(\cdot;\cdot)$ as a function represented by a neural network, that takes an image and weights and returns a vector). In the case with augmentation, we can average neural network outputs across different augmentations,

$$\mathbf{f}_i = \int d\mathbf{x}_i' \, \mathrm{P}\left(\mathbf{x}_i'|\mathbf{x}_i\right) \mathbf{f}(\mathbf{x}_i'; \mathbf{w}) = \mathbb{E}\left[\mathbf{f}(\mathbf{x}_i'; \mathbf{w})\right]. \tag{32}$$

Note that in this case we need additional conditions for the multi-sample estimator to form a lower bound. In particular, we need $\log \pi\left(y_i; \mathbf{f}_i\right)$ to be concave when treated as a function of $\mathbf{f}_i$ for a fixed y_i.

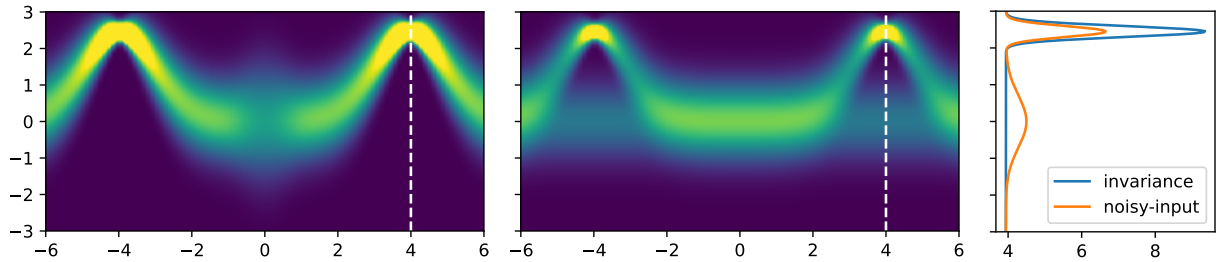
E PERSPECTIVES ON PROBABILISTIC DATA AUGMENTATION

Here we explore in depth the two general approaches to probabilistic data augmentation (Eq. 30) and (Eq. 32). We discuss their justifications in Sec. E.1 and E.2, and compare their properties in Sec. E.3.

E.1 INVARIANCE CONSTRUCTION

In the main text, we suggest two ways of incorporating data augmentation: 1) by averaging logits output by the neural network, and 2) by averaging the predicted probabilities. In classification, both of these methods can be justified by attempting to create a prediction that is more invariant to the transformations in the data augmentation.

When averaging logits, we aim to make the neural network mapping $\mathbf{f} : \mathbb{R}^D \to \mathbb{R}^C$ more invariant by averaging the outputs as in (Eq. 32). This construction influences only the regression function, and so has a similar effect to changing the neural network architecture or changing the prior on the functions $\mathbf{f}(\cdot)$ in the Bayesian case [van der Wilk et al., 2018]. Since only the outputs are affected, this can be directly applied to any likelihood that depends only on an evaluation of the function, i.e. any likelihood which can be written as $\mathrm{P}(y_i|\mathbf{f}_i)$.



(a) Invariant GP (b) Noisy-input GP (c) $p(y|x = 4, \mathcal{D})$

Figure 6: Posterior densities for the model constructions for a single observation at $x_1 = -4, y_1 = 2.5$.

In the case of averaging the probabilities, we can consider the model to be learning a mapping from image inputs to probability vectors $\mathbf{p} : \mathbb{R}^D \to \mathbb{P}^{\mathbb{C}}$. We can make this mapping more invariant in the same way:

$$\begin{aligned} \mathbf{p}_{\text{inv}}(\mathbf{x}_i; \mathbf{w}) &= \mathbb{E}\left[\text{softmax}\left(\mathbf{f}(\mathbf{x}'_i; \mathbf{w})\right)\right] \\ &= \int \text{softmax}\left(\mathbf{f}(\mathbf{x}'_i; \mathbf{w})\right) \mathrm{P}(\mathbf{x}'_i|\mathbf{x}_i) \mathrm{d}\mathbf{x}'_i \,. \end{aligned} \tag{33}$$

The straightforward generalization of this construction would be to replace the softmax with the appropriate likelihood (see general case in (Eq. 30)). When considering likelihoods other than softmax classification (e.g. Gaussian likelihoods for regression), stronger differences between these constructions emerge in both behaviour and justification. We investigate further in Appendix E.3.

E.2 NOISY-INPUT MODEL

As stated above, we can generalize averaging the classification probabilities by replacing the softmax with the appropriate likelihood as in (Eq. 30). This modified likelihood, which incorporates data augmentation, was also discussed in Wenzel et al. [2020, Appendix K] and is a (potentially continuous) mixture model on the observation y_n, where each augmentation introduces a mixture component. This is as a *noisy-input* model [Girard and Murray-Smith, 2003, McHutchon and Rasmussen, 2011, Damianou et al., 2016] where the input $\mathbf{x}_i$ is corrupted via the augmentation distribution.

