Pre-Train Your Loss: Easy Bayesian Transfer Learning with Informative Priors

Ravid Shwartz-Ziv*
New York University
ravid.shwartz.ziv@nyu.edu

Micah Goldblum*
New York University
goldblum@nyu.edu

Hossein Souri
Johns Hopkins University

Sanyam Kapoor
New York University

Chen Zhu
University of Maryland

Yann LeCun
New York University
Meta AI Research

Andrew Gordon Wilson
New York University
andrewgw@cims.nyu.edu

Abstract

Deep learning is increasingly moving towards a transfer learning paradigm whereby large foundation models are fine-tuned on downstream tasks, starting from an initialization learned on the source task. But an initialization contains relatively little information about the source task, and does not reflect the belief that our knowledge of the source task should affect the locations and shape of optima on the downstream task. Instead, we show that we can learn highly informative posteriors from the source task, through supervised or self-supervised approaches, which then serve as the basis for priors that modify the whole loss surface on the downstream task. This simple modular approach enables significant performance gains and more data-efficient learning on a variety of downstream classification and segmentation tasks, serving as a drop-in replacement for standard pre-training strategies. These highly informative priors also can be saved for future use, similar to pre-trained weights, and stand in contrast to the zero-mean isotropic uninformative priors that are typically used in Bayesian deep learning.

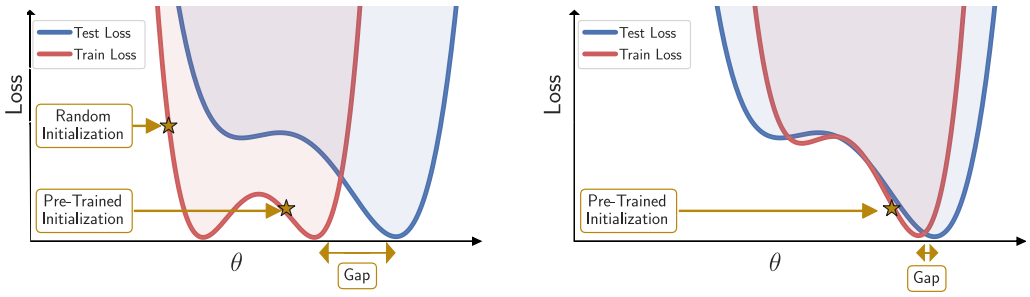
1 Introduction

The ability to transfer what is learned from one task to another — learning to ride a bicycle to then ride a unicycle — has historically set apart biological intelligence from machine learning approaches. However, *transfer learning* is quickly becoming mainstream practice in deep learning. Typically, large "foundation models" are pre-trained on massive volumes of source data, and then the learned parameter vector is used as an initialization for training in a downstream task. While this approach has had great empirical success, reliance on an initialization is a very limited way to perform transfer learning. If we are doing a good job of optimization, then our final solution should be independent of initialization, barring local minima with identical training loss. Moreover, our knowledge of the source task should affect the locations and shapes of optima on the downstream knowledge.

We propose to instead use a re-scaled Bayesian parameter posterior from the first task as a *pre-trained* prior for the downstream task. Since the negative log posterior on the downstream task is our loss function, this procedure has the effect of reshaping our training objective on the downstream task,

*Authors contributed equally.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).



(a) Standard Transfer Learning

(b) Transfer Learning with Learned Priors

Figure 1: **Learned priors reshape the loss surface.** Red curves represent training loss, and blue curves represent the true population loss. The typical transfer learning approach avoids poor-generalizing minima by improving initializations (left). Learned priors further improve the alignment between training and test loss, improving accuracy (right). The gap between learned and optimal parameters is denoted by a double-sided gold arrow.

to reflect more nuanced information that we have learned from the source task, as in Figure 1. The posterior on the source task is re-scaled before use as a prior on the downstream task to reflect the belief that the source and downstream tasks are drawn from different distributions.

With our now highly informative prior for the downstream task, we can then proceed with optimization, or perform full Bayesian model averaging with the posterior on the downstream task. We find both procedures profoundly improve upon standard deep transfer learning, making use of a simple pipeline that involves easy-to-use pre-existing components. Indeed, the simplicity of the approach, combined with its promising empirical performance, is one of its greatest features — enabling its use as a drop-in replacement for standard approaches for deep transfer learning. The pre-training posterior can be saved as a prior for a wide array of downstream tasks, similar to pre-trained weights.

Despite the simplicity and effectiveness of this approach, Bayesian neural networks are almost always used with uninformative isotropic zero-mean priors [50, 25, 15], and have not harnessed recent advances in self-supervised learning. Furthermore, as we will show, the details of how we perform Bayesian transfer learning with modern neural networks are crucial to practical success. For example, it is important that the prior on the source task be represented with an informative covariance matrix, indicating which directions in parameter space we expect to provide good solutions on the source task, which significantly outperforms isotropic or diagonal counterparts. We also provide several key conceptual findings — for example, that the informative priors provide an inductive bias that enables more data efficient fine-tuning than a pre-trained initialization, and that pre-trained priors based on self-supervised learning are more transferable than supervised pre-trained priors. Finally, we observe through extensive experiments across both image classification and semantic segmentation on multiple neural architectures, and on both supervised and self-supervised (SimCLR [6]) pre-training loss functions, that Bayesian inference is particularly advantageous in the transfer learning setting.

We emphasize that the proposed approach outperforms standard transfer learning, without requiring any expert intervention or knowledge of Bayesian methods. One only needs to tune a single additional hyperparameter using standard cross-validation, which incurs minimal computational overhead. We release `PyTorch` pre-trained priors and code for learning and using priors for downstream inference: `https://github.com/hsouri/BayesianTransferLearning`.

2 Background

Bayesian Neural Networks. Consider a neural network f with weights w and a training dataset $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}$ of independently drawn samples. In standard non-Bayesian neural network training, we minimize the negative log posterior, $-\log p(w|\mathcal{D}, f) \propto -\log p(\mathcal{D}|w, f) - \log p(w|f)$, where the likelihood $p(\mathcal{D}|w, f)$ denotes the probability that a model with weights w would generate the observed labels $\{y^{(i)}\}$ (e.g., cross-entropy). The log prior $\log p(w|f)$ often takes the form of weight decay, corresponding to a zero-mean isotropic Gaussian prior. In Bayesian modeling,

we instead make predictions with a *Bayesian Model Average* (BMA) of all models weighted by their posterior probabilities: $p(y|x, \mathcal{D}, f) = \int p(y|x, w, f)p(w|\mathcal{D}, f)dw$. There are many ways to approximate this integral, such as MCMC, or variational approaches, which take a finite number of approximate samples w_j from the parameter posterior to form the simple Monte Carlo estimate: $p(y|x, \mathcal{D}, f) \approx \frac{1}{J}\sum_j p(y|x, w_j, f)$. Almost always, one uses zero-mean isotropic priors [50]. There are also specialized priors, including heavy-tailed priors [15, 25], noise-contrastive priors for high uncertainty under distribution shift [17], and input-dependent priors for domain generalization [24].

