Bayesian Deep Learning and a Probabilistic Perspective of Generalization

Andrew Gordon Wilson **Pavel Izmailov**
New York University

Abstract

The key distinguishing property of a Bayesian approach is marginalization, rather than using a single setting of weights. Bayesian marginalization can particularly improve the accuracy and calibration of modern deep neural networks, which are typically underspecified by the data, and can represent many compelling but different solutions. We show that deep ensembles provide an effective mechanism for approximate Bayesian marginalization, and propose a related approach that further improves the predictive distribution by marginalizing within basins of attraction, without significant overhead. We also investigate the prior over functions implied by a vague distribution over neural network weights, explaining the generalization properties of such models from a probabilistic perspective. From this perspective, we explain results that have been presented as mysterious and distinct to neural network generalization, such as the ability to fit images with random labels, and show that these results can be reproduced with Gaussian processes. Finally, we provide a Bayesian perspective on tempering for calibrating predictive distributions.

1. Introduction

Imagine fitting the airline passenger data in Figure 1. Which model would you choose: (1) $f_1(x) = a_0 + a_1 x$, (2) $\sum_{j=0}^{3} a_j x^j$, or (3) $f_3(x) = \sum_{j=0}^{10^4} a_j x^j$?

Put this way, most audiences overwhelmingly favour choices (1) and (2), for fear of overfitting. But of these options, choice (3) most honestly represents our beliefs. Indeed, it is likely that the ground truth explanation for the data is out of class for any of these choices, but there is some setting of the coefficients $\{a_j\}$ in choice (3) which provides a better description of reality than could be managed by choices (1) and (2), which are special cases of choice (3). Moreover, our beliefs about the generative processes for our observations, which are often very sophisticated, typically ought to be independent of how many data points we happen to observe.

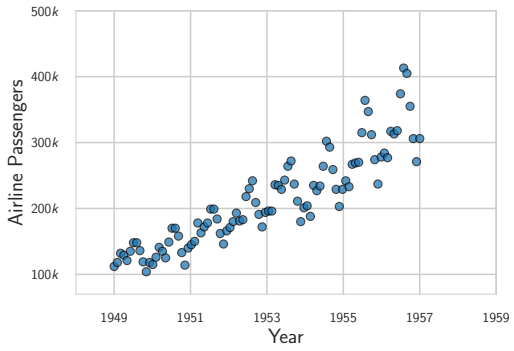


Figure 1. Airline passenger numbers recorded monthly.

And in modern practice, we are implicitly favouring choice (3): we often use neural networks with millions of parameters to fit datasets with thousands of points. Furthermore, non-parametric methods such as Gaussian processes often involve infinitely many parameters, enabling the flexibility for universal approximation (Rasmussen & Williams, 2006), yet in many cases provide very simple predictive distributions. Indeed, parameter counting is a poor proxy for understanding generalization behaviour.

From a probabilistic perspective, we argue that generalization depends largely on *two* properties, the *support* and the *inductive biases* of a model. Consider Figure 2(a), where on the horizontal axis we have a conceptualization of all possible datasets, and on the vertical axis the Bayesian *evidence* for a model. The evidence, or marginal likelihood, $p(\mathcal{D}|\mathcal{M}) = \int p(\mathcal{D}|\mathcal{M}, w)p(w)dw$, is the probability we would generate a dataset if we were to randomly sample from the prior over functions $p(f(x))$ induced by a prior over parameters $p(w)$. We define the support as the range of datasets for which $p(\mathcal{D}|\mathcal{M}) > 0$. We define the inductive biases as the relative prior probabilities of different datasets — the *distribution of support* given by $p(\mathcal{D}|\mathcal{M})$. A similar schematic to Figure 2(a) was used by MacKay (1992) to understand an Occam's razor effect in using the evidence for model selection; we believe it can also be used to reason about model construction and generalization.

From this perspective, we want the support of the model to be large so that we can represent any hypothesis we believe to be possible, even if it is unlikely. We would even want

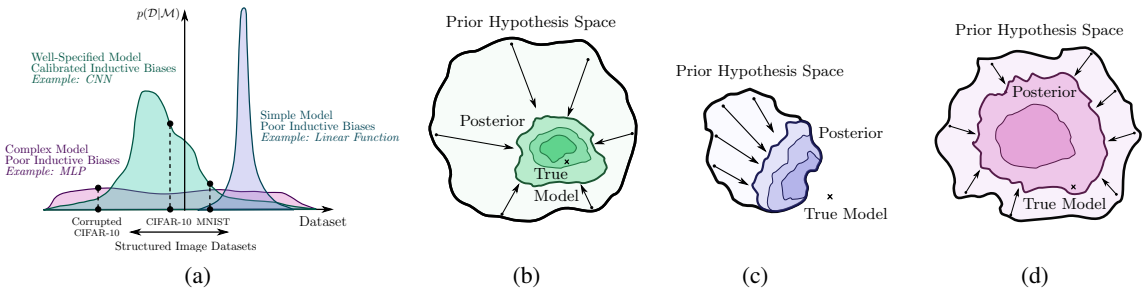


Figure 2. **A probabilistic perspective of generalization.** (a) Ideally, a model supports a wide range of datasets, but with inductive biases that provide high prior probability to a particular class of problems being considered. Here, the CNN is preferred over the linear model and the fully-connected MLP for CIFAR-10 (while we do not consider MLP models to in general have poor inductive biases, here we are considering a hypothetical example involving images and a very large MLP). (b) By representing a large hypothesis space, a model can contract around a true solution, which in the real-world is often very sophisticated. (c) With truncated support, a model will converge to an erroneous solution. (d) Even if the hypothesis space contains the truth, a model will not efficiently contract unless it also has reasonable inductive biases.

the model to be able to represent pure noise, such as noisy CIFAR (Zhang et al., 2016), as long as we honestly believe there is some non-zero, but potentially arbitrarily small, probability that the data are simply noise. Crucially, we also need the inductive biases to carefully represent which hypotheses we believe to be a priori likely for a particular problem class. If we are modelling images, then our model should have statistical properties, such as convolutional structure, which are good descriptions of images.

Figure 2(a) illustrates three models. We can imagine the blue curve as a simple linear function, $f(x) = a_0 + a_1 x$, combined with a distribution over parameters $p(a_0, a_1)$, e.g., $\mathcal{N}(0, I)$, which induces a distribution over functions $p(f(x))$. Parameters we sample from our prior $p(a_0, a_1)$ give rise to functions $f(x)$ that correspond to straight lines with different slopes and intercepts. This model thus has truncated support: it cannot even represent a quadratic function. But because the marginal likelihood must normalize over datasets $\mathcal{D}$, this model assigns much mass to the datasets it does support. The red curve could represent a large fully-connected MLP. This model is highly flexible, but distributes its support across datasets too evenly to be particularly compelling for many image datasets. The green curve could represent a convolutional neural network, which represents a compelling specification of support and inductive biases for image recognition: this model has the flexibility to represent many solutions, but its structural properties provide particularly good support for many image problems.

With large support, we cast a wide enough net that the posterior can contract around the true solution to a given problem as in Figure 2(b), which in reality we often believe to be very sophisticated. On the other hand, the simple model will have a posterior that contracts around an erroneous solution if it is not contained in the hypothesis space as in Figure 2(c). Moreover, in Figure 2(d), the model has wide support, but does not contract around a good solution because its support is too evenly distributed.

Returning to the opening example, we can justify the high order polynomial by wanting large support. But we would still have to carefully choose the prior on the coefficients to induce a distribution over functions that would have reasonable inductive biases. Indeed, this Bayesian notion of generalization is not based on a single number, but is a two dimensional concept. From this probabilistic perspective, it is crucial not to conflate the *flexibility* of a model with the *complexity* of a model class. Indeed Gaussian processes with RBF kernels have large support, and are thus flexible, but have inductive biases towards very simple solutions. We also see that *parameter counting* has no significance in this perspective of generalization: what matters is how a distribution over parameters combines with a functional form of a model, to induce a distribution over solutions. Rademacher complexity (Mohri & Rostamizadeh, 2009), VC dimension (Vapnik, 1998), and many conventional metrics, are by contrast *one dimensional notions*, corresponding roughly to the support of the model, which is why they have been found to provide an incomplete picture of generalization in deep learning (Zhang et al., 2016).

In this paper we reason about Bayesian deep learning from a probabilistic perspective of generalization. The key distinguishing property of a Bayesian approach is marginalization instead of optimization, where we represent solutions given by all settings of parameters weighted by their posterior probabilities, rather than bet everything on a single setting of parameters. Neural networks are typically underspecified by the data, and can represent many different but high performing models corresponding to different settings of parameters, which is exactly when marginalization will

make the biggest difference for accuracy and calibration. Moreover, we clarify that the recent deep ensembles (Lakshminarayanan et al., 2017) are not a competing approach to Bayesian inference, but can be viewed as a compelling mechanism for Bayesian marginalization. Indeed, we empirically demonstrate that deep ensembles can provide a *better* approximation to the Bayesian predictive distribution than standard Bayesian approaches. We further propose a new method, inspired by deep ensembles, which marginalizes within basins of attraction — achieving significantly improved performance, with a similar training time.

We then investigate the properties of priors over functions induced by priors over the weights of neural networks, showing that they have reasonable inductive biases. We also show that the mysterious generalization properties recently presented in Zhang et al. (2016) can be understood by reasoning about prior distributions over functions, and are not specific to neural networks. Indeed, we show Gaussian processes can also perfectly fit images with random labels, yet generalize on the noise-free problem. These results are a consequence of large support but reasonable inductive biases for common problem settings. We further show that while Bayesian neural networks can fit the noisy datasets, the marginal likelihood has much better support for the noise free datasets, in line with Figure 2.

In the Appendix we provide several additional experiments and results, including a discussion of tempering in Bayesian deep learning. We also provide code at `https://github.com/izmailovpavel/understandingbdl`.

2. Related Work

Notable early works on Bayesian neural networks include MacKay (1992), MacKay (1995), and Neal (1996). These works generally argue in favour of making the model class for Bayesian approaches as flexible as possible, in line with Box & Tiao (1973). Accordingly, Neal (1996) pursued the limits of large Bayesian neural networks, showing that as the number of hidden units approached infinity, these models becomes a Gaussian processes with a particular kernel functions. This work harmonizes with recent work describing the neural tangent kernel (e.g., Jacot et al., 2018).

The marginal likelihood is often used for Bayesian hypothesis testing, model comparison, and hyperparameter tuning, with *Bayes factors* used to select between models (Kass & Raftery, 1995). MacKay (2003, Ch. 28) uses a diagram similar to Fig 2(a) to show the marginal likelihood has an *Occam's razor* property, favouring the simplest model consistent with a given dataset, even if the prior assigns equal probability to the various models. Rasmussen & Ghahramani (2001) reasons about how the marginal likelihood can favour large flexible models, as long as such models correspond to a reasonable distribution over functions.

There has been much recent interest in developing Bayesian approaches for modern deep learning, with new challenges and architectures quite different from what had been considered in early work. Recent work has largely focused on scalable inference (e.g., Gal & Ghahramani, 2016; Kendall & Gal, 2017; Ritter et al., 2018; Khan et al., 2018; Maddox et al., 2019), function-space inspired priors (e.g., Sun et al., 2019; Yang et al., 2019; Louizos et al., 2019; Hafner et al., 2018), and developing flat objective priors in parameter space, directly leveraging the biases of the neural network functional form (e.g, Nalisnick, 2018). Wilson (2020) provides a note motivating Bayesian deep learning.

Additionally, Pearce et al. (2018) propose a modification of deep ensembles and argue that it performs approximate Bayesian inference, and Gustafsson et al. (2019) briefly mention how deep ensembles can be viewed as samples from an approximate posterior. In the context of deep ensembles, we believe it is natural to consider the BMA integral separately from the simple Monte Carlo approximation that is often used to approximate this integral; to compute an accurate predictive distribution, we do not need samples from a posterior, or even a faithful approximation to the posterior.

Fort et al. (2019) considered the diversity of predictions produced by models from a single SGD run, and models from independent SGD runs, and suggested to ensemble averages of SGD iterates. Although MultiSWA (one of the methods considered in Section 4) is related to this idea, the crucial practical difference is that MultiSWA uses a learning rate schedule that selects for flat regions of the loss, the key to the success of the SWA method (Izmailov et al., 2018).

3. Bayesian Marginalization

Often the predictive distribution we want to compute is given by

$$p(y|x, \mathcal{D}) = \int p(y|x, w)p(w|\mathcal{D})dw\,. \tag{1}$$

The outputs are y (e.g., regression values, class labels, ...), indexed by inputs x (e.g. spatial locations, images, ...), the weights (or parameters) of the neural network $f(x; w)$ are w, and $\mathcal{D}$ are the data. Eq. (1) represents a *Bayesian model average* (BMA). Rather than bet everything on one hypothesis — with a single setting of parameters w — we want to use all settings of parameters, weighted by their posterior probabilities. This procedure is called *marginalization* of the parameters w, as the predictive distribution of interest no longer conditions on w. This is not a controversial equation, but simply the sum and product rules of probability.

3.1. Importance of Marginalization in Deep Learning

In general, we can view classical training as performing approximate Bayesian inference, using the approximate posterior $p(w|\mathcal{D}) \approx \delta(w = \hat{w})$ to compute Eq. (1), where δ is a Dirac delta function that is zero everywhere except at $\hat{w} = \text{argmax}_w p(w|\mathcal{D})$. In this case, we recover the standard predictive distribution $p(y|x, \hat{w})$. From this perspective, many alternatives, albeit imperfect, will be preferable — including impoverished Gaussian posterior approximations for $p(w|\mathcal{D})$, even if the posterior or likelihood are actually highly non-Gaussian and multimodal.

The difference between a classical and Bayesian approach will depend on how sharp the posterior $p(w|\mathcal{D})$ becomes. If the posterior is sharply peaked, and the conditional predictive distribution $p(y|x, w)$ does not vary significantly where the posterior has mass, there may be almost no difference, since a delta function may then be a reasonable approximation of the posterior for the purpose of BMA. However, modern neural networks are usually highly underspecified by the available data, and therefore have diffuse likelihoods $p(\mathcal{D}|w)$. Not only are the likelihoods diffuse, but different settings of the parameters correspond to a diverse variety of compelling hypotheses for the data (Garipov et al., 2018; Izmailov et al., 2019). This is exactly the setting when we *most* want to perform a Bayesian model average, which will lead to an ensemble containing many different but high performing models, for better calibration *and* accuracy than classical training.