E.3 MODEL COMPARISON

The forms of the invariance construction (Eq. 32) and the noisy-input model (Eq. 30) imply a difference of purpose. In using the invariance construction, we seek a regression function with the specified symmetry, which is consistent with the data according to the likelihood function $\mathrm{P}\left(y_i|\mathbf{f}_i\right)$. Conversely, with the noisy-input model (Eq. 30) we aim to find a function which gives rise to an invariant likelihood, consistent with the observed outputs for inputs randomly perturbed by $\mathrm{P}(\mathbf{x}'|\mathbf{x})$. The role of $\mathbf{x}'$ is different in each case. In the noisy-input model, $\mathbf{x}'$ is a latent variable on which we could, in principle, do inference (with e.g. an amortized variational approach). While in the invariance construction, we integrate over $\mathbf{x}'$ to parameterize $\mathbf{f}(\mathbf{x}; \mathbf{w})$.

We now compare the behaviours of the invariance and noisy-input constructions. We will see that they result in quite different posteriors.

In the main text, we compared the empirical performance of averaging probabilities and averaging logits for BNN classification (see Figs. 2 and 4). However, as the invariance perspective justifies both averaging logits and probabilities, this comparison does not clearly distinguish between the noisy-input and invariance viewpoints. Further, we are interested not only in predictive performance but also in understanding how each construction behaves. With this in mind, we investigate the models with an illustrative example, where we can both integrate over the orbit and do inference in closed form.

We consider Gaussian process (GP) regression with a one-dimensional input and data augmentation which enforces symmetry about $x = 0$, i.e. $\mathrm{P}(x'|x) = \frac{1}{2}\left(\delta(x' - x) + \delta(x' + x)\right)$. From Van der Wilk et al. [2018], the invariance view

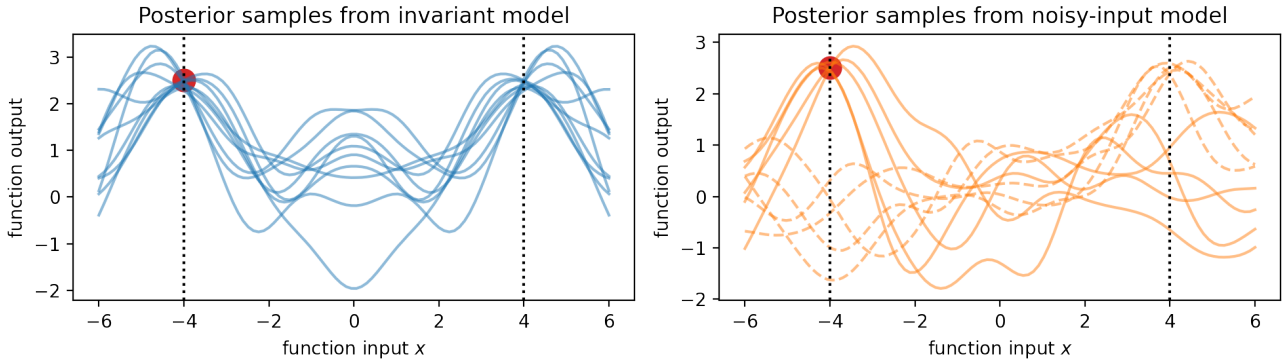


Figure 7: Samples from the model posteriors. The red dot marks the location of the observation $(x_1 = -4, y_1 = 2.5)$. The noisy input posterior comprises two components: one conditioned on (x_1, y_1) (dashed lines), the other on $(-x_1, y_1)$ (solid lines).

can be expressed in the kernel of the GP:

$$g \sim \mathcal{GP}(\mathbf{0}, k_{\text{base}}) \tag{34}$$

$$f(x) = g(x) + g(-x) \tag{35}$$

$$\implies f \sim \mathcal{GP}(\mathbf{0}, k_{\text{inv}}), \tag{36}$$

$$\text{where} \quad k_{\text{inv}}(x_i, x_j) = \sum_{c_i \in \{-1,1\}} \sum_{c_j \in \{-1,1\}} k_{\text{base}}(c_i x_i, c_j x_j). \tag{37}$$

We then follow standard GP inference to find the posterior over invariant functions. Note that unlike Van der Wilk et al. [2018], we are not concerned with learning invariances here.