Transfer Learning. In *transfer learning*, we wish to recycle the representation learned on one task to improve performance on another. Transfer learning is now widely applied in deep learning, and forms the basis for *foundation models* [1, 12, 8, 43, 4, 20, 10], which are exceptionally large neural networks pre-trained on massive volumes of data, and then fine-tuned on a downstream task. Recent work has found self-supervised pre-training can transfer better than supervised pre-training [19], in line with our finding in Section 4 that self-supervised pre-trained priors transfer better.

In *continual learning*, we wish to learn tasks sequentially without forgetting what has been learned previously. *Elastic Weight Consolidation* (EWC) prevents forgetting by imposing a penalty, the diagonal of a Fisher information matrix computed on previous tasks, for adapting to a new task [29]. We can contextualize the EWC penalty within Bayesian transfer learning as a negative log Gaussian prior with diagonal covariance used for maximum a posteriori (MAP) estimation on sequential tasks, such as digit classification or RL [29]. In our experiments, we find that MAP estimation, diagonal covariance, and supervised pre-training are all suboptimal in the transfer learning setting.

Leveraging Auxiliary Data and Knowledge Transfer in Bayesian Modeling. A number of works outside of deep learning have considered knowledge transfer in Bayesian modeling, especially in settings such as domain adaptation or homogeneous transfer learning in which source and target tasks contain similar feature and label spaces but differ in their marginal distributions. For example, Xuan et al. [51] considers Bayesian transfer learning methods for probabilistic graphical models, and Karbalayghareh et al. [28] develop a theoretical framework for understanding optimal Bayes classifiers given prior knowledge from other domains. Other works on Bayesian transfer learning learn a Dirichlet prior over naive Bayes classifiers [44]. Bayesian optimization methods can also find transferable hyperparameter settings [40] or select data which will yield transferable models on shifted domains [45]. Schnaus [47] uses the Laplace approximation to learn posteriors with Kronecker-factored covariance and then adjusts the posteriors by optimizing PAC-Bayes generalization bounds to create priors for downstream applications.

Bayesian tools have additionally been used for leveraging auxiliary data or multiple domains in deep learning. Chandra and Kapoor [2] learn from multiple domains simultaneously using a round-robin task sampling procedure and a single-layer neural network. Bayesian methods for continual learning update the posterior to accommodate new tasks without forgetting how to perform previous ones [35, 48, 13, 27, 46], or develop kernels based on neural networks trained on previous tasks for Gaussian process inference [33, 37]. Semi-supervised algorithms can incorporate unlabeled data into the training pipelines of BNNs using biologically plausible Bayesian Confidence Propagation Neural Networks (BCPNN) which model the cortex [41], by perturbing weights and using consistency regularization [11], or via semi-supervised deep kernel learning [26]. Gao et al. [16] also show how to harness unlabeled data to learn reference priors, uninformative priors which depend on the amount of training data and allow labels to most efficiently inform inference. Deep kernel learning has also been used for Bayesian meta-learning in few-shot classification and regression [39].

In contrast to these works, we do not perform multi-task learning nor is our goal to harness auxiliary unlabeled downstream data; we instead approach transfer learning in which we leverage pre-training data for expressive priors that maximize performance on a single downstream task.

3 Bayesian Transfer Learning with Pre-Trained Priors

In order to transfer knowledge acquired through pre-training to downstream tasks, we adopt a pipeline with three simple components, composed of easy-to-use existing tools:

1. First, we fit a probability distribution with closed-form density to the posterior over feature extractor parameters using a pre-trained checkpoint and SWAG [34] (Section 3.1).

2. Second, we re-scale this distribution, viewed as a prior for new tasks, to reflect the mismatch between the pre-training and downstream tasks and to add coverage to parameter settings which might be consistent with the downstream, but not pre-training, task. To this end, we tune a single scalar coefficient on held out validation data (Section 3.2).

   Finally, we use this re-scaled prior, along with a zero-mean isotropic Gaussian prior over added parameters (e.g. classification head), to form a posterior on the downstream task. We then either optimize the posterior, or use it to perform full Bayesian inference with SGLD and SGHMC samplers [42, 5] (Section 3.3)

3. Finally, we plug the re-scaled prior into a Bayesian inference algorithm, along with a zero-mean isotropic Gaussian prior over added parameters (e.g. classification head) to form a posterior on the downstream task. We then either optimize the posterior, or use it to perform full Bayesian inference with SGLD and SGHMC samplers [49, 5] (Section 3.3).

We illustrate this pipeline in Figure 5 in the Appendix. The simplicity and modularity of this framework are key strengths: by carefully combining easy-to-use existing components, we will see in Section 4 that we can straightforwardly improve the default approaches to deep transfer learning. The potential for impact is significant: we can use this pipeline as a drop-in replacement for standard procedures used in deploying foundation models. At the same time, there is significant novelty: while Bayesian neural networks are typically used with simple zero-mean isotropic priors, we leverage the significant developments in self-supervised learning to produce highly informative priors. Moreover, in the following subsections (3.1, 3.2, and 3.3) we will gain a nuanced understanding of each of these three components and the key considerations for practical success. We discuss computational considerations in Section 3.4.

For experiments in this section, we use a prior learned over the parameters of an ImageNet pre-trained SimCLR ResNet-50 feature extractor [9, 6, 18], and we choose CIFAR-10 and CIFAR-100 for downstream tasks [30]. We perform Bayesian inference with stochastic gradient Hamiltonian Monte Carlo (SGHMC) [5]. The experiments in this section are primarily intended to gain conceptual insights into each step of our approach. In Section 4, we present our main empirical evaluations.

3.1 Learning Transferable Priors

We begin by building a probability distribution over the parameters of a feature extractor which represents knowledge we acquire from pre-training. To this end, we fit the distribution to the Bayesian posterior, or regularized loss function, on the pre-training task. This pre-training posterior will become the prior for downstream tasks and can be saved, or publicly released, for future use, similar to pre-trained weights. This stage requires two major design choices: a pre-training task and an algorithm for constructing the probability distribution.