Loss Valleys. Flat regions of low loss (negative log posterior density $-\log p(w|\mathcal{D})$) are associated with good generalization (Izmailov et al., 2018; Hochreiter & Schmidhuber, 1997; Keskar et al., 2016). While flat solutions that generalize poorly can be contrived through reparametrization (Dinh et al., 2017), the flat regions that lead to good generalization contain a *diversity* of high performing models on test (Izmailov et al., 2018), corresponding to different parameter settings in those regions. And indeed, there are large contiguous regions of low loss that contain such solutions, even connecting together different SGD solutions (Garipov et al., 2018; Izmailov et al., 2019) (see also Figure 8, Appendix).

Since these regions of the loss represent a large volume in a high-dimensional space (Huang et al., 2019), and provide a diversity of solutions, they will dominate in forming the predictive distribution in a Bayesian model average. By contrast, if the parameters in these regions provided similar functions, as would be the case in flatness obtained through reparametrization, these functions would be redundant in the model average. That is, although the solutions of high posterior density can provide poor generalization, it is the solutions that generalize well that will have greatest posterior *mass*, and thus be automatically favoured by the BMA.

Calibration by Epistemic Uncertainty Representation. It has been noticed that modern neural networks are often *miscalibrated* in the sense that their predictions are typically *overconfident* (Guo et al., 2017). For example, in classification the highest softmax output of a convolutional neural network is typically much larger than the probability of the associated class label. The fundamental reason for miscalibration is ignoring epistemic uncertainty. A neural network can represent many models that are consistent with our observations. By selecting only one, in a classical procedure, we lose uncertainty when the models disagree for a test point. In regression, we can visualize epistemic uncertainty by looking at the spread of the predictive distribution; as we move away from the data, there are a greater variety of consistent solutions, leading to larger uncertainty, as in Figure 4. We can further calibrate the model with tempering, which we discuss in the Appendix Section F.

Accuracy. An often overlooked benefit of Bayesian model averaging in *modern* deep learning is improved *accuracy*. If we average the predictions of many high performing models that disagree in some cases, we should see significantly improved accuracy. This benefit is now starting to be observed in practice (e.g., Izmailov et al., 2019). Improvements in accuracy are very convincingly exemplified by *deep ensembles* (Lakshminarayanan et al., 2017), which have been perceived as a competing approach to Bayesian methods, but in fact provides a compelling mechanism for approximate Bayesian model averaging, as we show in Section 3.3.

3.2. Beyond Monte Carlo

Nearly all approaches to estimating the integral in Eq. (1), when it cannot be computed in closed form, involve a *simple Monte Carlo* approximation: $p(y|x, \mathcal{D}) \approx \frac{1}{J}\sum_{j=1}^{J} p(y|x, w_j)\,, w_j \sim p(w|\mathcal{D})$. In practice, the samples from the posterior $p(w|\mathcal{D})$ are also approximate, and found through MCMC or deterministic methods. The deterministic methods approximate $p(w|\mathcal{D})$ with a different more convenient density $q(w|\mathcal{D}, \theta)$ from which we can sample, often chosen to be Gaussian. The parameters θ are selected to make q close to p in some sense; for example, variational approximations (e.g., Beal, 2003), which have emerged as a popular deterministic approach, find $\text{argmin}_\theta \mathcal{KL}(q||p)$. Other standard deterministic approximations include Laplace (e.g., MacKay, 1995), EP (Minka, 2001a), and INLA (Rue et al., 2009).

From the perspective of estimating the predictive distribution in Eq. (1), we can view simple Monte Carlo as approximating the posterior with a set of point masses, with locations given by samples from another approximate posterior q, even if q is a continuous distribution. That is, $p(w|\mathcal{D}) \approx \sum_{j=1}^{J} \delta(w = w_j)\,, w_j \sim q(w|\mathcal{D})$.

Ultimately, the goal is to accurately compute the predictive distribution in Eq. (1), rather than find a generally accurate representation of the posterior. In particular, we must carefully represent the posterior in regions that will make the greatest contributions to the BMA integral. In terms of efficiently computing the predictive distribution, we do not necessarily want to place point masses at locations given by samples from the posterior. For example, functional diversity is important for a good approximation to the BMA integral, because we are summing together terms of the form $p(y|x, w)$; if two settings of the weights w_i and w_j each provide high likelihood (and consequently high posterior density), but give rise to similar functions $f(x; w_i)$, $f(x; w_j)$, then they will be largely redundant in the model average, and the second setting of parameters will not contribute much to the estimating the BMA integral for the unconditional predictive distribution. In Sections 3.3 and 4, we consider how various approaches approximate the predictive distribution.

3.3. Deep Ensembles are BMA

Deep ensembles (Lakshminarayanan et al., 2017) is fast becoming a gold standard for accurate and well-calibrated predictive distributions. A recent report (Ovadia et al., 2019) shows that deep ensembles appear to outperform some particular approaches to Bayesian neural networks, leading to the confusion that deep ensembles and Bayesian methods are competing approaches. To the contrary, deep ensembles are actually a compelling approach to Bayesian model averaging, in the vein of Section 3.2.

There is a fundamental difference between a Bayesian model average and some approaches to ensembling. The Bayesian model average assumes that *one* hypothesis (one parameter setting) is correct, and averages over models due to an inability to distinguish between hypotheses given limited information (Minka, 2000). As we observe more data, the posterior collapses onto a single hypotheses. If the true explanation for the data is a combination of hypotheses, then the Bayesian model average may appear to perform worse as we observe more data. Some ensembling methods work by enriching the hypothesis space, and therefore do not collapse in this way. Deep ensembles, however, are formed by MAP or maximum likelihood retraining of the same architecture multiple times, leading to different basins of attraction. The deep ensemble will therefore collapse in the same way as a Bayesian model average, as the posterior concentrates. Since the hypotheses space (support) for a modern neural network is large, containing many different possible explanations for the data, posterior collapse will often be desirable.

Furthermore, by representing multiple basins of attraction, deep ensembles can provide a *better* approximation to the BMA than the Bayesian approaches in Ovadia et al. (2019). Indeed, the functional diversity is important for a good approximation to the BMA integral, as per Section 3.2. The approaches referred to as Bayesian in Ovadia et al. (2019) instead focus their approximation on a single basin, which may contain a lot of redundancy in function space, making a relatively minimal contribution to computing the Bayesian predictive distribution. On the other hand, retraining a neural network multiple times for deep ensembles incurs a significant computational expense. The single basin approaches may be preferred if we are to control for computation. We explore these questions in Section 4.

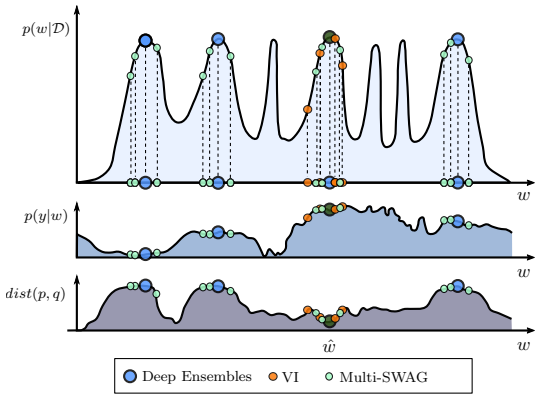
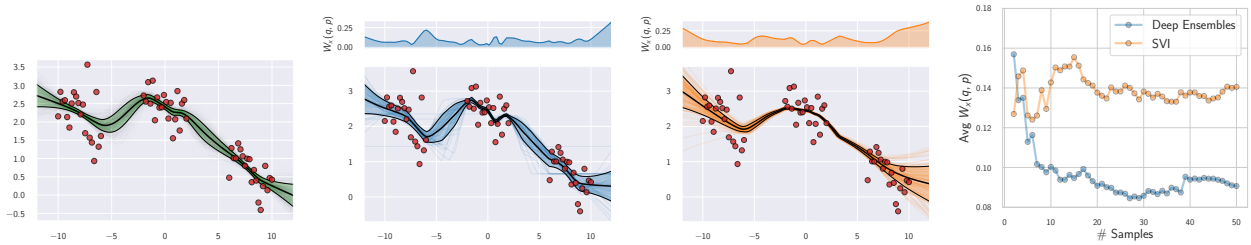


Figure 3. **Approximating the BMA.** $p(y|x, \mathcal{D}) = \int p(y|x, w)p(w|\mathcal{D})dw$. **Top:** $p(w|\mathcal{D})$, with representations from VI (orange) deep ensembles (blue), MultiSWAG (red). **Middle:** $p(y|x, w)$ as a function of w for a test input x. This function does not vary much within modes, but changes significantly between modes. **Bottom:** Distance between the true predictive distribution and the approximation, as a function of representing a posterior at an additional point w, assuming we have sampled the mode in dark green. There is more to be gained by exploring new basins, than continuing to explore the same basin.

4. An Empirical Study of Marginalization

We have shown that deep ensembles can be interpreted as an approximate approach to Bayesian marginalization, which selects for functional diversity by representing multiple basins of attraction in the posterior. Most Bayesian deep learning methods instead focus on faithfully approximating a posterior within a single basin of attraction. We propose a new method, MultiSWAG, which combines these two types of approaches. MultiSWAG combines multiple independently trained SWAG approximations (Maddox et al., 2019), to create a mixture of Gaussians approximation to the posterior, with each Gaussian centred on a different basin of attraction. We note that MultiSWAG does not require *any* additional training time over standard deep ensembles.



(a) Exact (b) Deep Ensembles (c) Variational Inference (d) Discrepancy with True BMA

Figure 4. **Approximating the true predictive distribution. (a)**: A close approximation of the true predictive distribution obtained by combining 200 HMC chains. **(b)**: Deep ensembles predictive distribution using 50 independently trained networks. **(c)**: Predictive distribution for factorized variational inference (VI). **(d)**: Convergence of the predictive distributions for deep ensembles and variational inference as a function of the number of samples; we measure the average Wasserstein distance between the marginals in the range of input positions. The multi-basin deep ensembles approach provides a more faithful approximation of the Bayesian predictive distribution than the conventional single-basin VI approach, which is overconfident between data clusters. The top panels show the Wasserstein distance between the true predictive distribution and the deep ensemble and VI approximations, as a function of inputs x.

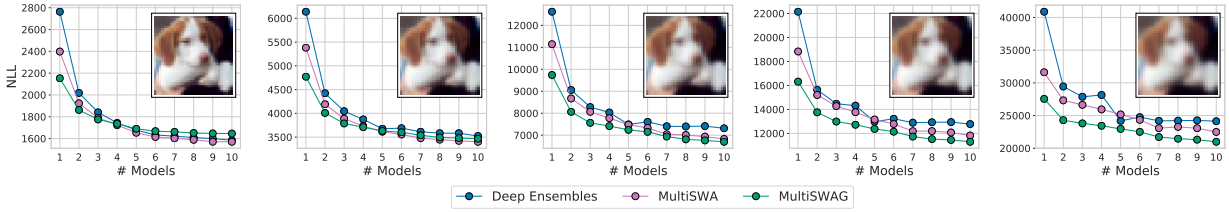
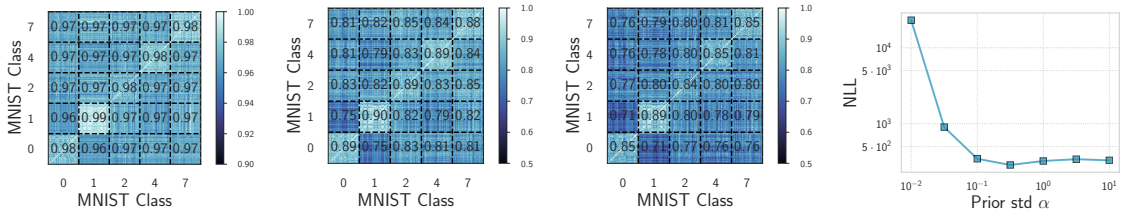


Figure 5. Negative log likelihood for Deep Ensembles, MultiSWAG and MultiSWA using a PreResNet-20 on CIFAR-10 with varying intensity of the *Gaussian blur* corruption. The image in each plot shows the intensity of corruption. For all levels of intensity, MultiSWAG and MultiSWA outperform Deep Ensembles for a small number of independent models. For high levels of corruption MultiSWAG significantly outperforms other methods even for many independent models. We present results for other corruptions in the Appendix.

We illustrate the conceptual difference between deep ensembles, a standard variational single basin approach, and MultiSWAG, in Figure 3. In the top panel, we have a conceptualization of a multimodal posterior. VI approximates the posterior with multiple samples within a single basin. But we see in the middle panel that the conditional predictive distribution $p(y|x, w)$ does not vary significantly within the basin, and thus each additional sample contributes minimally to computing the marginal predictive distribution $p(y|x, \mathcal{D})$. On the other hand, $p(y|x, w)$ varies significantly between basins, and thus each point mass for deep ensembles contributes significantly to the marginal predictive distribution. By sampling within the basins, MultiSWAG provides additional contributions to the predictive distribution. In the bottom panel, we have the gain in approximating the predictive distribution when adding a point mass to the representation of the posterior, as a function of its location, assuming we have already sampled the mode in dark green. Including samples from different modes provides significant gain over continuing to sample from the same mode, and including weights in wide basins provide relatively more gain than the narrow ones.

In Figure 4 we evaluate single basin and multi-basin approaches in a case where we can near-exactly compute the predictive distribution. We provide details for generating the data and training the models in Appendix D.1. We see that the predictive distribution given by deep ensembles is qualitatively closer to the true distribution, compared to the single basin variational method: between data clusters, the deep ensemble approach provides a similar representation of epistemic uncertainty, whereas the variational method is extremely overconfident in these regions. Moreover, we see that the Wasserstein distance between the true predictive distribution and these two approximations quickly shrinks with number of samples for deep ensembles, but is roughly independent of number of samples for the variational approach. Thus the deep ensemble is providing a better approximation of the Bayesian model average in Eq. (1) than the single basin variational approach, which has traditionally been labelled as the Bayesian alternative.

Next, we evaluate MultiSWAG under distribution shift on the CIFAR-10 dataset (Krizhevsky et al., 2014), replicating the setup in Ovadia et al. (2019). We consider 16 data corruptions, each at 5 different levels of severity, introduced by Hendrycks & Dietterich (2019). For each corruption, we evaluate the performance of deep ensembles and Mul-



(a) $\alpha = 0.02$ (b) $\alpha = 0.1$ (c) $\alpha = 1$ (d)

Figure 6. **Induced Prior Correlation Function.** Average pairwise prior correlations for pairs of objects in classes $\{0, 1, 2, 4, 7\}$ of MNIST induced by LeNet-5 for $p(f(x; w))$ when $p(w) = \mathcal{N}(0, \alpha^2 I)$. Images in the same class have higher prior correlations than images from different classes, suggesting that $p(f(x; w))$ has desirable inductive biases. The correlations slightly decrease with increases in α. **(d)**: NLL of an ensemble of 20 SWAG samples on MNIST as a function of α using a LeNet-5.

tiSWAG varying the training budget. For deep ensembles we show performance as a function of the number of independently trained models in the ensemble. For MultiSWAG we show performance as a function of the number of independent SWAG approximations that we construct; we then sample 20 models from each of these approximations to construct the final ensemble.