The noisy-input model for this case is

$$\mathrm{P}(\mathbf{x}, \mathbf{y}, \mathbf{f}) = \mathrm{P}(\mathbf{f}) \prod_{i=1}^{N} \int \mathrm{P}(y_i | f(x_i')) \, \mathrm{P}(x_i' | x_i) dx_i' \tag{38}$$

$$\mathrm{P}(y_i | f(x_i')) = \mathcal{N}\left(y_i; f(x_i'), \sigma^2\right) \tag{39}$$

$$f \sim \mathcal{GP}(0, k). \tag{40}$$

Given a single observation (x_1, y_1), the noisy-input posterior is

$$\mathrm{P}(f | x_1, y_1) = \frac{1}{Z} \mathrm{P}(f(x_1), x_1, y_1) \tag{41}$$

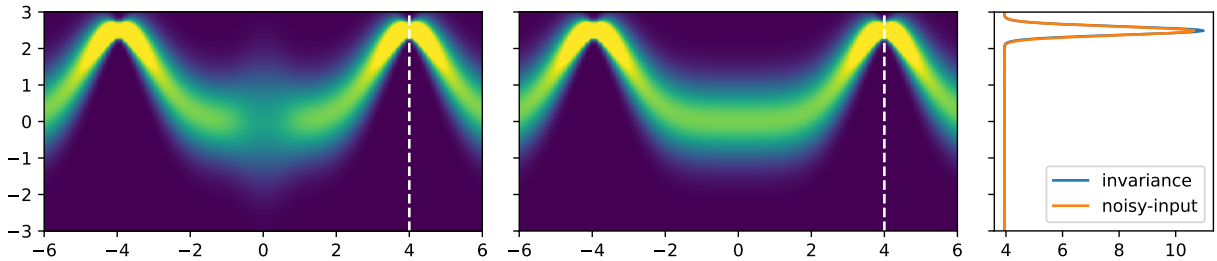
$$= \frac{1}{2Z} \mathrm{P}\left(f(x_1)\right) \left[\mathrm{P}(y_1 | f(x_1)) + \mathrm{P}(y_1 | f(-x_1))\right] \tag{42}$$

$$= \frac{1}{2} \left[\mathrm{P}(f | x_1, y_1) + \mathrm{P}(f | -x_1, y_1)\right], \tag{43}$$

a mixture of GP posteriors, with two components (one for each point in the orbit).

How do these posteriors compare? For an observation at $(x_1 = -4, y_1 = 2.5)$ we plot the posterior predictive densities in Fig. 6. Both posteriors are symmetric around $x = 0$ as we expect, however the noisy-input model is bimodal in the regions surrounding $x = 4$ and $x = -4$, where the invariance posterior has unimodal density concentrated around the observed y value of 2.5. The difference is clear in Fig. 6c, which shows the marginal predictive densities at $x = 4$.

In the noisy-input case, our observation is (x_1, y_1), but x is uncertain, so the observation could have been generated by $(-x_1, y_1)$ with equal probability. This results in a mixture posterior with two components: one component has "seen" (x_1, y_1), while the other "saw" $(-x_1, y_1)$. The first component's prediction at $-x_1$ remains uninformed by its "observation" and the same is true for the second component's prediction at x_1. Thus, the predictions made by these components at these locations revert to the zero-mean prior.



(a) Invariant GP (b) Noisy-input GP (c) $p(y|x = 4, \mathcal{D})$

Figure 8: Posterior densities for the model constructions for ten observations at $\{(x_i = -4, y_i = 2.5)\}_{i=1}^{10}$.

From the invariance perspective, we condition on the point (x_1, y_1) but the double-sum kernel forces the function to be the same at $(-x_1, y_1)$. As the posterior is a single GP, it has unimodal marginals with high density around both points.

We can gain further intuition by looking at samples from both posteriors (Fig. 7). We can see that the *every* sample from the invariance posterior is symmetric about $x = 0$, where the functions drawn from the noisy input posterior are not symmetric in general.

The samples illustrate the key difference between the models. For the noisy input model, we can see the two components of the mixture posterior arise from conditioning on different locations in the orbit of x_1 as described above. The component going through $(x = 4, y_1)$ (samples drawn with dashed lines) is close to the prior at $(x = -4, y_1)$, the other (solid lines) goes through $(x = -4, y_1)$ and is close to the prior at $(x = 4, y_1)$. However, under the invariance model, inference on the observation concentrates all model density around y_1 for both points in the orbit of x_1.

We now consider how this comparison changes as we observe more data. The noisy-input model (Eq. 38) requires integration over $\mathrm{P}(x'|x)$ to compute the likelihood of each datapoint, all of which are multiplied together to calculate their combined likelihood. Thus, the number of posterior components grows exponentially with the number of observations: A^N (for orbit size A). Suppose all observations are at the same location (x_1, y_1). In this case, the posterior density due to prior reversion at $\{x_1, -x_1\}$ decreases exponentially with N. This is because the fraction of mixture components conditioned on all input observations being at the same point in the orbit of x_1, i.e. all at x_1 or $-x_1$, is given by A^{1-N}. The predictive posteriors for ten observations, each at $(x = -4, y = 2.5)$ is shown in Fig. 8. Contrasting this noisy-input posterior (Fig. 8b) to that for one observation (Fig. 6b), we can see the reduction in density around to the prior mean for points around the orbit of x_1. In summary, the noisy-input and invariance posteriors become more alike as we observe more data in the same orbit.