Our downstream inference procedures require that our prior is represented as a closed-form density function. Thus, we must use a method that can provide a closed-form posterior approximation for the source task, which can be then re-scaled and used as a prior in the downstream task. We opt for SWA-Gaussian (SWAG) [34] due to its simplicity, scalability, popularity, and non-diagonal covariance. Other procedures that provide closed-form posterior approximations, such as the Laplace approximation or variational methods, could also be applied, though as we will see, a non-diagonal covariance is particularly important to the success of this approach.

SWAG starts from a pre-trained model, and runs a small number M of fine-tuning epochs with a modified learning rate schedule [34]. The SWAG approximate posterior distribution is given by $\mathcal{N}(\overline{w}, \frac{1}{2}\Sigma_{\text{diag}} + \frac{1}{2}\Sigma_{\text{low-rank}})$, where $\overline{w} = \frac{1}{M}\sum_{t=1}^{M} w_t$ is the SWA [23] solution, $\Sigma_{\text{diag}} = \frac{1}{L-1}\sum_{t=M-L+1}^{M} \text{diag}(w_t - \overline{w})$, and $\Sigma_{\text{low-rank}} = \frac{1}{L-1}\sum_{t=M-L+1}^{M} (w_t - \overline{w})(w_t - \overline{w})^\top$, where L is a hyperparameter controlling the rank of the low-rank component of the covariance matrix. After obtaining a closed-form distribution using SWAG, we remove the head from on top of the feature extractor, for example, a linear classification module, and we consider only the distribution's restriction to the parameters of the feature extractor. New layers that are added for downstream tasks receive a non-learned prior over their parameters.

To highlight the versatility of Bayesian transfer learning, we will focus both on supervised image classification and also on self-supervised learning pre-training tasks and leverage existing `torchvision` and SimCLR [6] checkpoints. For both `torchvision` and SimCLR models, we learn the prior

using the associated loss function — cross-entropy and InfoNCE, respectively, regularized by weight decay. In both settings, the regularized loss function can be represented as the sum of a negative log-likelihood and the negative log Gaussian density (weight decay).

While standard SGD-based transfer learning uses a learned parameter vector, our Bayesian transfer learning approach uses a covariance matrix over feature extractor parameters which contains information about the pre-training loss surface geometry.

Can we really "learn" a prior? A data-dependent prior may sound odd, because a prior reflects our beliefs "before we see the data". However, it is entirely principled to learn a prior, as long as it is not learned using exactly the same labeled data we use in our downstream likelihood for the downstream task. Indeed, any informative prior is based on data that has shaped our beliefs.

Alignment of Pre-Training and Downstream Loss Geometry. If the covariance matrix of our learned prior is to be more effective than an isotropic one, then it must not favour poorly generalizing directions for the downstream task. Thus, we compare the alignment of the learned covariance matrix with a downstream task's test loss.

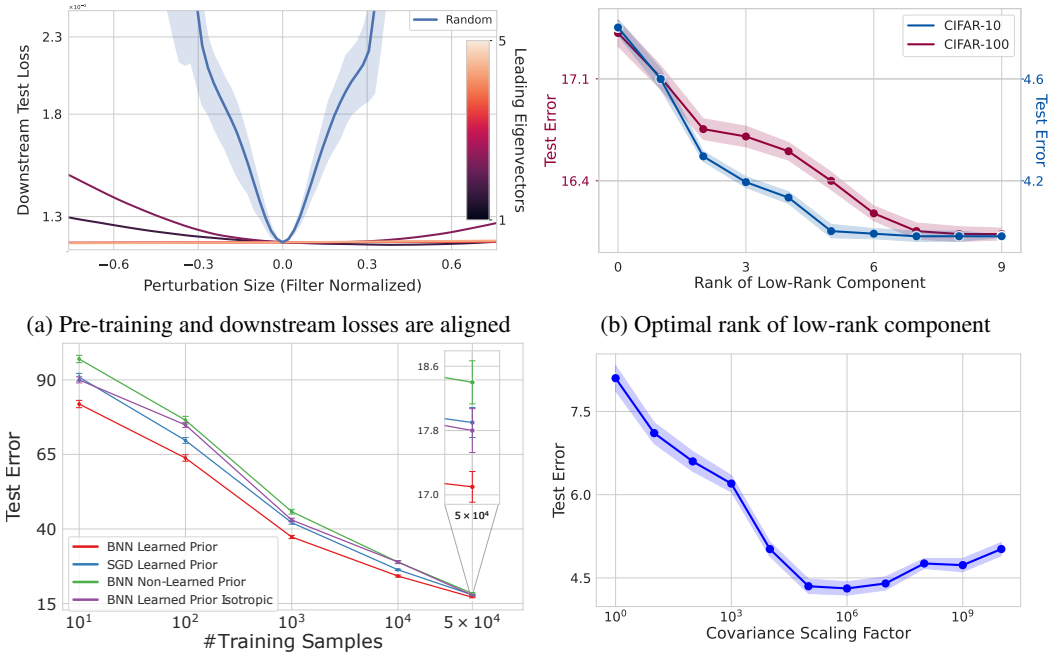
We begin by computing the top 5 leading singular vectors of a SWAG learned covariance matrix over parameters of the SimCLR ImageNet-trained ResNet-50 feature extractor. We then train a linear classifier head on CIFAR-10 training data on top of the fixed pre-trained feature extractor. Starting at the learned parameter vector, we perturb the feature extractor parameters in the direction of the singular vectors (fixing fully-connected classifier parameters), and measure the increase in test loss. We compare to the test loss when instead perturbing by each of 10 random vectors. All perturbation distances are filter normalized (as in [31, 21]), to account for invariance with respect to filter-wise parameter rescaling. In Figure 2a, we see the CIFAR-10 test loss is far flatter in the directions of leading eigenvectors of the pre-trained covariance than in a random direction, indicating the learned prior indeed promotes directions consistent with the downstream task.

Learned Covariance Outperforms Only a Learned Mean. After verifying that our pre-trained priors do in fact identify flat directions of the pre-training loss which are aligned with the downstream loss, we directly compare the performance benefits of our learned covariance over an isotropic covariance with a learned mean. To this end, we swap out our learned prior's covariance with an isotropic version αI, where α is tuned on a held out validation set. Figure 2c shows that a learned covariance consistently outperforms its isotropic counterpart, indicating that the shape of the pre-training loss surface's basin is informative for downstream tasks. The x-axis here denotes the number of training samples used for fine-tuning on the downstream task. We also see Bayesian model averaging provides further performance gains, which we discuss further in Section 3.3. We now dissect just how important the low-rank component is.