While the training time for MultiSWAG is the same as for deep ensembles, at test time MultiSWAG is more expensive, as the corresponding ensemble consists of a larger number of models. To account for situations when inference time is constrained, we also propose MultiSWA, a method that ensembles independently trained SWA solutions (Izmailov et al., 2018). SWA solutions are the means of the corresponding Gaussian SWAG approximations. Izmailov et al. (2018) argue that SWA solutions approximate the local ensembles represented by SWAG with a single model.

In Figure 5 we show the negative log-likelihood as a function of the number of independently trained models for a Preactivation ResNet-20 on CIFAR-10 corrupted with Gaussian blur with varying levels of intensity (increasing from left to right) in Figure 5. MultiSWAG outperforms deep ensembles significantly on highly corrupted data. For lower levels of corruption, MultiSWAG works particularly well when only a small number of independently trained models are available. We note that MultiSWA also outperforms deep ensembles, and has the same computational requirements at training and test time as deep ensembles. We present results for other types of corruption in Appendix Figures 12, 13, 14, 15, showing similar trends.

5. Neural Network Priors

A prior over parameters $p(w)$ combines with the functional form of a model $f(x; w)$ to induce a distribution over functions $p(f(x; w))$. It is this distribution over functions that controls the generalization properties of the model; the prior over parameters, in isolation, has no meaning. Neural networks are imbued with structural properties that provide good inductive biases, such as translation equivariance, hierarchical representations, and sparsity. In the sense of Figure 2, the prior will have large support, due to the flexibility of neural networks, but its inductive biases provide the most mass to datasets which are representative of problem settings where neural networks are often applied. In this section, we study the properties of the induced distribution over functions. We directly continue the discussion of priors in Section 6, with a focus on examining the noisy CIFAR results in Zhang et al. (2016), from a probabilistic perspective of generalization. These sections are best read together.

We also provide several additional experiments in the Appendix. In Section E, we present analytic results on the dependence of the prior distribution in function space on the variance of the prior over parameters, considering also layer-wise parameter priors with ReLU activations. As part of a discussion on tempering, in Section F.4 we study the effect of α in $p(w) = \mathcal{N}(0, \alpha^2 I)$ on prior class probabilities for individual sample functions $p(f(x; w))$, the predictive distribution, and posterior samples as we observe varying amounts of data. In Section G, we further study the correlation structure over images induced by neural network priors, subject to perturbations of the images. In Section D.3 we provide additional experimental details.

5.1. Deep Image Prior and Random Network Features

Two recent results provide strong evidence that vague Gaussian priors over parameters, when combined with a neural network architecture, induce a distribution over functions with useful inductive biases. In the *deep image prior*, Ulyanov et al. (2018) show that *randomly initialized* convolutional neural networks *without training* provide excellent performance for image denoising, super-resolution, and inpainting. This result demonstrates the ability for a sample function from a random prior over neural networks $p(f(x; w))$ to capture low-level image statistics, before any training. Similarly, Zhang et al. (2016) shows that pre-

processing CIFAR-10 with a *randomly initialized untrained* convolutional neural network dramatically improves the test performance of a simple Gaussian kernel on pixels from 54% accuracy to 71%. Adding ℓ_2 regularization only improves the accuracy by an additional 2%. These results again indicate that *broad* Gaussian priors over parameters induce reasonable priors over networks, with a minor additional gain from decreasing the variance of the prior in parameter space, which corresponds to ℓ_2 regularization.

5.2. Prior Class Correlations

In Figure 6 we study the prior correlations in the outputs of the LeNet-5 convolutional network (LeCun et al., 1998) on objects of different MNIST classes. We sample networks with weights $p(w) = \mathcal{N}(0, \alpha^2 I)$, and compute the values of logits corresponding to the first class for all pairs of images and compute correlations of these logits. For all levels of α the correlations between objects corresponding to the same class are consistently higher than the correlation between objects of different classes, showing that the network induces a reasonable prior similarity metric over these images. Additionally, we observe that the prior correlations somewhat decrease as we increase α, showing that bounding the norm of the weights has some minor utility, in accordance with Section 5.1. Similarly, in panel (d) we see that the NLL significantly decreases as α increases in $[0, 0.5]$, and then slightly increases, but is relatively constant thereafter.

In the Appendix, we further describe analytic results and illustrate the effect of α on sample functions.

5.3. Effect of Prior Variance on CIFAR-10

We further study the effect of the parameter prior standard deviation α, measuring performance of approximate Bayesian inference for CIFAR-10 with a Preactivation ResNet-20 (He et al., 2016) and VGG-16 (Simonyan & Zisserman, 2014). For each of these architectures we run SWAG (Maddox et al., 2019) with fixed hyper-parameters and varying α. We report the results in Figure 9(d), (h). For both architectures, the performance is near-optimal in the range $\alpha \in [10^{-2}, 10^{-1}]$. Smaller α constrains the weights too much. Performance is reasonable and becomes mostly insensitive to α as it continues to increase, due to the inductive biases of the functional form of the neural network.

6. Rethinking Generalization

Zhang et al. (2016) demonstrated that deep neural networks have sufficient capacity to fit randomized labels on popular image classification tasks, and suggest this result requires re-thinking generalization to understand deep learning.

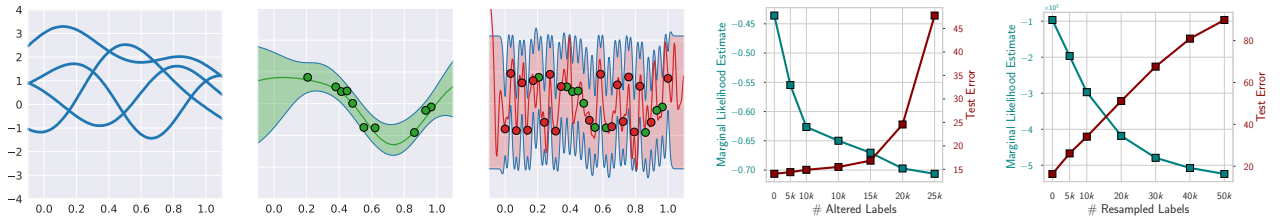
We argue, however, that this behaviour is not puzzling from a probabilistic perspective, is not unique to neural networks, and cannot be used as evidence against Bayesian neural networks (BNNs) with vague parameter priors. Fundamentally, the resolution is the view presented in the introduction: from a probabilistic perspective, generalization is at least a *two-dimensional* concept, related to support (flexibility), which should be as large as possible, supporting even noisy solutions, and inductive biases that represent relative prior probabilities of solutions.

Indeed, we demonstrate that the behaviour in Zhang et al. (2016) that was treated as mysterious and specific to neural networks can be exactly reproduced by Gaussian processes (GPs). Gaussian processes are an ideal choice for this experiment, because they are popular Bayesian non-parametric models, and they assign a prior directly in function space. Moreover, GPs have remarkable flexibility, providing universal approximation with popular covariance functions such as the RBF kernel. Yet the functions that are a priori *likely* under a GP with an RBF kernel are relatively simple. We describe GPs further in the Appendix, and Rasmussen & Williams (2006) provides an extensive introduction.

We start with a simple example to illustrate the ability for a GP with an RBF kernel to easily fit a corrupted dataset, yet generalize well on a non-corrupted dataset, in Figure 7. In Fig 7(a), we have sample functions from a GP prior over functions $p(f(x))$, showing that likely functions under the prior are smooth and well-behaved. In Fig 7(b) we see the GP is able to reasonably fit data from a structured function. And in Fig 7(c) the GP is also able to fit highly corrupted data, with essentially no structure; although these data are not a likely draw from the prior, the GP has support for a wide range of solutions, including noise.

We next show that GPs can replicate the generalization behaviour described in Zhang et al. (2016) (experimental details in the Appendix). When applied to CIFAR-10 images with random labels, *Gaussian processes achieve 100% train accuracy*, and 10.4% test accuracy (at the level of random guessing). However, the same model trained on the true labels achieves a training accuracy of 72.8% and a test accuracy of 54.3%. Thus, the generalization behaviour described in Zhang et al. (2016) is not unique to neural networks, and can be described by separately understanding the support and the inductive biases of a model.

Indeed, although Gaussian processes support CIFAR-10 images with random labels, they are not likely under the GP prior. In Fig 7(d), we compute the approximate GP marginal likelihood on a binary CIFAR-10 classification problem, with labels of varying levels of corruption. We see as the noise in the data increases, the approximate marginal likelihood, and thus the prior support for these data, decreases. In Fig 7(e), we see a similar trend for a Bayesian neural network. Again, as the fraction of corrupted labels increases, the approximate marginal likelihood decreases,



(a) Prior Draws (b) True Labels (c) Corrupted Labels (d) Gaussian Process (e) PreResNet-20

Figure 7. **Rethinking generalization. (a)**: Sample functions from a Gaussian process prior. **(b)**: GP fit (with 95% credible region) to structured data generated as $y_{\text{green}}(x) = \sin(x \cdot 2\pi) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, 0.2^2)$. **(c)**: GP fit, with no training error, after a significant addition of corrupted data in red, drawn from Uniform$[0.5, 1]$. **(d)**: Variational GP marginal likelihood with RBF kernel for two classes of CIFAR-10. **(e)**: Laplace BNN marginal likelihood for a PreResNet-20 on CIFAR-10 with different fractions of random labels. The marginal likelihood for both the GP and BNN decreases as we increase the level of corruption in the labels, suggesting reasonable inductive biases in the prior over functions. Moreover, both the GP and BNN have 100% training accuracy on images with fully corrupted labels.

showing that the prior over functions given by the Bayesian neural network has less support for these noisy datasets. We provide further experimental details in the Appendix.

7. Discussion

> *"It is now common practice for Bayesians to fit models that have more parameters than the number of data points... Incorporate every imaginable possibility into the model space: for example, if it is conceivable that a very simple model might be able to explain the data, one should include simple models; if the noise might have a long-tailed distribution, one should include a hyperparameter which controls the heaviness of the tails of the distribution; if an input variable might be irrelevant to a regression, include it in the regression anyway."* MacKay (1995)

We have presented a probabilistic perspective of generalization, which depends on the support and inductive biases of the model. The support should be as large possible, but the inductive biases must be well-calibrated to a given problem class. We argue that Bayesian neural networks embody these properties — and through the lens of probabilistic inference, explain generalization behaviour that has previously been viewed as mysterious. Moreover, we argue that Bayesian marginalization is particularly compelling for neural networks, show how deep ensembles provide a practical mechanism for marginalization, and propose a new approach that generalizes deep ensembles to marginalize within basins of attraction.

There are certainly many challenges to estimating the integral for a Bayesian model average in modern deep learning, including a high-dimensional parameter space, and a complex posterior landscape. But viewing the challenge indeed as an integration problem, rather than an attempt to obtain posterior samples for a simple Monte Carlo approximation, provides opportunities for future progress. Bayesian deep learning has been making fast practical advances, with approaches that now enable better accuracy and calibration over standard training, with minimal overhead.

We finish with remarks about future developments for Bayesian neural network priors, and approaches to research in Bayesian deep learning.

7.1. The Future for BNN Priors

We provide some brief remarks about future developments for BNN priors. Here we have explored relatively simple parameter priors $p(w) = \mathcal{N}(0, \alpha^2 I)$. While these priors are simple in parameter space, they interact with the neural network architecture to induce a sophisticated prior over functions $p(f(x; w))$, with many desirable properties, including a reasonable correlation structure over images. However, these parameter priors can certainly still be improved. As we have seen, even tuning the value of the signal variance α^2, an analogue of the L_2 regularization often used in deep learning, can have a noticeable affect on the induced prior over functions — though this affect is quickly modulated by data. Layer-wise priors, such that parameters in each layer have a different signal variance, are intuitive: we would expect later layers require precise determination, while parameters in earlier layers could reasonably take a range of values. But one has to be cautious; as we show in Appendix Section E, with ReLU activations different signal variances in different layers can be degenerate, combining together to affect only the output scale of the network.

A currently popular sentiment is that we should directly build function-space BNN priors, often taking inspiration from Gaussian processes. While we believe this is a promising direction, one should proceed with caution. If we contrive priors over parameters $p(w)$ to induce distributions

over functions $p(f)$ that resemble familiar models such as Gaussian processes with RBF kernels, we could be throwing the baby out with the bathwater. Neural networks are useful as their own model class precisely because they have different inductive biases from other models.

A similar concern applies to taking infinite width limits in Bayesian neural networks. In these cases we recover Gaussian processes with interpretable kernel functions; because these models are easier to use and analyze, and give rise to interpretable and well-motivated priors, it is tempting to treat them as drop-in replacements for the parametric analogues. However, the kernels for these models are *fixed*. In order for a model to do effective representation learning, we must learn a similarity metric for the data. Training a neural network in many ways is like *learning* a kernel, rather than using a fixed kernel. MacKay (1998) has also expressed concerns in treating these limits as replacements for neural networks, due to the loss of representation learning power.

Perhaps the distribution over functions induced by a network in combination with a generic distribution over parameters $p(w)$ may be hard to interpret — but this distribution will contain the equivariance properties, representation learning abilities, and other biases that make neural networks a compelling model class in their own right.

7.2. "But is it *really* Bayesian?"

We finish with an editorial comment about approaches to research within Bayesian deep learning. There is sometimes a tendency to classify work as *Bayesian* or *not Bayesian*, with very stringent criteria for what qualifies as *Bayesian*. Moreover, the implication, and sometimes even explicit recommendation, is that if an approach is not unequivocally Bayesian in every respect, then we should not term it as Bayesian, and we should instead attempt to understand the procedure through entirely different non-Bayesian mechanisms. We believe this mentality encourages tribalism, which is not conducive to the best research, or creating the best performing methods. What matters is not a debate about semantics, but making rational modelling choices given a particular problem setting, and trying to understand these choices. Often these choices can largely be inspired by a Bayesian approach — in which case it desirable to indicate this source of inspiration. And in the semantics debate, who would be the arbiter of what gets to be called Bayesian? Arguably it ought to be an evolving definition.

Broadly speaking, what makes Bayesian approaches distinctive is a posterior weighted marginalization over parameters. And at a high level, Bayesian methods are about combining our honest beliefs with data to form a posterior. In actuality, no fair-minded researcher entirely believes the prior over parameters, the functional form of the model (which is part of the prior over functions), or the likelihood. From this perspective, it is broadly compatible with a Bayesian philosophy to reflect misspecification in the modelling procedure itself, which is achieved through tempering. In this sense, the *tempered posterior* is more reflective of a *true posterior* than the posterior that results from ignoring our belief that the model is misspecified.