How is the Prior Best Structured? In the context of continual learning, elastic weight consolidation (EWC) [29] uses purely diagonal covariance priors to help prevent forgetting in continual learning. In this paper, we are interested in transfer learning, and wish to understand the benefits of a low-rank component for capturing particularly important directions in the pre-training loss surface for transferring to downstream tasks. Omitting the low-rank component slightly reduces the memory footprint, but loses potentially important information about the shape of the pre-training posterior mode. In practice, the rank of the matrix is determined by the number of samples collected when running SWAG. In Figure 2b, we present the performance of a prior learned via SWAG on a SimCLR ResNet-50 feature extractor for transfer learning to CIFAR-10 and CIFAR-100. As the rank of the low-rank component increases from zero (diagonal covariance), we see the performance improves dramatically until it saturates, indicating that only a small number of dominant directions are important for the transferability of the prior. Note that performance saturation occurs later for CIFAR-100 than CIFAR-10 — likely due to the higher complexity of CIFAR-100 compared to CIFAR-10, which has $10\times$ fewer classes.

3.2 Rescaling the Prior

In *incremental learning*, it is common to acquire some data, form a posterior, and then use this posterior as our new prior in acquiring future data. This procedure is equivalent to forming the posterior from all of the data at once, assuming all data are drawn from the same distribution with the same likelihood. However, in transfer learning, we assume the data from the source and downstream task are drawn from *different* but related distributions. Thus we do not want to directly re-use a



(a) Pre-training and downstream losses are aligned

(b) Optimal rank of low-rank component

(c) Importance of a learned covariance

(d) Optimal prior scaling factor

Figure 2: **Learning Transferable Priors.** Error bars corresponds to standard error over 5 trials. All experiments performed with ResNet-50 backbone pre-trained via SimCLR on ImageNet. **(a)** Test loss (CIFAR-10) is less sensitive to perturbations in the directions of leading singular vectors of a learned prior's covariance compared to random directions. Estimated over 10 random directions. **(b)** A non-diagonal covariance is important, though performance on downstream tasks plateaus at a relatively low rank. CIFAR-100 benefits from a higher rank covariance than CIFAR-10, due to its relative complexity. **(c)** A non-diagonal covariance outperforms an isotropic covariance with learned mean on CIFAR-100. Bayesian inference provides further performance gains. **(d)** We must re-scale the posterior from the source task, since data from each task is drawn from a different distribution. If λ is too small, then we assume too much similarity between tasks. If λ is too large, we lose useful information from the source task. Downstream inference is performed on CIFAR-10.

posterior from the source as a prior for the downstream task, without modification. For example, as we acquire more data from the source task, our posterior will become increasingly concentrated. This concentration is an issue, as certainty with respect to the ideal parameters for the pre-training task does not imply certainty on the ideal parameters for the downstream task.

To address this consideration, we rescale the learned Gaussian prior by multiplying its covariance matrix by a scalar value. We select the highest performing scalar value across a grid on a holdout set from the downstream task. If we do not scale the covariance enough, our prior will become concentrated around parameters which are inconsistent with the downstream task, and if we scale the covariance too much, our prior will be diffuse and assign too much mass to regions of parameter space which are again inconsistent with the downstream task.

We now put this intuition to the test. We fit the SimCLR pre-training loss using a Gaussian prior with mean μ and covariance matrix Σ. Given our uncertainty regarding the strength of the relationship between the pre-training and downstream tasks, it is unclear if our learned prior would lead to over-confidence on the downstream task. To rectify this problem, we instead assign prior covariance matrix $\lambda\Sigma$ with $\lambda \geq 1$. Figure 2d shows the accuracy of our method on CIFAR-10 as a function of λ. As expected, we see that we need to make the prior more diffuse if we are to optimize performance. In fact, prior rescaling can be the difference between strong and poor performance. We also see there is an optimal scaling factor where the prior is neither overly diffuse nor concentrated around a solution which is inconsistent with the downstream task. Additionally, small scaling factors constrain ourselves to poor parameters, hurting performance much more than large scaling factors, which simply inducing a near-uniform prior, negating the benefits of transfer learning.

3.3 Bayesian Inference

After learning a prior and re-scaling it, we finally must draw samples from the downstream task's posterior over the parameters of the entire model, including additional modules, such as classification heads, which were added after pre-training specifically for a particular downstream task. We use a zero-mean isotropic Gaussian prior over these additional parameters, with a scaling that is again tuned on held-out training data from the downstream task. Since we obtain a closed-form prior, the re-scaled distributions we learn are compatible with a wide variety of Bayesian inference algorithms. In our experiments, we choose stochastic gradient Hamiltonian Monte Carlo (SGHMC) [5] and Stochastic Gradient Langevin Dynamics (SGLD) [49] since these samplers simultaneously provide strong performance and tractable computation costs. Once we have obtained samples from the downstream posterior, we use these samples to form a Bayesian model average for test-time predictions.

In Figure 2c, we also observe the advantage of Bayesian inference (SGHMC) over MAP estimation (SGD) on the negative log-posterior yielded by our learned prior. Bayesian inference outperforms MAP estimation across all dataset sizes we consider.

In general, Bayesian inference can benefit most from an expressive prior. Indeed, priors reshape the whole posterior landscape — and the distinctive feature of a Bayesian approach is that it marginalizes over the whole posterior, rather than simply using an optimum as with standard training.

3.4 Practical Considerations

While our method requires a learned prior and a re-scaling coefficient, it is very easy to use and has minimal computational costs over standard fine-tuning routines. In particular, no expert intervention or knowledge of Bayesian methods is required, and only a single hyperparameter needs to be tuned. In the Appendix Section A, we evaluate the costs of each of the three stages of our pipeline in detail: (i) inferring the posterior on the source task; (ii) re-scaling the posterior to become an informative prior for the downstream task; (iii) using the informative prior in the downstream task.

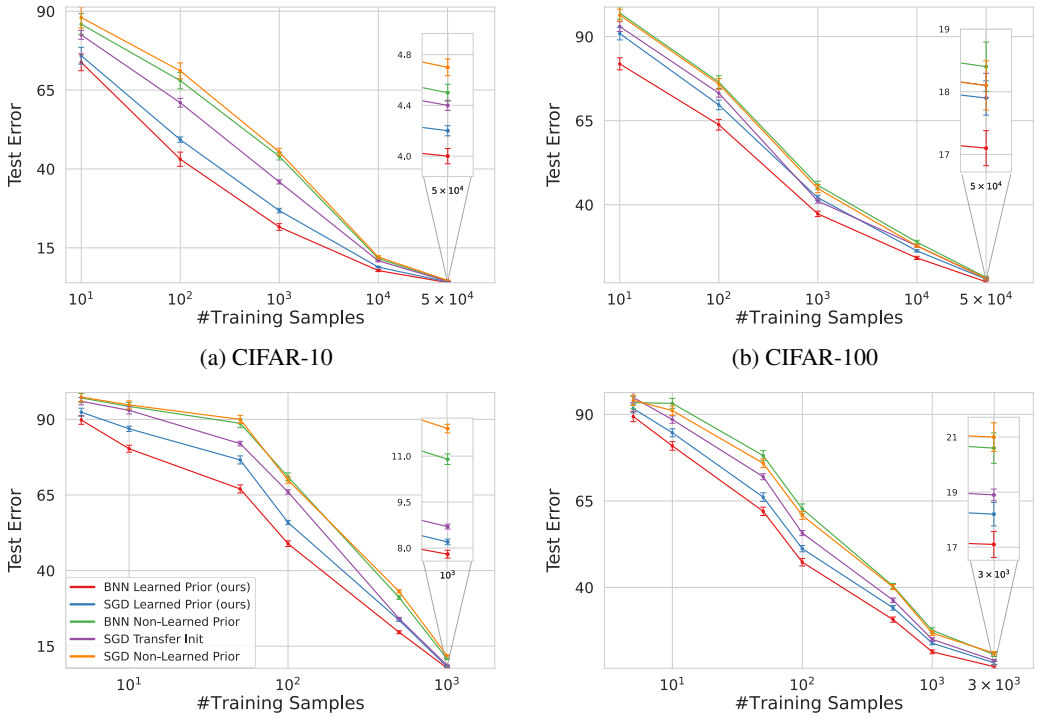
In short, (i) has a minor cost (e.g., $\frac{1}{4}$ of an epoch on ImageNet) that is a one-time cost when a pre-trained posterior is publicly released, as we have done; (ii) involves a single hyperparameter that can be tuned in the same way as other hyperparameters (simple validation grid search), does not require specialized expertise, and adds $\frac{1}{7}$ to the runtime of fine-tuning; (iii) has no additional cost if we do MAP optimization and has a cost comparable to deep ensembles if we run our implementation of Bayesian inference, both in terms of fine-tuning and test-time, and we show that Bayesian inference significantly outperforms deep ensembles.

4 Experiments

We now conduct a thorough empirical evaluation of our transfer learning pipeline on image classification and semantic segmentation. We generally consider five approaches, which we find have the following performance ordering: **(1)** Bayesian inference with learned priors, **(2)** SGD with learned priors, **(3)** SGD with standard pre-training, **(4)** Bayesian inference with non-learned zero-mean priors, **(5)** SGD with non-learned zero-mean priors. We note that both (1) and (2) are part of our framework, and that SGD with learned priors (2) significantly outperforms standard transfer learning (3).

4.1 Experimental Setting

We adopt the ResNet-50 and ResNet-101 architectures [18], and we scale the input image to 224×224 pixels to accommodate these feature extractors designed for ImageNet data. We use a SimCLR (SSL) ResNet-50 checkpoint [6] pre-trained on the ImageNet 1k dataset [9] and fit our prior to the SimCLR loss function. For the supervised setting, we use pre-trained `torchvision` ResNet-50 and ResNet-101 backbones. We perform image classification experiments on four downstream tasks: CIFAR-10, CIFAR-100 [30], Oxford Flowers-102 [36], and Oxford-IIIT Pets [38]. On semantic segmentation, we use a DeepLabv3+ system [4] with ResNet-50 and ResNet-101 backbone architectures, and we evaluate performance on the Pascal VOC 2012 [14] and Cityscapes [7] datasets. All error bars represent one standard error over 5 runs. We evaluate over a variety of downstream training set sizes, as transfer learning often involves limited downstream data.We provide a detailed description of hyperparameters in Appendix C.1.



(a) CIFAR-10 (b) CIFAR-100

(c) Oxford Flowers-102 (d) Oxford-IIIT Pets

Figure 3: **Performance comparison.** Learned priors outperform non-learned priors, and Bayesian inference generally outperforms SGD training, especially with learned priors. Moreover, learned priors tend to be more data-efficient. We use a ResNet-50 and SimCLR for pre-trained priors. The horizontal axis denotes the number of downstream training samples on a logarithmic scale. Error bars correspond to a standard error over 5 trials.

4.2 Performance Comparison

In Figure 3, we compare the five approaches described above across various dataset sizes. We observe the following: (i) learned priors consistently outperform SGD transfer learning which in turn outperforms non-learned priors; (ii) conditioned on using the same prior, Bayesian inference often outperforms SGD training; (iii) Bayesian inference adds more value when used with informative priors; (iv) learned priors are relatively most valuable on intermediate data sizes for the downstream task. Point (iv) is particularly interesting: unlike standard SGD transfer learning, which involves an initialization from the pretraining task, the learned priors provide an explicit inductive bias, enabling the resulting model to learn more efficiently from downstream data. Once there is a sufficient amount of downstream data, the source pre-training becomes less important, and the methods become more similar in performance, although there is still a significant gap.

Computation and comparison to deep ensembles. Our Bayesian model average (BMA) contains 10 samples from the posterior, and thus incurs a higher test-time cost than a single SGD-trained model. We therefore additionally compare to an equally sized ensemble of SGD-trained transfer learning models in Appendix B.3. We see that our BMA strongly outperforms the deep ensemble. In transfer learning, deep ensemble members are initialized with the same pre-trained checkpoint, and fine-tuning tends to stay in the same basin, such that the ensemble components are relatively homogenous in their predictions. Recall that SGD with our learned priors incurs essentially the same computational costs as standard transfer learning, but has much better performance.