Moreover, basic probability theory indicates that marginalization is desirable. While marginalization cannot in practice be achieved exactly, we can try to improve over conventional training, which as we have discussed can be viewed as approximate marginalization. Given computational constraints, effective marginalization is not equivalent to obtaining accurate samples from a posterior. As we have discussed, simple Monte Carlo is only one of many mechanisms for marginalization. Just like we how expectation propagation (Minka, 2001a) focuses its approximation to factors in a posterior where it will most affect the end result, we should focus on representing the posterior where it will make the biggest difference to the model average. As we have shown, deep ensembles are a reasonable mechanism up to a point. After having trained many independent models, there are added benefits to marginalizing within basins, given the computational expense associated with retraining an additional model to find an additional basin of attraction.

We should also not hold Bayesian methods to a double standard. Indeed, it can be hard to interpret or understand the prior, the posterior, and whether the marginalization procedure is optimal. But it is also hard to interpret the choices behind the functional form of the model, or the rationale behind classical procedures where we bet everything on a single global optimum — when we know there are many global optima and many of them will perform well but provide different solutions, and many others will not perform well. We should apply the same level of scrutiny to all modelling choices, consider the alternatives, and not be paralyzed if a procedure is not optimal in every respect.

References

Barron, A. R. and Cover, T. M. Minimum complexity density estimation. *IEEE transactions on information theory*, 37(4):1034–1054, 1991.

Beal, M. J. *Variational algorithms for approximate Bayesian inference*. university of London, 2003.

Bishop, C. M. *Pattern Recognition and Machine Learning*. Springer, 2006.

Bissiri, P. G., Holmes, C. C., and Walker, S. G. A general framework for updating belief distributions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 78(5):1103–1130, 2016.

Box, G. E. and Tiao, G. C. Bayesian inference in statistical analysis, addision-wesley. *Reading, MA*, 1973.

Cobb, A. D., Baydin, A. G., Markham, A., and Roberts, S. J. Introducing an explicit symplectic integration scheme for riemannian manifold hamiltonian monte carlo. *arXiv preprint arXiv:1910.06243*, 2019.

Darnieder, W. F. *Bayesian methods for data-dependent priors*. PhD thesis, The Ohio State University, 2011.

de Heide, R., Kirichenko, A., Mehta, N., and Grünwald, P. Safe-Bayesian generalized linear regression. *arXiv preprint arXiv:1910.09227*, 2019.

Dinh, L., Pascanu, R., Bengio, S., and Bengio, Y. Sharp minima can generalize for deep nets. *arXiv preprint arXiv:1703.04933*, 2017.

Draxler, F., Veschgini, K., Salmhofer, M., and Hamprecht, F. A. Essentially no barriers in neural network energy landscape. *arXiv preprint arXiv:1803.00885*, 2018.

Fort, S. and Jastrzebski, S. Large scale structure of neural network loss landscapes. In *Advances in Neural Information Processing Systems*, pp. 6706–6714, 2019.

Fort, S., Hu, H., and Lakshminarayanan, B. Deep ensembles: A loss landscape perspective. *arXiv preprint arXiv:1912.02757*, 2019.

Gal, Y. and Ghahramani, Z. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059, 2016.

Gardner, J. R., Pleiss, G., Bindel, D., Weinberger, K. Q., and Wilson, A. G. GPyTorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration. In *Neural Information Processing Systems*, 2018.

Garipov, T., Izmailov, P., Podoprikhin, D., Vetrov, D. P., and Wilson, A. G. Loss surfaces, mode connectivity, and fast ensembling of DNNs. In *Neural Information Processing Systems*, 2018.

Grünwald, P. The safe Bayesian. In *International Conference on Algorithmic Learning Theory*, pp. 169–183. Springer, 2012.

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

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1321–1330. JMLR. org, 2017.

Gustafsson, F. K., Danelljan, M., and Schön, T. B. Evaluating scalable bayesian deep learning methods for robust computer vision. *arXiv preprint arXiv:1906.01620*, 2019.

Hafner, D., Tran, D., Irpan, A., Lillicrap, T., and Davidson, J. Reliable uncertainty estimates in deep neural networks using noise contrastive priors. *arXiv preprint arXiv:1807.09289*, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Hochreiter, S. and Schmidhuber, J. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

Huang, W. R., Emam, Z., Goldblum, M., Fowl, L., Terry, J. K., Huang, F., and Goldstein, T. Understanding generalization through visualizations. *arXiv preprint arXiv:1906.03291*, 2019.

Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D., and Wilson, A. G. Averaging weights leads to wider optima and better generalization. In *Uncertainty in Artificial Intelligence (UAI)*, 2018.

Izmailov, P., Maddox, W. J., Kirichenko, P., Garipov, T., Vetrov, D., and Wilson, A. G. Subspace inference for Bayesian deep learning. *Uncertainty in Artificial Intelligence*, 2019.

Jacot, A., Gabriel, F., and Hongler, C. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pp. 8571–8580, 2018.

Kass, R. E. and Raftery, A. E. Bayes factors. *Journal of the American Statistical Association*, 90(430):773–795, 1995.

Kendall, A. and Gal, Y. What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pp. 5574–5584, 2017.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

Khan, M. E., Nielsen, D., Tangkaratt, V., Lin, W., Gal, Y., and Srivastava, A. Fast and scalable Bayesian deep learning by weight-perturbation in adam. *arXiv preprint arXiv:1806.04854*, 2018.

Krizhevsky, A., Nair, V., and Hinton, G. The CIFAR-10 dataset. 2014. http://www.cs.toronto.edu/kriz/cifar.html.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pp. 6402–6413, 2017.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Louizos, C., Shi, X., Schutte, K., and Welling, M. The functional neural process. In *Advances in Neural Information Processing Systems*, 2019.

MacKay, D. J. *Bayesian methods for adaptive models*. PhD thesis, California Institute of Technology, 1992.

MacKay, D. J. Probable networks and plausible predictions?a review of practical Bayesian methods for supervised neural networks. *Network: computation in neural systems*, 6(3):469–505, 1995.

MacKay, D. J. Introduction to Gaussian processes. In Bishop, C. M. (ed.), *Neural Networks and Machine Learning*, chapter 11, pp. 133–165. Springer-Verlag, 1998.

MacKay, D. J. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

Maddox, W. J., Izmailov, P., Garipov, T., Vetrov, D. P., and Wilson, A. G. A simple baseline for Bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems 32*, pp. 13153–13164. Curran Associates, Inc., 2019.

Minka, T. Expectation propagation for approximate Bayesian inference. In *Uncertainty in Artificial Intelligence*, volume 17, pp. 362–369, 2001a.

Minka, T. P. Bayesian model averaging is not model combination. 2000.

Minka, T. P. Automatic choice of dimensionality for pca. In *Advances in neural information processing systems*, pp. 598–604, 2001b.

Mohri, M. and Rostamizadeh, A. Rademacher complexity bounds for non-iid processes. In *Advances in Neural Information Processing Systems*, pp. 1097–1104, 2009.

Nalisnick, E. *On priors for Bayesian neural networks*. PhD thesis, University of California, Irvine, 2018.

Neal, R. *Bayesian Learning for Neural Networks*. Springer Verlag, 1996. ISBN 0387947248.

Ovadia, Y., Fertig, E., Ren, J., Nado, Z., Sculley, D., Nowozin, S., Dillon, J. V., Lakshminarayanan, B., and Snoek, J. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. *arXiv preprint arXiv:1906.02530*, 2019.

Pearce, T., Zaki, M., Brintrup, A., Anastassacos, N., and Neely, A. Uncertainty in neural networks: Bayesian ensembling. *arXiv preprint arXiv:1810.05546*, 2018.

Rasmussen, C. E. and Ghahramani, Z. Occam's razor. In *Neural Information Processing Systems (NIPS)*, 2001.

Rasmussen, C. E. and Williams, C. K. I. *Gaussian processes for Machine Learning*. The MIT Press, 2006.

Ritter, H., Botev, A., and Barber, D. A scalable Laplace approximation for neural networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Rue, H., Martino, S., and Chopin, N. Approximate Bayesian inference for latent gaussian models by using integrated nested laplace approximations. *Journal of the royal statistical society: Series b (statistical methodology)*, 71(2): 319–392, 2009.

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

Sun, S., Zhang, G., Shi, J., and Grosse, R. Functional variational Bayesian neural networks. *arXiv preprint arXiv:1903.05779*, 2019.

Ulyanov, D., Vedaldi, A., and Lempitsky, V. Deep image prior. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9446–9454, 2018.

Vapnik, V. N. Adaptive and learning systems for signal processing communications, and control. *Statistical learning theory*, 1998.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature Methods*, pp. 1–12, 2020.

Walker, S. and Hjort, N. L. On Bayesian consistency. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(4):811–821, 2001.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688, 2011.

Wenzel, F., Roth, K., Veeling, B. S., Światkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. How good is the bayes posterior in deep neural networks really? *arXiv preprint arXiv:2002.02405*, 2020.

Wilson, A. G. The case for Bayesian deep learning. *arXiv preprint arXiv:2001.10995*, 2020.

Xu, X., Lu, P., MacEachern, S., and Xu, R. Calibrated bayes factors for model comparison. *Journal of Statistical Computation and Simulation*, 89(4):591–614, 2019.

Yang, W., Lorch, L., Graule, M. A., Srinivasan, S., Suresh, A., Yao, J., Pradier, M. F., and Doshi-Velez, F. Output-constrained Bayesian neural networks. *arXiv preprint arXiv:1905.06287*, 2019.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*, 2016.

Zhang, T. Information-theoretic upper and lower bounds for statistical estimation. *IEEE Transactions on Information Theory*, 52(4):1307–1321, 2006.

Appendix Outline

This appendix is organized as follows. In Section A, we visualize predictive functions corresponding to weight samples within high posterior density valleys on a regression problem. In Section B, we provide background material on Gaussian processes. In Section C, we present further results comparing MultiSWAG and MultiSWA to Deep Ensembles under data distribution shift on CIFAR-10. In Section D, we provide the details of all experiments presented in the paper. In Section E, we present analytic results on the dependence of the prior distribution in function space on the variance of the prior over parameters. In Section F, we provide a discussion of posterior tempering in Bayesian deep learning, including considerations of the results of the recent study of Wenzel et al. (2020). In Section G, we study the prior correlations between BNN logits on perturbed images.

A. Loss Valleys

We demonstrate that different points along the valleys of high posterior density (low loss) connecting pairs of independently trained optima (Garipov et al., 2018; Draxler et al., 2018; Fort & Jastrzebski, 2019) correspond to different predictive functions. We use the regression example from Izmailov et al. (2019) and show the results in Figure 8.

B. Gaussian processes

With a Bayesian neural network, a distribution over parameters $p(w)$ induces a distribution over functions $p(f(x;w))$ when combined with the functional form of the network. Gaussian processes (GPs) are often used to instead *directly* specify a distribution over functions.

A Gaussian process is a distribution over functions, $f(x) \sim \mathcal{GP}(m,k)$, such that any collection of function values, queried at any finite set of inputs $x_1,\dots,x_n$, has a joint Gaussian distribution:

$$f(x_1),\dots,f(x_n) \sim \mathcal{N}(\mu, K)\,. \tag{2}$$

The mean vector, $\mu_i = \mathbb{E}[f(x_i)] = m(x_i)$, and covariance matrix, $K_{ij} = \text{cov}(f(x_i), f(x_j)) = k(x_i, x_j)$, are determined by the *mean function* m and *covariance function* (or *kernel*) k of the Gaussian process.

The popular RBF kernel has the form

$$k(x_i,x_j) = \exp\left(-\frac{1}{2\ell^2}\|x_i - x_j\|^2\right)\,. \tag{3}$$

The *length-scale* hyperparameter ℓ controls the extent of correlations between function values. If ℓ is large, sample functions from a GP prior are simple and slowly varying with inputs x.

Gaussian processes with RBF kernels (as well as many other standard kernels) assign positive density to any set of observations. Moreover, these models are *universal approximators* (Rasmussen & Williams, 2006): as the number of observations increase, they are able to approximate any function to arbitrary precision.

Work on Gaussian processes in machine learning was triggered by the observation that Bayesian neural networks become Gaussian processes with particular kernel functions as the number of hidden units approaches infinity (Neal, 1996). This result resembles recent work on the neural tangent kernel (e.g., Jacot et al., 2018).

C. Deep Ensembles and MultiSWAG Under Distribution Shift

In Figures 12, 13, 14, 15 we show the negative log-likelihood for Deep Ensembles, MultiSWA and MultiSWAG using PreResNet-20 on CIFAR-10 with various corruptions as a function of the number of independently trained models (SGD solutions, SWA solutions or SWAG models, respectively). For MultiSWAG, we generate 20 samples from each independent SWAG model. Typically MultiSWA and MultiSWAG significantly outperform Deep Ensembles when a small number of independent models is used, or when the level of corruption is high.

In Figure 16, following Ovadia et al. (2019), we show the distribution of negative log likelihood, accuracy and expected calibration error as we vary the type of corruption. We use a fixed training time budget: 10 independently trained models for every method. For MultiSWAG we ensemble 20 samples from each of the 10 SWAG approximations. MultiSWAG particularly achieves better NLL than the other two methods, and MultiSWA outperforms Deep Ensembles; the difference is especially pronounced for higher levels of corruption. In terms of ECE, MultiSWAG again outperforms the other two methods for higher corruption intensities.

We note that Ovadia et al. (2019) found Deep Ensembles to be a very strong baseline for prediction quality and calibration under distribution shift. For this reason, we focus on Deep Ensembles in our comparisons.

D. Details of Experiments

In this section we provide additional details of the experiments presented in the paper.

D.1. Approximating the True Predictive Distribution

For the results presented in Figure 4 we used a network with 3 hidden layers of size 10 each. The network takes two inputs: x and x^2. We pass both x and x^2 as input to ensure

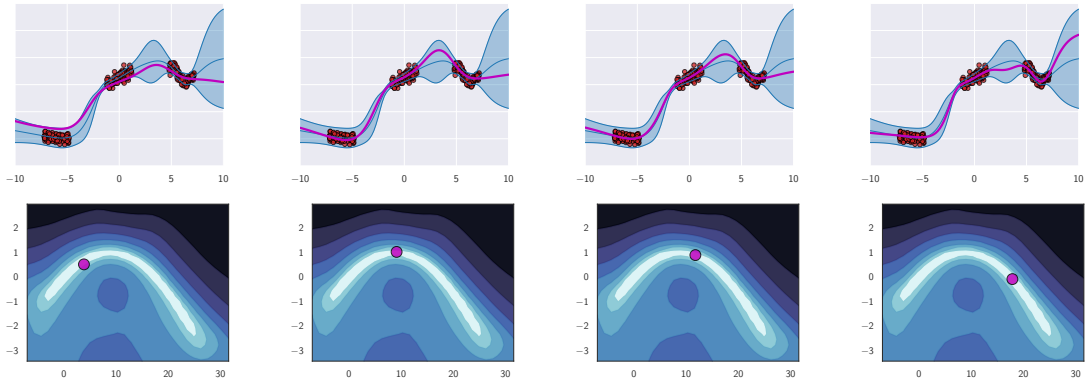


Figure 8. **Diversity of high performing functions. Bottom**: a contour plot of the posterior log-density in the subspace containing a pair of independently trained modes (as with deep ensembles), and a path of high posterior density connecting these modes. In each panel, the purple point represents a sample from the posterior in the parameter subspace. **Top**: the predictive distribution constructed from samples in the subspace. The shaded blue area shows the 3σ-region of the predictive distribution at each of the input locations, and the blue line shows the mean of the predictive distribution. In each panel, the purple line shows the predictive function corresponding to the sample shown in the corresponding bottom row panel. For the details of the experimental setup see Section 5.1 of Izmailov et al. (2019).

that the network can represent a broader class of functions. The network outputs a single number $y = f(x)$.

To generate data for the plots, we used a randomly-initialized neural network of the same architecture described above. We sampled the weights from an isotropic Gaussian with variance 0.1^2 and added isotropic Gaussian noise with variance 0.1^2 to the outputs:

$$y = f(x; w) + \epsilon(x),$$

with $w \sim \mathcal{N}(0, 0.1^2 \cdot I)$, $\epsilon(x) \sim \mathcal{N}(0, 0.1^2 \cdot I)$. The training set consists of 120 points shown in Figure 4.

For estimating the ground truth we ran 200 chains of Hamiltonian Monte Carlo (HMC) using the `hamiltorch` package (Cobb et al., 2019). We initialized each chain with a network pre-trained with SGD for 3000 steps, then ran Hamiltonian Monte Carlo (HMC) for 2000 steps, producing 200 samples.

For Deep Ensembles, we independently trained 50 networks with SGD for 20000 steps each. We used minus posterior log-density as the training loss. For SVI, we used a fully-factorized Gaussian approximation initialized at an SGD solution trained for 20000 steps. For all inference methods we set prior variance to 10^2 and noise variance to 0.02^2.

Discrepancy with true BMA. For the results presented in panel (d) of Figure 4 we computed Wasserstein distance between the predictive distribution approximated with HMC and the predictive distribution for Deep Ensembles and SVI. We used the one-dimensional Wasserstein distance function[1] from the *scipy* package (Virtanen et al., 2020). We computed the Wasserstein distance between marginal distributions at each input location, and averaged the results over the input locations. In the top sub-panels of panels (b), (c) of Figure 4 we additionally visualize the marginal Wasserstein distance between the HMC predictive distribution and Deep Ensembles and SVI predictive distrbutions respectively for each input location.

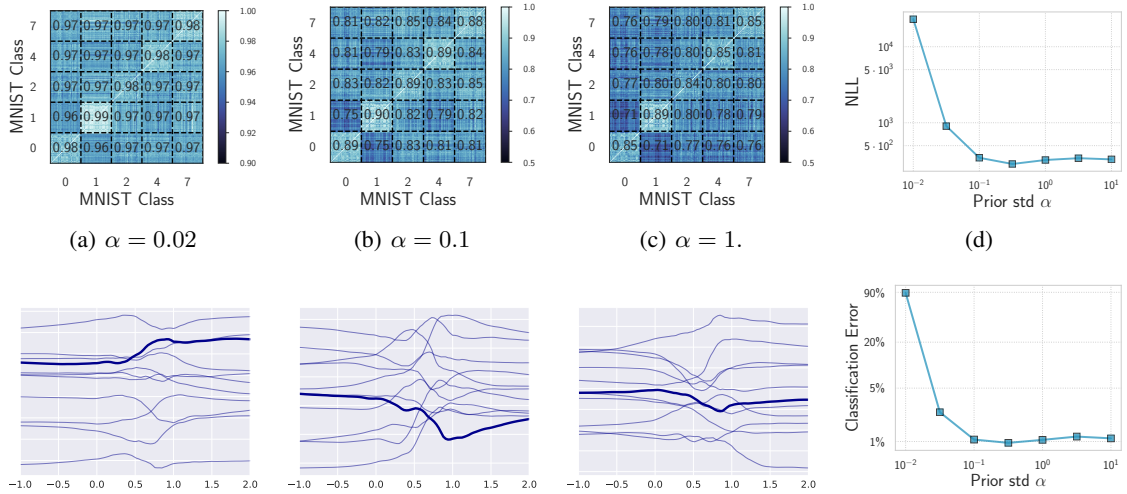
D.2. Deep Ensembles and MultiSWAG

We evaluate Deep Ensembles, MultiSWA and MultiSWAG under distribution shift in Section 4. Following Ovadia et al. (2019), we use a PreResNet-20 network and the CIFAR-10 dataset with different types of corruptions introduced in Hendrycks & Dietterich (2019). For training individual SGD, SWA and SWAG models we use the hyper-parameters used for PreResNet-164 in Maddox et al. (2019). For each SWAG model we sample 20 networks and ensemble them. So, Deep Ensembles, MultiSWA and MultiSWAG are all evaluated under the same training budget; Deep Ensembles and MultiSWA also use the same test-time budget.

For producing the corrupted data we used the code[2] released by Hendrycks & Dietterich (2019). We had issues producing the data for the *frost* corruption type, so we omit it in our evaluation, and include *Gaussian blur* which was not included in the evaluation of Hendrycks & Dietterich (2019).

[1] `https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.wasserstein_distance.html`

[2] `https://github.com/hendrycks/robustness/blob/master/ImageNet-C/create_c/make_cifar_c.py`



(a) $\alpha = 0.02$ (b) $\alpha = 0.1$ (c) $\alpha = 1.$ (d)

(e) $\alpha = 0.02$ (f) $\alpha = 0.1$ (g) $\alpha = 1$ (h)

Figure 9. **(a)–(c)**: Average pairwise prior correlations for pairs of objects in classes $\{0, 1, 2, 4, 7\}$ of MNIST induced by LeNet-5 for $p(f(x; w))$ when $p(w) = \mathcal{N}(0, \alpha^2 I)$. Images in the same class have higher prior correlations than images from different classes, suggesting that $p(f(x; w))$ has desirable inductive biases. The correlations slightly decrease with increases in α. Panels **(e)–(g)** show sample functions from LeNet-5 along the direction connecting a pair of MNIST images of 0 and 1 digits. The complexity of the samples increases with α. **(d)**: NLL and **(h)** classification error of an ensemble of 20 SWAG samples on MNIST as a function of α using a LeNet-5. The NLL is high for overly small α and near-optimal for larger values with an optimum near $\alpha = 0.3$.

D.3. Neural Network Priors

In the main text we considered different properties of the prior distribution over functions induced by a spherical Gaussian distribution over the weights, with different variance scales.

Prior correlation diagrams. In panels (a)–(c) of Figure 9 we show pairwise correlations of the logits for different pairs of datapoints. To make these plots we produce $S = 100$ samples of the weights w_i of a LeNet-5 from the prior distribution $\mathcal{N}(0, \alpha^2 I)$ and compute the logits corresponding to class 0 for each data point and each weight sample. We then compute the correlations for each pair x, x' of data points as follows:

$$\text{corr}_{\text{logit}}(x, x') = \frac{\sum_{i=1}^{S}(f(x, w_i) - \bar{f}(x))(f(x', w_i) - \bar{f}(x'))}{\sqrt{\sum_{i=1}^{S}(f(x, w_i) - \bar{f}(x))^2 \cdot \sum_{i=1}^{S}(f(x', w_i) - \bar{f}(x'))^2}},$$

where $f(x, w)$ is the logit corresponding to class 0 of the network with weights w on the input x, and $\bar{f}(x)$ is the mean value of the logit $\bar{f}(x) = \frac{1}{S}\sum_i f(x, w_i)$. For evaluation, we use 200 random datapoints per class for classes $0, 1, 2, 4, 7$ (a total of 1000 datapoints). We use this set of classes to ensure that the structure is clearly visible in the figure. We combine the correlations into a diagram, additionally showing the average correlation for each pair of classes. We repeat the experiment for different values of $\alpha \in \{0.02, 0.1, 1\}$. For a discussion of the results see Section 5.2.

Sample functions. In panels (e)–(g) of Figure 9 we visualize the functions sampled from the LeNet-5 network along the direction connecting a pair of MNIST images. In particular, we take a pair of images x_0 and x_1 of digits 0 and 1, respectively, and construct the path $x(t) = t \cdot x_0 + (1 - t) \cdot x_1$. We then study the samples of the logits $z(t) = f(x(t) \cdot \|x_0\|/\|x(t)\|, w)$ along the path; here we adjusted the norm of the images along the path to be constant as the values of the logits are sensitive to the norm of the inputs. The complexity of the samples increases as we increase the variance of the prior distribution over the weights. This increased complexity of sample functions explains why we might expect the prior correlations for pairs of images to be lower when we increase the variance of the prior distribution over the weights.

Performance dependence on prior variance. In panels (d), (h) of Figure 9 we show the test negative log-likelihood and accuracy of SWAG applied to LeNet-5 on MNIST. We train the model for 50 epochs, constructing the rank-20 SWAG approximation from the last 25 epochs. We use an initial learning rate of 0.05 and SWAG learning rate of 0.01 with the learning rate schedule of Maddox et al. (2019). We use posterior log-density as the objective, and vary the prior variance α^2. In panels (f), (g) of Figure 10

we perform an analogous experiment using a PreResNet-20 and a VGG-16 on CIFAR-10, using the hyper-parameters reported in Maddox et al. (2019) (for PreResNet-20 we use the hyper-parameters used with PreResNet-164 in Maddox et al. (2019)). Both on MNIST and CIFAR-10 we observe that the performance is poor for overly small values of α, close to optimal for intermediate values, and still reasonable for larger values of α. For further discussion of the results see Section 5.3.

Predictions from prior samples. Following Wenzel et al. (2020) we study the predictive distributions of prior samples using PreResNet-20 on CIFAR-10. In Figure 10 we show the sample predictive functions averaged over datapoints for different scales α of the prior distribution. We also show the predictive distribution for each α, which is the average of the sample predictive distributions over 200 samples of weights. In Figure 11 we show how the predictive distribution changes as we vary the number of observed data for prior scale $\alpha = \sqrt{10}$. We see that the marginal predictive distribution for all considered values of α is reasonable — roughly uniform across classes, when averaged across the dataset. For the latter experiment we used stochastic gradient Langevin dynamics (SGLD) (Welling & Teh, 2011) with a cosine lerning rate schedule. For each sample we restart SGLD, and we only use the sample obtained at the last iteration. We discuss the results in Section F.4.

Prior correlations with corrupted images. In Section G and Figure 17 we study the decay of the prior correlations between logits on an original image and a perturbed image as we increase the intensity of perturbations. For the BNN we use PreResNet-20 architecture with the standard Gaussian prior $\mathcal{N}(0, I)$. For the linear model, the correlations are not affected by the prior variance α^2:

$$cov(w^T x, w^T y) = \mathbb{E}(w^T x \cdot w^T y) = \mathbb{E} x^T w w^T y = x^T \mathbb{E} w w^T y = \alpha^2 x^T y,$$

and hence

$$corr(w^T x, w^T y) = \frac{cov(w^T x, w^T y)}{\sqrt{cov(w^T y, w^T y) \cdot cov(w^T x, w^T x)}} = x^T y.$$

We use the $\mathcal{N}(0, I)$ prior for the weights of the linear model. Finally, we also evaluate the correlations associated with an RBF kernel (see Equation (3)). To set the lengthscale ℓ of the kernel we evaluate the pairwise correlations for the PreResnet-20 and RBF kernel on the 100 uncorrupted CIFAR-10 images that were used for the experiment, and ensure that the average correlations match. The resulting value of ℓ is 10000, and the average correlation for the RBF kernel and PreResNet was ≈ 0.9; for the linear model the average correlation was ≈ 0.82. For the perturbations we used the same set of corruptions introduced in Hendrycks & Dietterich (2019) as in the experiments in Section 4 with the addition of a random translation: for a random translation of intensity i we pad the image with $2 \cdot i$ zeros on each side and crop the image randomly to 32×32.

D.4. Rethinking Generalization

In Section 6, we experiment with Bayesian neural networks and Gaussian processes on CIFAR-10 with noisy labels, inspired by the results in Zhang et al. (2016) that suggest we need to re-think generalization to understand deep learning.

Following Zhang et al. (2016), we train PreResNet-20 on CIFAR-10 with different fractions of random labels. To ensure that the networks fits the train data, we turn off weight decay and data augmentation, and use a lower initial learning rate of 0.01. Otherwise, we follow the hyper-parameters that were used with PreResNet-164 in Maddox et al. (2019). We use diagonal Laplace approximation to compute an estimate of marginal likelihood for each level of label corruption. Following Ritter et al. (2018) we use the diagonal of the Fisher information matrix rather than the Hessian.

We perform a similar experiment with a Gaussian process with RBF kernel on the binary classification problem for two classes of CIFAR-10. We use variational inference to fit the model, and we use the variational evidence lower bound to approximate the marginal likelihood. We use variational inference to overcome the non-Gaussian likelihood and not for scalability reasons; i.e., we are not using inducing inputs. We use the `GPyTorch` package (Gardner et al., 2018) to train the models. We use an RBF kernel with default initialization from `GPyTorch` and divide the inputs by 5000 to get an appropriate input scale. We train the model on a binary classification problem between classes 0 and 1.

For the 10-class GP classification experiment we train 10 one-vs-all models that classify between a given class and the rest of the data. To reduce computation, in training we subsample the data not belonging to the given class to $10k$ datapoints, so each model is trained on a total of $15k$ datapoints. We then combine the 10 models into a single multi-class model: an observation is attributed to the class that corresponds to the one-vs-all model with the highest confidence. We use the same hyper-parameters as in the binary classification experiments.

E. Analysis of Prior Variance Effect

In this section we provide simple analytic results for the effect of prior variance in ReLU networks. A related derivation is presented in the Appendix Section A.8 of Garipov et al. (2018) about connecting paths from symmetries in parametrization.

We will consider a multilayer network $f(x, w)$ of the form

$$f(x, \{W_i, b_i\}_{i=1}^n) = \\ W_n(\dots \phi(W_2 \phi(W_1 x + b_1) + b_2)) + b_n,$$

where ϕ is the ReLU (or in fact any positively-homogeneous activation function), W_i are weight matrices and b_i are bias vectors. In particular, f can be a regular CNN with ReLU activations up to the logits (with softmax activation removed).