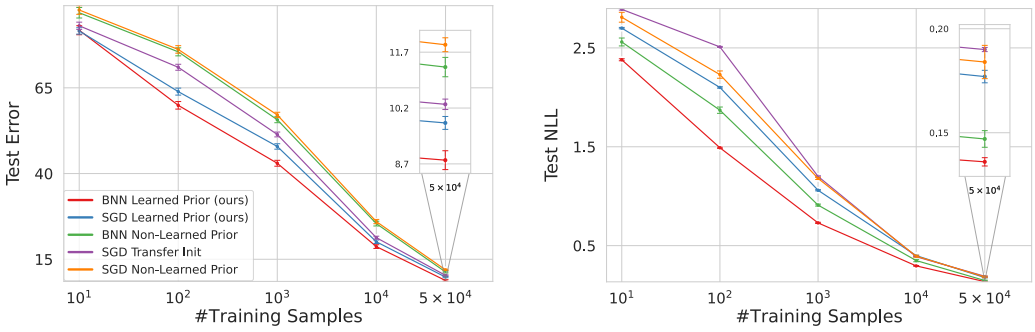
Comparison between self-supervised and supervised pre-training. In Appendix B.2 we provide additional evaluations with priors generated using `torchvision` ResNet-50 and ResNet-101 checkpoints. These priors differ from the SimCLR prior in that they are learned with labeled data rather than a self-supervision task. We see here also that learned supervised priors paired with Bayesian

inference consistently outperforms all baselines. We also see, notably, that the SSL priors are more transferable and outperform their supervised counterparts.

Evaluating uncertainty. We also measure predictive uncertainty via negative log test-likelihood (NLL) in Figure 4b, with additional results in Appendix B.4. We find that Bayesian transfer learning outperforms all other methods. However, we note that even though Bayesian inference with a non-learned prior has inferior accuracy to SGD with pre-training, it has superior test-likelihood — indicating that the confidence of SGD-based transfer learners is significantly miscalibrated, as likelihood accounts for both accuracy and uncertainty. We also evaluate the calibration of uncertainty using reliability diagrams (Figure 9 in the Appendix). These plots demonstrate that Bayesian inference with learned priors is the best calibrated among methods we consider. .

4.3 Out-of-Distribution Generalization

CIFAR-10.1 [42] is a test set of 2000 natural images modeled after CIFAR-10 with the same classes and image dimensions, following a similar dataset creation process to that described in the original CIFAR-10 paper. Models trained on CIFAR-10 consistently perform much worse on CIFAR-10.1 despite the images being similarly easy for humans to classify. Figure 4a indicates that our method achieves superior performance to SGD-based transfer learning and Bayesian inference with non-learned priors on CIFAR-10.1 across training set sizes, where training sets are sampled from CIFAR-10 training data.



(a) Distributional shift on CIFAR-10.1.

(b) Test Negative Log-Likelihood on CIFAR-10.

Figure 4: **Predictive likelihood and distribution shift.** The horizontal axis denotes the number of downstream CIFAR-10 training samples on a logarithmic scale. Error bars correspond to standard error over 5 trials. All experiments performed with ResNet-50 backbone pre-trained via SimCLR on ImageNet. **(a)** BNNs with our learned priors similarly achieve superior accuracy on OOD testing. **(b)** Bayesian inference with a learned prior achieves superior test NLL, but BNNs with a non-learned prior outperforms all other methods despite often performing worst in terms of accuracy.

4.4 Semantic Segmentation with ImageNet Priors

Popular segmentation methods use ImageNet pre-trained weights as an initialization for backbone parameters [4]. We observe that semantic segmentation models, which contain numerous parameters aside from those of the backbone, still benefit immensely from a learned prior over backbone parameters. Placing a strong prior, even if over only the backbone component of the segmentation system, enables us to more fully harness pre-training and boost performance across benchmark datasets PASCAL VOC 2012 and Cityscapes.

When constructing a prior, we apply SWAG to `torchvision` ("Supervised") and SimCLR ("SSL") ResNet-50 models and a `torchvision` ResNet-101 model (the authors of SimCLR did not release a ResNet-101 model). We assign a zero-mean isotropic Gaussian prior to the decoder and atrous convolution parameters of DeepLabv3+ which are not pre-trained.

In Table 1, we see that Bayesian inference with learned priors achieves superior Mean-IoU to both baselines (SGD and SGLD) without learned priors. Priors learned on the SimCLR objective again outperform those learned on labeled data. We also present experiments with MAP estimates (SGD)

for the loss functions induced by both learned priors in Appendix C.2. We again find that our learned priors boost the performance of MAP estimates as well, and thus are preferable to standard transfer learning, even for practitioners who do not intend to perform Bayesian inference.

Dataset	Backbone	SGD Transfer	Non-Learned Prior	Learned Prior Supervised	Learned Prior SSL
PASCAL VOC 2012	ResNet-50	73.17	73.16	73.72	**74.15**
	ResNet-101	75.53	75.79	**76.27**	*
Cityscapes	ResNet-50	73.17	75.52	76.09	**76.52**
	ResNet-101	77.04	77.02	**77.63**	*

Table 1: **Bayesian inference for semantic segmentation.** The performance (Mean-IoU) of SGLD with learned priors (both supervised and SSL) and a non-learned prior over ResNet-50 and ResNet-101 parameters for segmentation tasks. *SimCLR ResNet-101 checkpoints have not been released.

5 Discussion

Our work reveals several new key insights about deep transfer learning:

- *Modifying the loss surface on the downstream task through informative priors leads to significant performance gains, with and without Bayesian inference.*
- *Bayesian inference provides a particular performance boost with the informative priors.*
- *Informative priors lead to more data-efficient performance on the downstream task.*
- *The success of this approach depends on capturing key directions in the loss surface of the source task, which we represent through a low-rank plus diagonal covariance matrix.*
- *Standard transfer learning can be significantly miscalibrated, even providing worse likelihood than Bayesian methods from scratch on the downstream task.*
- *Priors learned via self-supervised pre-training transfer better than those learning via supervised learning.*

In short, pre-training your loss with care provides an easy drop-in replacement for conventional transfer learning that relies on initialization.

Acknowledgements

We thank Pavel Izmailov, Polina Kirichenko, and Wesley Maddox for helpful discussions. This research is supported by NSF CAREER IIS-2145492, NSF I-DISRE 193471, NIH R01DA048764-01A1, NSF IIS-1910266, NSF 1922658 NRT-HDR: FUTURE Foundations, Translation, and Responsibility for Data Science, Meta Core Data Science, Google AI Research, BigHat Biosciences, Capital One, and an Amazon Research Award.

References

[1] Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[2] Rohitash Chandra and Arpit Kapoor. Bayesian neural multi-source transfer learning. *Neurocomputing*, 378:54–64, 2020.

[3] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[4] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[5] Tianqi Chen, Emily Fox, and Carlos Guestrin. Stochastic gradient hamiltonian monte carlo. In *International conference on machine learning*, pages 1683–1691. PMLR, 2014.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[7] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[8] Zihang Dai, Hanxiao Liu, Quoc V Le, and Mingxing Tan. Coatnet: Marrying convolution and attention for all data sizes. *arXiv preprint arXiv:2106.04803*, 2021.