Now, suppose we have a prior distribution of the form

$$W_i \sim \mathcal{N}(0, \alpha_i^2 I), \;\; b_i \sim \mathcal{N}(0, \beta_i^2 I),$$

where the identity matrices I are implicitly assumed to be of appropriate shapes, so each weight matrix and bias vector has a spherical Gaussian distribution. We can reparameterize this distribution as

$$W_i = \alpha_i \mathcal{E}_i, \;\; \mathcal{E}_i \sim \mathcal{N}(0, I), \\ b_i = \beta_i \epsilon_i, \;\; \epsilon_i \sim \mathcal{N}(0, I).$$

We can then express the predictions of the network on the input x for weights sampled from the prior as the random variable

$$f(x, \{\alpha_i, \beta_i\}_{i=1}^n) = \\ \alpha_n \cdot \mathcal{E}_n(\dots \phi(\alpha_1 \mathcal{E}_1 x + \beta_1 \cdot \epsilon_1)) + \beta_n \cdot \epsilon_n. \tag{4}$$

Through Equation (4), we can observe some simple properties of the dependence between the prior scales α_i, β_i and the induced function-space prior.

Proposition 1. *Suppose the network has no bias vectors, i.e. $\beta_1 = \dots = \beta_n = 0$. Then the scales α_i of the prior distribution over the weights only affect the output scale of the network.*

Proof. In the case when there are no bias vectors Equation (4) simplifies to

$$f(x, \{\alpha_i, \beta_i = 0\}_{i=1}^n) = \\ \alpha_n \cdot \mathcal{E}_n(\dots \phi(\alpha_1 \mathcal{E}_1 x + \beta_1 \cdot \epsilon_1)) + \beta_n \cdot \epsilon_n = \\ \alpha_n \cdot \dots \cdot \alpha_1 \cdot \mathcal{E}_n(\dots \phi(\mathcal{E}_1 x)) = \\ \alpha_n \cdot \dots \cdot \alpha_1 \cdot f(x, \{\alpha_i = 1, \beta_i = 0\}_{i=1}^n).$$

In the derivation above we used positive homogeneity of ReLU: $\phi(\alpha z) = \alpha \phi(z)$ for any positive α. □

In other words, to sample from the distribution over functions corresponding to a prior with variances $\{\alpha_i, \beta_i = 0\}_{i=1}^n$, we can sample from the spherical Gaussian prior (without bias terms) $\{\alpha_i = 1, \beta_i = 0\}_{i=1}^n$ and then rescale the outputs of the network by the product of variances $\alpha_n \cdot \dots \cdot \alpha_2 \cdot \alpha_1$.

We note that the result above is different from the results for *sigmoid* networks considered in MacKay (1995), where varying the prior on the weights leads to changing the length-scale of the sample functions. For ReLU networks without biases, increasing prior variance only increases the output scale of the network and not the complexity of the samples. If we apply the softmax activation on the outputs of the last layer of such network, we will observe increasingly confident predictions as we increase the prior variance. We observe this effect in Figure 10 and discuss it in Section F.4.

In case bias vectors are present, we can obtain a similar result using a specific scaling of the prior variances with layer, as in the following proposition.

Proposition 2. *Suppose the prior scales depend on the layer of the network as follows for some $\gamma > 0$:*

$$\alpha_i = \gamma, \;\; \beta_i = \gamma^i,$$

for all layers $i = 1 \dots n$. Then γ only affects the scale of the predictive distribution at any input x:

$$f(x, \{\alpha_i = \gamma, \beta_i = \gamma^i\}_{i=1}^n) = \gamma^n \cdot f(x, \{\alpha_i = 1, \beta_i = 1\}_{i=1}^n).$$

Proof. The proof is analogous to the proof of Proposition 1. We can use the positive homogenety of ReLU activations to factor the prior scales outside of the network:

$$f(x, \{\alpha_i = \gamma, \beta_i = \gamma^i\}_{i=1}^n) = \\ \gamma \cdot \mathcal{E}_n(\dots \phi(\gamma \cdot \mathcal{E}_1 x + \gamma \cdot \epsilon_1)) + \gamma^n \cdot \epsilon_n = \\ \gamma^n \cdot \big(\mathcal{E}_n(\dots \phi(\mathcal{E}_1 x + \epsilon_1))) + \epsilon_n\big) = \\ \gamma^n \cdot f(x, \{\alpha_i = 1, \beta_i = 1\}_{i=1}^n).$$

□

The analysis above can be applied to other simple scaling rules of the prior, e.g.

$$f(x, \{\alpha_i = \gamma \hat{\alpha}_i, \beta_i = \gamma^i \hat{\beta}_i\}_{i=1}^n) = \\ \gamma^n \cdot f(x, \{\alpha_i = \hat{\alpha}_i, \beta_i = \hat{\beta}_i\}_{i=1}^n), \tag{5}$$

can be shown completely analogously to Proposition 2.

More general types of scaling of the prior affect both the output scale of the network and also the relative effect of prior and variance terms. For example, by Equation (5) we have

$$f(x, \{\alpha_i = \gamma, \beta_i = \gamma\}_{i=1}^n) = \\ f(x, \{\alpha_i = \gamma \cdot 1, \beta_i = \gamma^i \cdot \gamma^{1-i}\}_{i=1}^n) = \\ \gamma^n \cdot f(x, \{\alpha_i = 1, \beta_i = \gamma^{1-i}\}_{i=1}^n).$$

We note that the analysis does not cover residual connections and batch normalization, so it applies to LeNet-5 but cannot be directly applied to PreResNet-20 networks used in many of our experiments.

F. Temperature Scaling

The standard Bayesian posterior distribution is given by

$$p(w|\mathcal{D}) = \frac{1}{Z} p(\mathcal{D}|w)p(w), \tag{6}$$

where $p(\mathcal{D}|w)$ is a likelihood, $p(w)$ is a prior, and Z is a normalizing constant.

In Bayesian deep learning it is typical to consider the *tempered* posterior

$$p_T(w|\mathcal{D}) = \frac{1}{Z(T)} p(\mathcal{D}|w)^{1/T} p(w), \tag{7}$$

where T is a *temperature* parameter, and $Z(T)$ is the normalizing constant corresponding to temperature T. The temperature parameter controls how the prior and likelihood interact in the posterior:

- $T < 1$ corresponds to *cold posteriors*, where the posterior distribution is more concentrated around solutions with high likelihood.
- $T = 1$ corresponds to the standard Bayesian posterior distribution.
- $T > 1$ corresponds to *warm posteriors*, where the prior effect is stronger and the posterior collapse is slower.

Tempering posteriors is a well-known practice in statistics, where it goes by the names *Safe Bayes*, *generalized Bayesian inference*, and *fractional Bayesian inference* (e.g., de Heide et al., 2019; Grünwald et al., 2017; Barron & Cover, 1991; Walker & Hjort, 2001; Zhang, 2006; Bissiri et al., 2016; Grünwald, 2012). Safe Bayes has been shown to be natural from a variety of perspectives, including from prequential, learning theory, and minimum description length frameworks (e.g., Grünwald et al., 2017).

Concurrently with our work, Wenzel et al. (2020) noticed that successful Bayesian deep learning methods tend to use cold posteriors. They provide an empirical study that shows that Bayesian neural networks (BNNs) with cold posteriors outperform models with SGD based maximum likelihood training, while BNNs with $T = 1$ can perform worse than the maximum likelihood solution. They claim that cold posteriors sharply deviate from the Bayesian paradigm, and consider possible reasons for why tempering is helpful in Bayesian deep learning.

In this section, we provide an alternative view and argue that tempering is not at odds with Bayesian principles. Moreover, for virtually any realistic model class and dataset, it would be highly surprising if $T = 1$ *were* in fact the best setting of this hyperparameter. Indeed, as long as it is practically convenient, we would advocate tempering for essentially *any* model, especially parametric models that do not scale their capacity automatically with the amount of available information. Our position is that at a high level Bayesian methods are trying to combine honest beliefs with data to form a posterior. By reflecting the belief that the model is misspecified, the tempered posterior is often more of a *true posterior* than the posterior that results from ignoring our belief that the model misspecified.

Finding that $T < 1$ helps for Bayesian neural networks is neither surprising nor discouraging. And the actual results of the experiments in Wenzel et al. (2020), which show great improvements over standard SGD training, are in fact very encouraging of deriving inspiration from Bayesian procedures in deep learning.

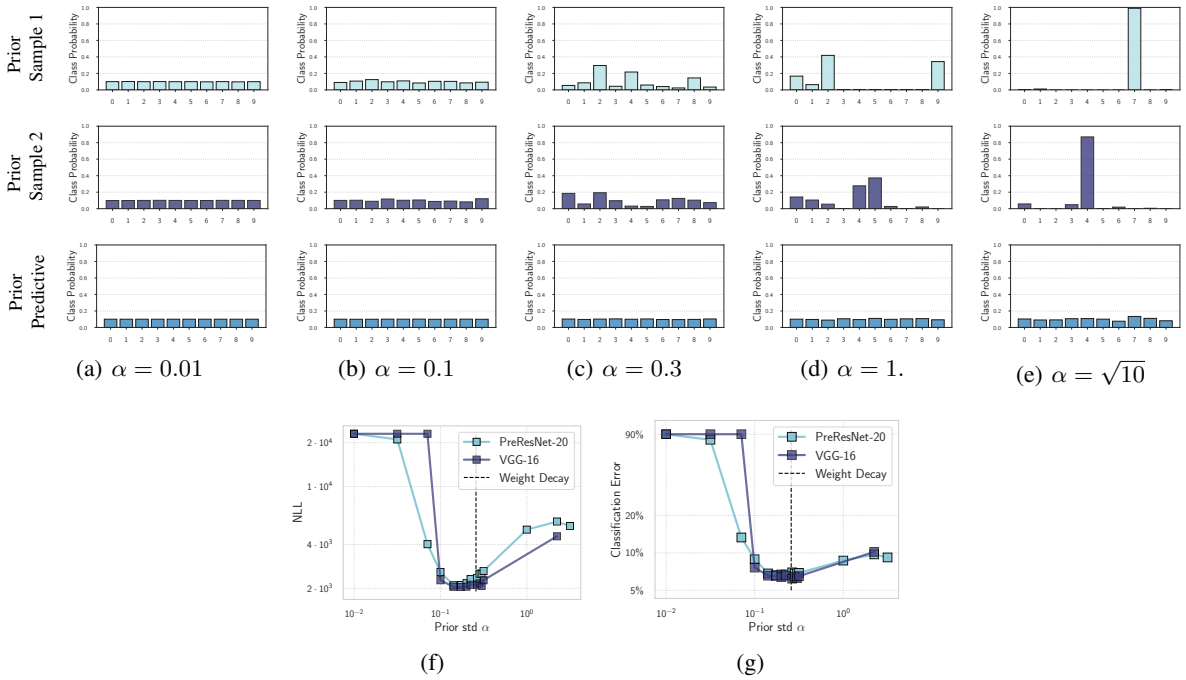
We consider (1) tempering under misspecification (Section F.1); (2) tempering in terms of overcounting data (Section F.2); (3) how tempering compares to changing the observation model (Section F.3); (4) the effect of the prior in relation to the experiments of Wenzel et al. (2020) (Section F.4); (5) the effect of approximate inference, including how tempering can help in efficiently estimating parameters even for the untempered posterior (Section F.5).

F.1. Tempering Helps with Misspecified Models

Many works explain how tempered posteriors help under model misspecification (e.g., de Heide et al., 2019; Grünwald et al., 2017; Barron & Cover, 1991; Walker & Hjort, 2001; Zhang, 2006; Bissiri et al., 2016; Grünwald, 2012). In fact, de Heide et al. (2019) and Grünwald et al. (2017) provide several simple examples where Bayesian inference fails to provide good convergence behaviour for untempered posteriors. While it is easier to show theoretical results for $T > 1$, several of these works also show that $T < 1$ can be preferred, even in well-specified settings, and indeed recommend learning T from data, for example by cross-validation (e.g., Grünwald, 2012; de Heide et al., 2019).

Are we in a misspecified setting for Bayesian neural networks? Of course. And it would be irrational to proceed as if it were otherwise. Every model is misspecified. In the context of Bayesian neural networks specifically, the mass of solutions expressed by the prior outside of the datasets we typically consider is likely much larger than desired for most applications. We can calibrate for this discrepancy through tempering. The resulting tempered posterior will be more in line with our beliefs than pretending the model is not misspecified and finding the untempered posterior.

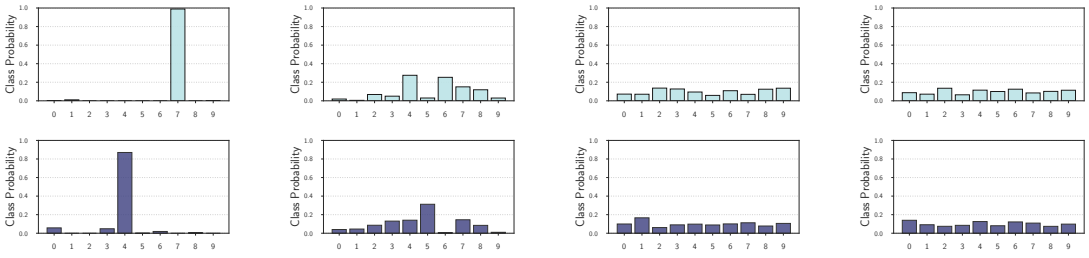
Non-parametric models, such as Gaussian processes, attempt to side-step model misspecification by growing the number of free parameters (information capacity) automatically with the amount of available data. In parametric



(a) $\alpha = 0.01$ (b) $\alpha = 0.1$ (c) $\alpha = 0.3$ (d) $\alpha = 1.$ (e) $\alpha = \sqrt{10}$

(f) (g)

Figure 10. **Effects of the prior variance α^2. (a)–(e)**: Average class probabilities over all of CIFAR-10 for two sample prior functions $p(f(x; w))$ (two top rows) and predictive distribution (average over 200 samples of weights, bottom row) for varying settings of α in $p(w) = \mathcal{N}(0, \alpha^2 I)$. **(f)**: NLL and **(g)** classification error of an ensemble of 20 SWAG samples on CIFAR-10 as a function of α using a Preactivation ResNet-20 and VGG-16. The NLL is high for overly small α and near-optimal in the range of $[0.1, 0.3]$. The NLL remains relatively low for vague priors corresponding to large values of α.



(a) Prior ($\alpha = \sqrt{10}$) (b) 10 datapoints (c) 100 datapoints (d) 1000 datapoints

Figure 11. **Adaptivity of posterior variance with data**. We sample two functions $f(x; w)$ from the distribution over functions induced by a distribution over weights, starting with the prior $p(w) = \mathcal{N}(0, 10 \cdot I)$, in combination with a PreResNet-20. We measure class probabilities averaged across the CIFAR-10 test set, as we vary the amount of available training data. Although the prior variance is too large, such that the softmax saturates for logits sampled from the prior, leading to one class being favoured, we see that the posterior quickly adapts to correct the scale of the logits in the presence of data. In Figure 10 we also show that the prior variance can easily be calibrated such that the prior predictive distribution, even before observing data, is high entropy.

models, we take much more of a manual guess about the model capacity. In the case of deep neural networks, this choice is not even close to a *best guess*; it was once the case that architectural design was a large component of works involving neural networks, but now it is more standard practice to choose an off-the-shelf architecture, without much consideration of model capacity. We do not believe that knowingly using a misspecified model to find a posterior is more reasonable (or Bayesian) than honestly reflecting the belief that the model is misspecified and then using a tempered posterior. For parametric models such as neural networks, it is to be expected that the capacity is particularly misspecified.