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

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[11] Kien Do, Truyen Tran, and Svetha Venkatesh. Semi-supervised learning with variational bayesian inference and maximum uncertainty regularization. *arXiv preprint arXiv:2012.01793*, 2020.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[13] Sayna Ebrahimi, Mohamed Elhoseiny, Trevor Darrell, and Marcus Rohrbach. Uncertainty-guided continual learning with bayesian neural networks. *arXiv preprint arXiv:1906.02425*, 2019.

[14] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

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

[16] Yansong Gao, Rahul Ramesh, and Pratik Chaudhari. Deep reference priors: What is the best way to pretrain a model? *arXiv preprint arXiv:2202.00187*, 2022.

[17] Danijar Hafner, Dustin Tran, Timothy Lillicrap, Alex Irpan, and James Davidson. Noise contrastive priors for functional uncertainty. In *Uncertainty in Artificial Intelligence*, pages 905–914. PMLR, 2020.

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

[19] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[20] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*, 2018.

[21] W Ronny Huang, Zeyad Emam, Micah Goldblum, Liam Fowl, Justin K Terry, Furong Huang, and Tom Goldstein. Understanding generalization through visualizations. *"I Can't Believe It's Not Better!" NeurIPS 2020 workshop*, 2020.

[22] Alexander Immer, Matthias Bauer, Vincent Fortuin, Gunnar Rätsch, and Khan Mohammad Emtiyaz. Scalable marginal likelihood estimation for model selection in deep learning. In *International Conference on Machine Learning*, pages 4563–4573. PMLR, 2021.

[23] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

[24] Pavel Izmailov, Patrick Nicholson, Sanae Lotfi, and Andrew Gordon Wilson. Dangers of bayesian model averaging under covariate shift. *arXiv preprint arXiv:2106.11905*, 2021.

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

[26] Neal Jean, Sang Michael Xie, and Stefano Ermon. Semi-supervised deep kernel learning: Regression with unlabeled data by minimizing predictive variance. *Advances in Neural Information Processing Systems*, 31, 2018.

[27] Sanyam Kapoor, Theofanis Karaletsos, and Thang D Bui. Variational auto-regressive gaussian processes for continual learning. In *International Conference on Machine Learning*, pages 5290–5300. PMLR, 2021.

[28] Alireza Karbalayghareh, Xiaoning Qian, and Edward R Dougherty. Optimal bayesian transfer learning. *IEEE Transactions on Signal Processing*, 66(14):3724–3739, 2018.

[29] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

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

[31] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. *arXiv preprint arXiv:1712.09913*, 2017.

[32] Hao Li, Pratik Chaudhari, Hao Yang, Michael Lam, Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. Rethinking the hyperparameters for fine-tuning. In *International Conference on Learning Representations*, 2019.

[33] Wesley Maddox, Shuai Tang, Pablo Moreno, Andrew Gordon Wilson, and Andreas Damianou. Fast adaptation with linearized neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 2737–2745. PMLR, 2021.

[34] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for bayesian uncertainty in deep learning. *Advances in Neural Information Processing Systems*, 32:13153–13164, 2019.

[35] Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

[36] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.

[37] Pingbo Pan, Siddharth Swaroop, Alexander Immer, Runa Eschenhagen, Richard Turner, and Mohammad Emtiyaz E Khan. Continual deep learning by functional regularisation of memorable past. *Advances in Neural Information Processing Systems*, 33:4453–4464, 2020.

[38] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats and dogs. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3498–3505. IEEE, 2012.

[39] Massimiliano Patacchiola, Jack Turner, Elliot J Crowley, Michael O'Boyle, and Amos J Storkey. Bayesian meta-learning for the few-shot setting via deep kernels. *Advances in Neural Information Processing Systems*, 33:16108–16118, 2020.

[40] Valerio Perrone, Huibin Shen, Matthias W Seeger, Cedric Archambeau, and Rodolphe Jenatton. Learning search spaces for bayesian optimization: Another view of hyperparameter transfer learning. *Advances in Neural Information Processing Systems*, 32, 2019.

[41] Naresh Balaji Ravichandran, Anders Lansner, and Pawel Herman. Semi-supervised learning with bayesian confidence propagation neural network. *arXiv preprint arXiv:2106.15546*, 2021.

[42] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do cifar-10 classifiers generalize to cifar-10? *arXiv preprint arXiv:1806.00451*, 2018.

[43] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.

[44] Daniel M Roy and Leslie Pack Kaelbling. Efficient bayesian task-level transfer learning. In *IJCAI*, volume 7, pages 2599–2604, 2007.

[45] Sebastian Ruder and Barbara Plank. Learning to select data for transfer learning with bayesian optimization. *arXiv preprint arXiv:1707.05246*, 2017.

[46] Tim GJ Rudner, Freddie Bickford Smith, Qixuan Feng, Yee Whye Teh, and Yarin Gal. Continual learning via function-space variational inference. In *ICML Workshop on Theory and Foundations of Continual Learning*, 2021.

[47] Dominik Schnaus. *Progressive Bayesian Neural Networks*. PhD thesis, Technical University of Munich, 2021.

[48] Hanna Tseran, Mohammad Emtiyaz Khan, Tatsuya Harada, and Thang D Bui. Natural variational continual learning. In *Continual Learning Workshop@ NeurIPS*, volume 2, 2018.

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

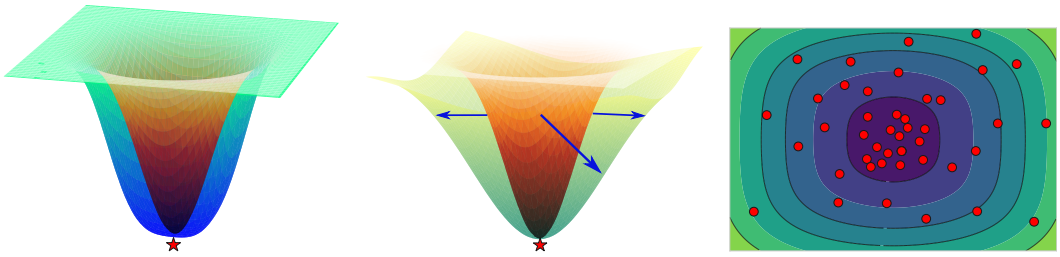
[50] Andrew Gordon Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *arXiv preprint arXiv:2002.08791*, 2020.