F.2. Overcounting Data with Cold Posteriors

The criticism of cold posteriors raised by (Wenzel et al., 2020) is largely based on the fact that decreasing temperature leads to overcounting data in the posterior distribution.

However, a similar argument can be made against marginal likelihood maximization (also known as *empirical Bayes* or *type 2 maximum likelihood*). Indeed, here, the prior will depend on the same data as the likelihood, which can lead to miscalibrated predictive distributions (Darnieder, 2011).

Nonetheless, empirical Bayes has been embraced and widely adopted in Bayesian machine learning (e.g., Bishop, 2006; Rasmussen & Williams, 2006; MacKay, 2003; Minka, 2001b), as embodying several Bayesian principles. Empirical Bayes has been particularly embraced in seminal work on Bayesian neural networks (e.g., MacKay, 1992; 1995), where it has been proposed as a principled approach to learning hyperparameters, such as the scale of the variance for the prior over weights, automatically embodying Occam's razor. While there is in this case some deviation from the fully Bayesian paradigm, the procedure, which depends on marginalization, is nonetheless clearly inspired by Bayesian thinking — and it is thus helpful to reflect this inspiration and provide understanding of how it works from a Bayesian perspective.

There is also work showing the marginal likelihood can lead to miscalibrated Bayes factors under model misspecification. Attempts to calibrate these factors (Xu et al., 2019), as part of the Bayesian paradigm, is highly reminiscent of work on safe Bayes.

F.3. Tempered Posterior or Different Likelihood?

The tempered posterior for one model is an untempered posterior using a different observation model. In other words, we can trivially change the likelihood function in our model so that the standard Bayesian posterior in the new model is equal to the posterior of temperature T in the original model. Indeed, consider the likelihood function

$$p_T(\mathcal{D}|w) \propto p(\mathcal{D}|w)^{1/T}\,, \tag{8}$$

where the posterior distribution for the model $\mathcal{M}_T$ with likelihood p_T is exactly the temperature T posterior for the model $\mathcal{M}$ with likelihood p.

The predictive distribution differs for the two models; even though the posteriors coincide, the likelihoods for a new datapoint y^* are different:

$$\int p(y^*|w)p(w)dw \neq \int p_T(y^*|w)p(w)dw\,. \tag{9}$$

As an example, consider a regression model $\mathcal{M}$ with a Gaussian likelihood function $y \sim \mathcal{N}(f, \sigma^2)$, where f is the output of the network and σ^2 is the noise variance. The predictive distributions for the two models $\mathcal{M}$ and $\mathcal{M}_T$ for a new input x will have different variance, but the same mean: $\mathbb{E}_w[f + \epsilon] = \mathbb{E}_w[f]$. Moreover, in this case the noise variance would typically be learned in either model $\mathcal{M}_T$ or $\mathcal{M}$.

A related construction is considered in Section 4.1 of Grünwald et al. (2017).

F.4. Effect of the Prior

While a somewhat misspecified prior will certainly interact with the utility of tempering, we do not believe the experiments in Wenzel et al. (2020) provide evidence that even the prior $p(w) = \mathcal{N}(0, I)$ is misspecified to any serious extent. For a relatively wide range of distributions over w, the functional form of the network $f(x; w)$ can produce a generally reasonable distribution over functions $p(f(x; w))$. In Figure 11, we reproduce the findings in Wenzel et al. (2020) that show that sample functions $p(f(x; w))$ corresponding to the prior $p(w) = \mathcal{N}(0, 10 \cdot I)$ strongly favour a single class over the dataset. While this behaviour appears superficially dramatic, we note it is simply an artifact of having a miscalibrated signal variance. A miscalibrated signal variance interacts with a quickly saturating soft-max link function to provide a seemingly dramatic preference to a given class. If we instead use $p(w) = \mathcal{N}(0, \alpha^2 I)$, for quite a range of α, then sample functions provide reasonably high entropy across labels, as in Figure 10. The value of α can be easily determined through cross-validation, as in Figure 10, or specified as a standard value used for L_2 regularization ($\alpha = 0.24$ in this case).

However, even with the inappropriate prior scale, we see in the bottom row of panels (a)–(e) of Figure 10 that the unconditional predictive distribution *is* completely reasonable. Moreover, the prior variance represents a *soft* prior bias, and will quickly update in the presence of data. In Figure 11 we show the posterior samples after observing $10, 100$, and 1000 data points.

Other aspects of the prior, outside of the prior signal variance, will have a much greater effect on the inductive biases of the model. For example, the induced covariance function $\text{cov}(f(x_i, w), f(x_i, w))$ reflects the induced similarity metric over data instances; through the covariance function we can answer, for instance, whether the model believes a priori that a translated image is similar to the original. Unlike the signal variance of the prior, the prior covariance function will continue to have a significant effect on posterior inference for even very large datasets, and strongly reflects the structural properties of the neural network. We explore these structures of the prior in Figure 9.

F.5. The Effect of Inexact Inference

We have to keep in mind what we ultimately use posterior samples to compute. Ultimately, we wish to estimate the predictive distribution given by the integral in Equation (1). With a finite number of samples, the tempered posterior could be used to provide a better approximation to the expectation of the predictive distribution associated with untempered posterior.

Consider a simple example, where we wish to estimate the mean of a high-dimensional Gaussian distribution $\mathcal{N}(0, I)$. Suppose we use J independent samples. The mean of these samples is also Gaussian distributed, $\mu \sim \mathcal{N}(0, \frac{1}{J}I)$. In Bayesian deep learning, the dimension d is typically on the order 10^7, and J would be on the order of 10. The norm of μ would be highly concentrated around $\frac{\sqrt{10^7}}{\sqrt{10}} = 1000$. In this case, sampling from a tempered posterior with $T < 1$ would lead to a better approximation of the Bayesian model average associated with an untempered posterior.

Furthermore, no current sampling procedure will be providing samples that are close to independent samples from the true posterior of a Bayesian neural network. The posterior landscape is far too multimodal and complex for there to be any reasonable coverage. The approximations we have are practically useful, and often preferable to conventional training, but we cannot realistically proceed with analysis assuming that we have obtained true samples from a posterior. While we would expect that some value of $T \neq 1$ would be preferred for any finite dataset in practice, it is conceivable that some of the results in Wenzel et al. (2020) may be affected by the specifics of the approximate inference technique being used.

We should be wary not to view Bayesian model averaging purely through the prism of simple Monte Carlo, as advised in Section 3.2. Given a finite computational budget, our goal in effectively approximating a Bayesian model average is *not* equivalent to obtaining good samples from the posterior.

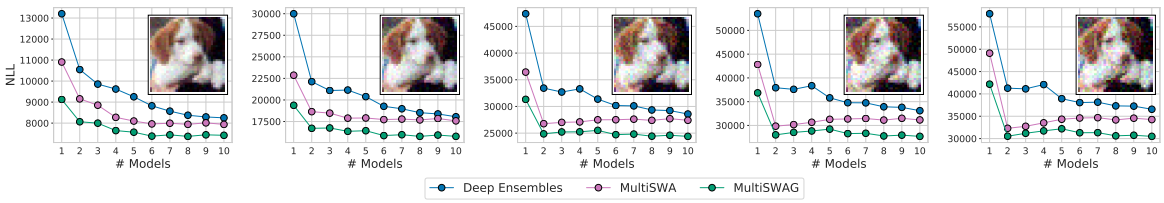
G. Prior Correlation Structure under Perturbations

In this section we explore the prior correlations between the logits on different pairs of datapoints induced by a spherical Gaussian prior on the weights of a PreResNet-20. We sample a 100 random images from CIFAR-10 (10 from each class) and apply 17 different perturbations introduced by Hendrycks & Dietterich (2019) at 5 different levels of intensity. We then compute correlations between the logits $f(x, w)$ for the original image x and $f(\tilde{x}, w)$ for the corrupted image $\tilde{x}$, as we sample the weights of the network from the prior $w \sim \mathcal{N}(0, I)$.

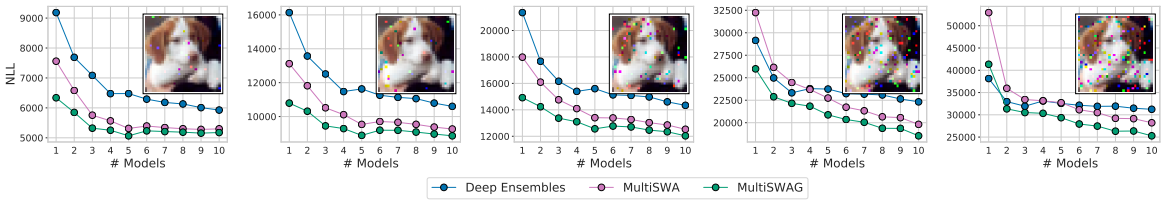
In Figure 17 we show how the correlations decay with perturbation intensity. For reference we also show how the correlations decay for a linear model and for an RBF kernel. For the RBF kernel we set the lengthscale so that the average correlations on the uncorrupted datapoints match those of a PreResNet-20. Further experimental details can be found in Appendix D.3.

For all types of corruptions except *saturate*, *snow*, *fog* and *brightness* the PreResNet logits decay slower compared to the RBF kernel and linear model. It appears that the prior samples are sensitive to corruptions that alter the brightness or more generally the colours in the image. For many types of corruptions (such as e.g. *Gaussian Noise*) the prior correlations for PreResNet are close to 1 for all levels of corruption.

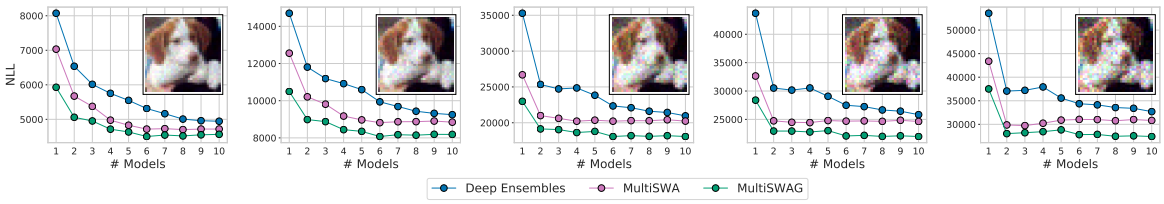
Overall, these results indicate that the prior over *functions* induced by a vague prior over parameters w in combination with a PreResNet has useful equivariance properties: before seeing data, the model treats images of the same class as highly correlated, even after an image has undergone significant perturbations representative of perturbations we often see in the real world. These types of symmetries are a large part of what makes a neural networks are powerful model class for high dimensional natural signals.



(a) Gaussian Noise

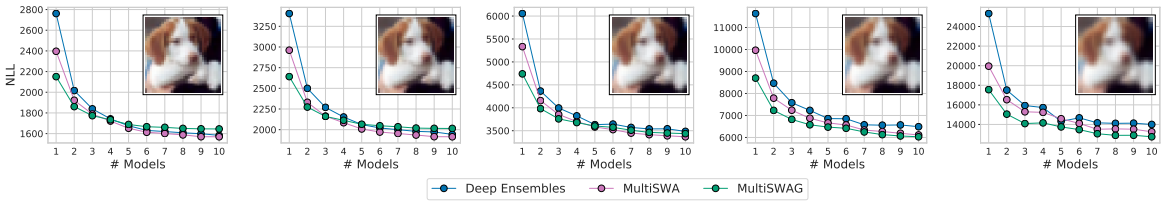


(b) Impulse Noise

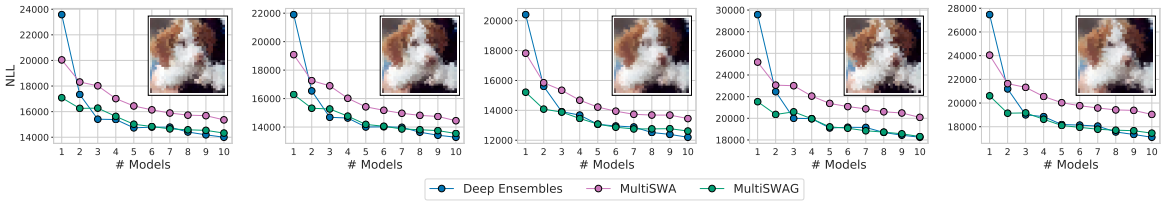


(c) Shot Noise

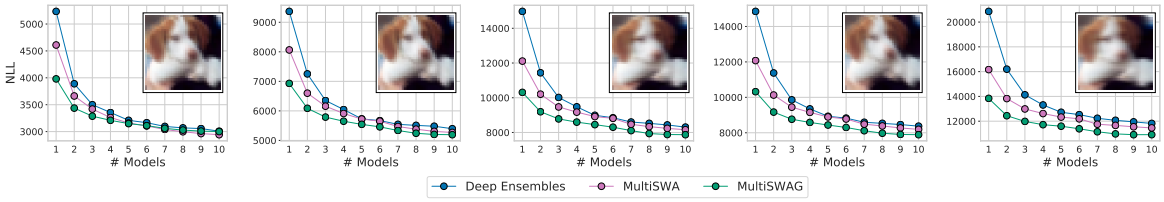
Figure 12. **Noise Corruptions.** Negative log likelihood on CIFAR-10 with a PreResNet-20 for Deep Ensembles, MultiSWAG and MultiSWA as a function of the number of independently trained models for different types of corruption and corruption intensity (increasing from left to right).



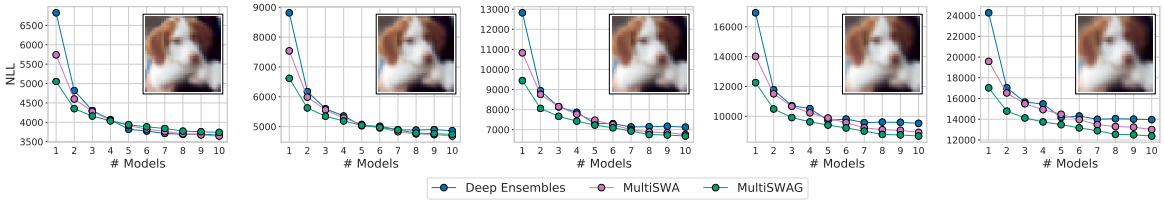
(a) Defocus Blur



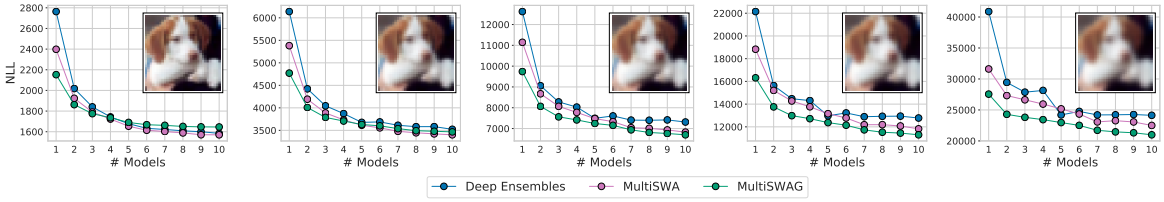
(b) Glass Blur



(c) Motion Blur

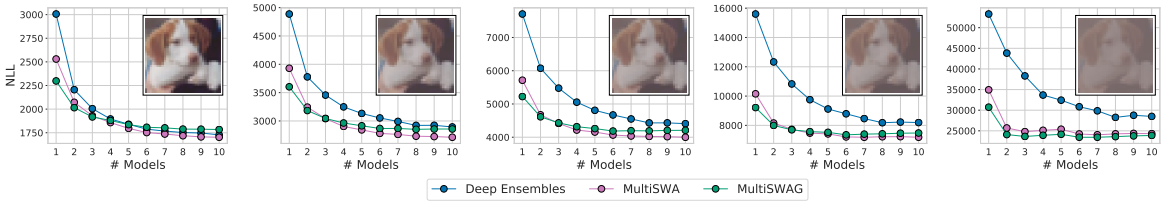


(d) Zoom Blur

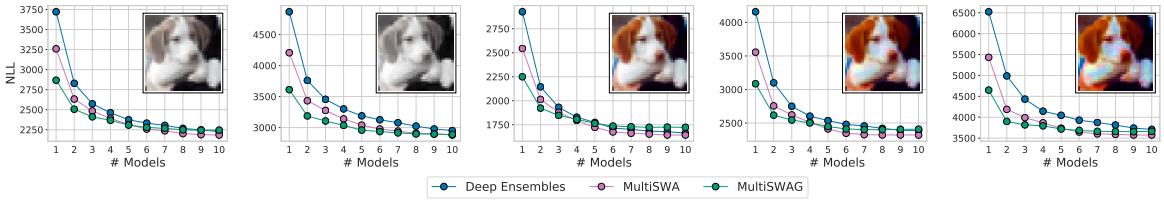


(e) Gaussian Blur

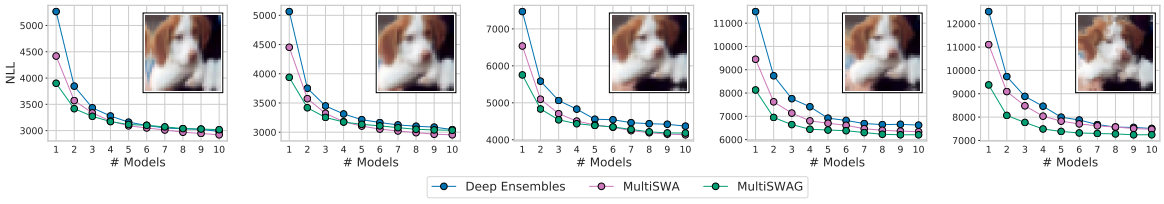
Figure 13. **Blur Corruptions.** Negative log likelihood on CIFAR-10 with a PreResNet-20 for Deep Ensembles, MultiSWAG and MultiSWA as a function of the number of independently trained models for different types of corruption and corruption intensity (increasing from left to right).



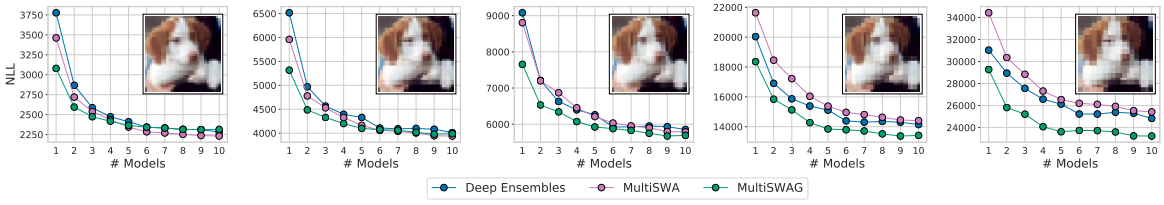
(a) Contrast



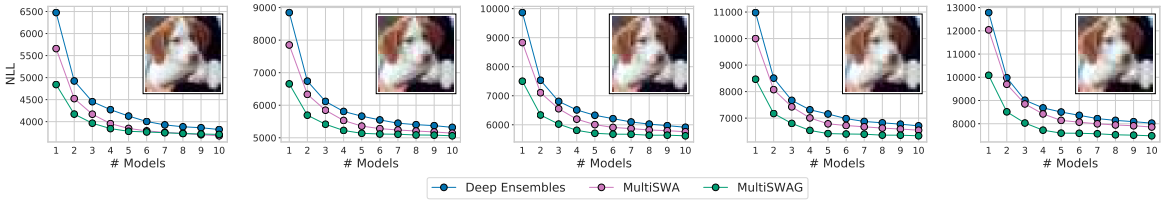
(b) Saturate



(c) Elastic Transform

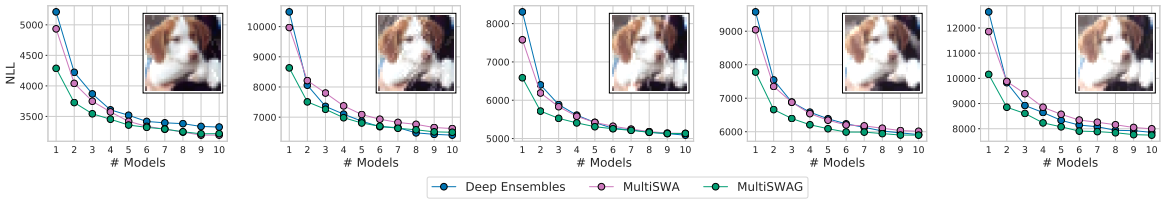


(d) Pixelate

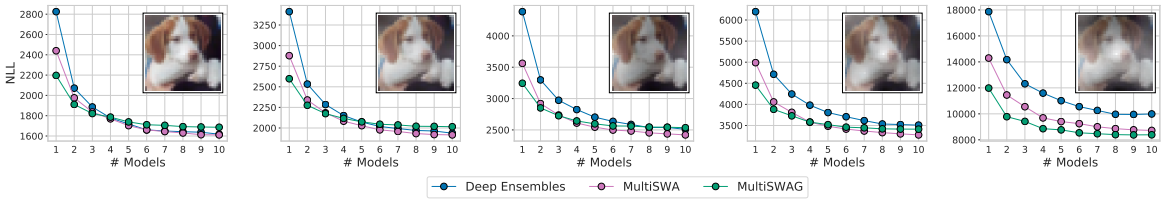


(e) JPEG Compression

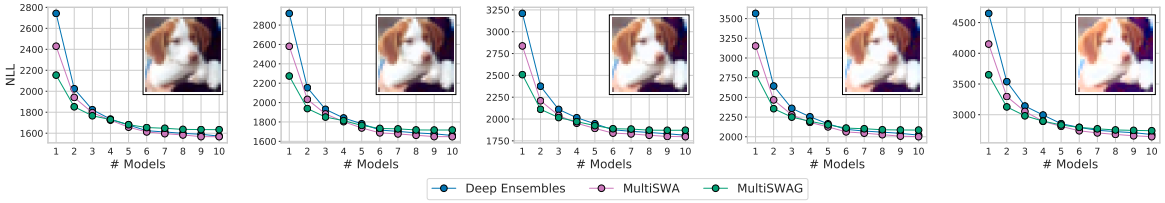
Figure 14. **Digital Corruptions.** Negative log likelihood on CIFAR-10 with a PreResNet-20 for Deep Ensembles, MultiSWAG and MultiSWA as a function of the number of independently trained models for different types of corruption and corruption intensity (increasing from left to right).



(a) Snow



(b) Fog



(c) Brightness

Figure 15. **Weather Corruptions.** Negative log likelihood on CIFAR-10 with a PreResNet-20 for Deep Ensembles, MultiSWAG and MultiSWA as a function of the number of independently trained models for different types of corruption and corruption intensity (increasing from left to right).

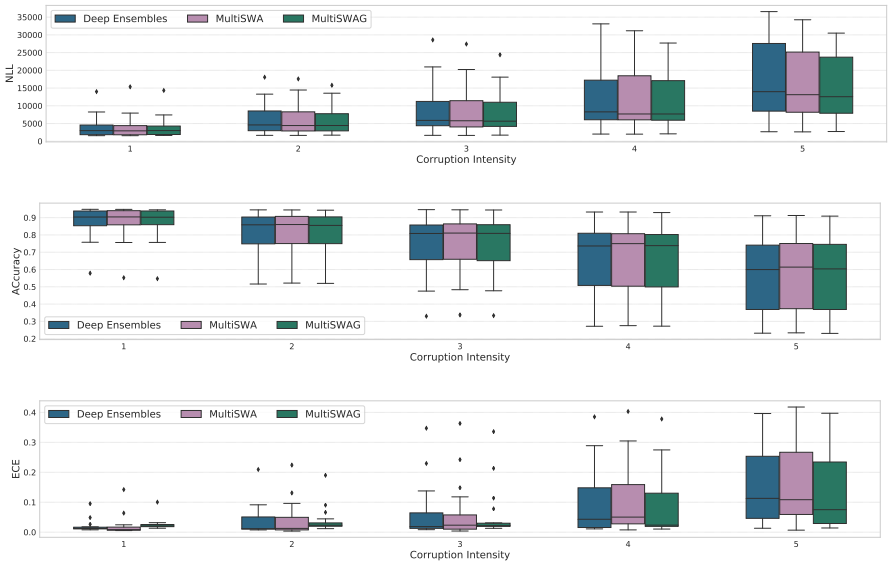


Figure 16. Negative log likelihood, accuracy and expected calibration error distribution on CIFAR-10 with a PreResNet-20 for Deep Ensembles, MultiSWAG and MultiSWA as a function of the corruption intensity. Following Ovadia et al. (2019) we summarize the results for different types of corruption with a boxplot. For each method, we use 10 independently trained models, and for MultiSWAG we sample 20 networks from each model. As in Figures 5, 11-14, there are substantial differences between these three methods, which are hard to see due to the vertical scale on this plot. MultiSWAG particularly outperforms Deep Ensembles and MultiSWA in terms of NLL and ECE for higher corruption intensities.

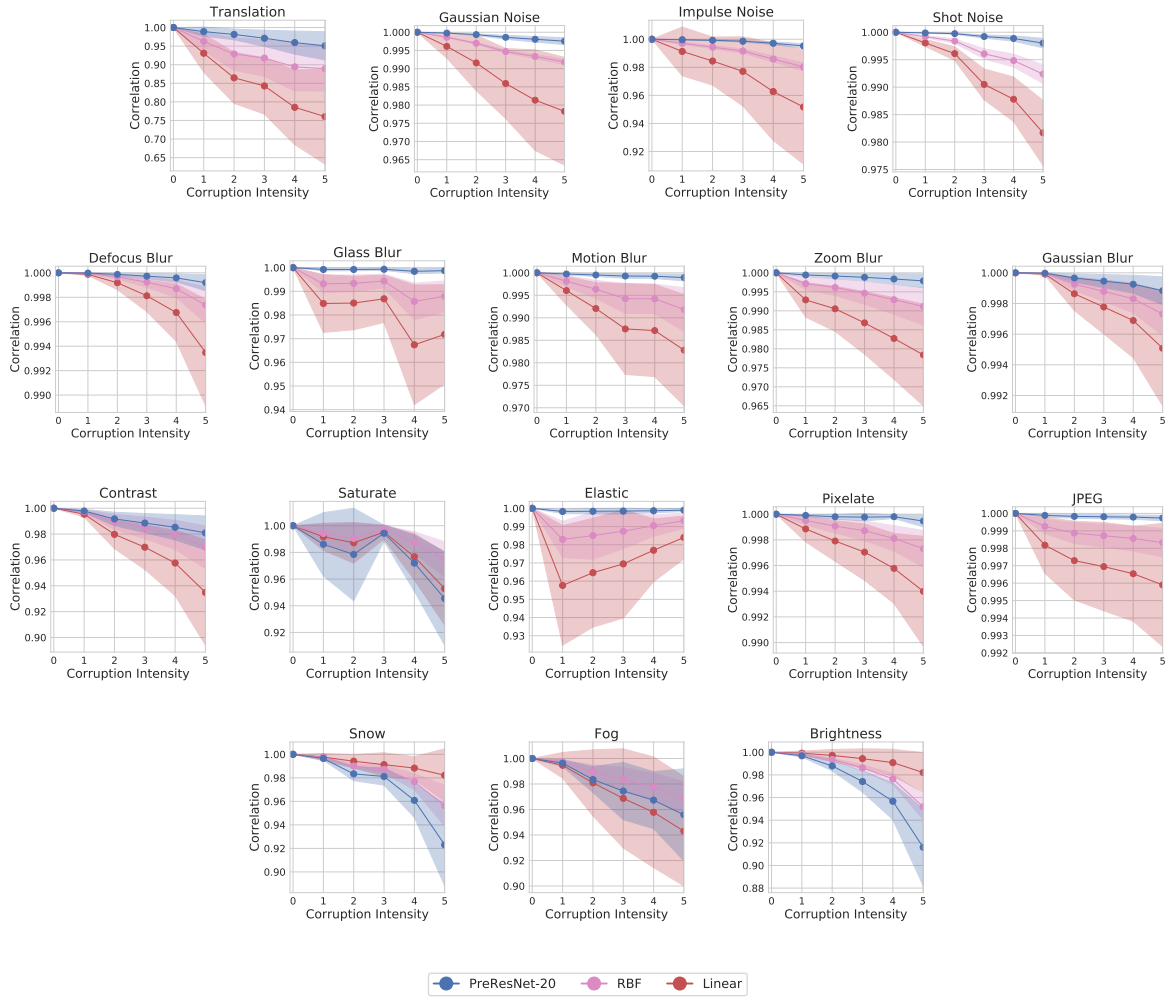


Figure 17. **Prior correlations under corruption.** Prior correlations between predictions (logits) for PreResNet-20, Linear Model and RBF kernel on original and corrupted images as a function of corruption intensity for different types of corruptions. The lengthscale of the RBF kernell is calibrated to produce similar correlations to PreResNet on uncorrupted datapoints. We report the mean correlation values over 100 different images and show the 1σ error bars with shaded regions. For all corruptions except Snow, Saturate, Fog and Brightness the correlations decay slower for PreResNet compared to baselines.