[51] Junyu Xuan, Jie Lu, and Guangquan Zhang. Bayesian transfer learning: An overview of probabilistic graphical models for transfer learning. *arXiv preprint arXiv:2109.13233*, 2021.

[52] Ruqi Zhang, Chunyuan Li, Jianyi Zhang, Changyou Chen, and Andrew Gordon Wilson. Cyclical stochastic gradient mcmc for bayesian deep learning. *arXiv preprint arXiv:1902.03932*, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Appendix E.
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] We include additional details about broader impact in the supplementary material.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We provide link to our code to replicate all experiments
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See sections B and A in the supp. material.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] We include error estimates in all figures.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We estimate compute in section A in the supp. material.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]



(a) Learning a prior (b) Prior rescaling (c) Bayesian inference

Figure 5: **Bayesian transfer learning pipeline.** We (a) learn a prior by fitting a probability distribution over feature extractor parameters to a pre-training posterior mode, (b) rescale the prior for a downstream task, and (c) use the prior for Bayesian inference on downstream tasks.

A Computational Considerations

While our method requires a learned prior and a re-scaling coefficient, it actually adds very little additional cost on top of standard fine-tuning routines. We now evaluate the costs of each of the three stages of our pipeline: (i) inferring the posterior on the source task; (ii) re-scaling the posterior to become an informative prior for the downstream task; (iii) using the informative prior in the downstream task.

For (i), we note that we used SWAG because it has already been well-established as a tractable, well-studied, and simple way to infer a non-diagonal posterior, and it can be used starting from a pre-trained checkpoint [34]. We apply SWAG to a pre-trained checkpoint. We then do 50 iterations of burn-in and 10 cycles of 200 iterations each for a total of 2050 iterations with a batch size of 128. In total, inferring the posterior costs only around a quarter of an epoch of training on ImageNet, and the computations in this step can be entirely avoided by users releasing pre-trained SWAG posteriors, as we do for all of the tasks in our paper.

For (ii), we only need to tune a single re-scaling hyperparameter. When we perform MAP optimization with our informative priors, tuning this scalar only adds $\frac{1}{7}$ of the total runtime of transfer learning (including tuning other hyperparameters such as learning rate). For example, full hyperparameter tuning (including the re-scaling coefficient) and running MAP on the full Oxford Flowers-102 dataset takes 20 hours in total, whereas hyperparameter tuning and running traditional SGD-based transfer learning takes 17 hours in total on a machine with a single NVIDIA A100 GPU, which is a comparable amount of hyperparameter tuning to other works and significantly less than others [32, 6]. We also note that our tuning is done with a simple validation grid-search, exactly the same way as any other hyperparameter tuning in standard transfer learning, and does not require any special expertise.

For (iii), we offer two options, (a) MAP optimization of the downstream posterior using the informative prior, and (b) full Bayesian inference with SGLD or SGHMC using the downstream posterior. (a) costs the same as standard transfer learning both at train and test time, and we show it works better than standard transfer learning. (b) Bayesian inference with our informative priors bears the same cost as deep ensembles or as Bayesian inference with non-learning priors, which we compare to in our work. A single chain of SGLD incurs the same cost as a single run of SGD, since the gradient of the log prior density can be computed without automatic differentiation in the same fashion as weight-decay, which we are replacing. We also compare multiple chains of SGLD with deep ensembles, which require the same cost, and show that Bayesian inference with our learned prior works significantly better.

B Classification

B.1 Implementation Details

Based on the evaluation protocols in the papers introducing the datasets, we report the top-one accuracy for CIFAR-10 and CIFAR-100 and mean per-class accuracy for Oxford-IIIT Pets and Oxford 102 Flowers. For Oxford 102 Flowers, we select hyperparameters based on the validation

sets specified by the dataset creators. While tuning hyperparameters, we hold out a subset of the training set for validation on the other datasets. After selecting the optimal hyperparameters from the validation set, we retrain the model using the selected parameters on both the training and validation images. Test results are reported.

We use ResNet-50 and ResNet-101 architectures [18] with supervised and SSL checkpoints [6] pre-trained on the ImageNet 1k dataset [9]. We use the same hyperparameters as in Chen et al. [6].

Learning the prior. In order to learn the prior, we use the SWAG algorithm [34] with cyclic learning rate presented in Zhang et al. [52]. We use SGD with a Nesterov momentum parameter of 0.9 for optimization. We select our initial learning rate from a logarithmic grid between 0.005 and 0.5 and evaluate 10000 iterations for each cycle. Other hyperparameters are from Chen et al. [6], including data augmentation which comprises color augmentation, blurring, random crops, and horizontal flips.

Downstream classification tasks. We train for $30,000$ steps with a batch size of 128 on CIFAR-10 and CIFAR-100, 16 for Oxford Flowers-102, and 32 for Oxford-IIIT Pets. We use SGD and SGHMC with momentum parameter of 0.9. During fine-tuning, we perform random crops with resizing to 224×224 and horizontal flips. At test time, we resize the images to 256 pixels along the shorter side and produce a 224×224 center crop. In our study, we select the learning rate and weight decay from a grid of 7 logarithmically spaced learning rates between 0.0001 and 0.1, and 7 logarithmically spaced weight decay values between $1e^{-6}$ and $1e^{-2}$, as well as without weight decay. These weight decay values are divided by the learning rate. For the SGHMC optimizer, we select the temperature from a logarithmic grid of 8 values between $1e^{0}$ and $1e^{-8}$, and we use predictions from 5 different chains for the final evaluation.

B.2 Supervised Pre-Training

As mentioned above, we conduct additional experiments with priors learned on labeled data. Figures 7 and 6 contain downstream task performance comparisons for Resnet-50 and Resnet-101 models with priors learned starting with `torchvision` checkpoints pre-trained on ImageNet 1k. These figures compare our method to SGD transfer learning with pre-trained initializations and Bayesian inference with non-learned Gaussian prior. Across both backbones, our method outperforms all baselines for nearly every number of train samples and for all four datasets, where the Bayesian inference model with a non-learned prior is always worse. (For full results see tables 2, 3, 4 and 5.

Model/# samples	10	100	1000	10000	50000
BNN Learned Prior	75.7	46.8	24.9	9.2	4.3
BNN Non-learned Prior	86.5	68.1	44.9	11.6	4.8
SGD Ensemble	78.7	57.2	31.6	10.9	5.2
SGD Transfer Init	81.9	62.1	36.5	11.6	4.4

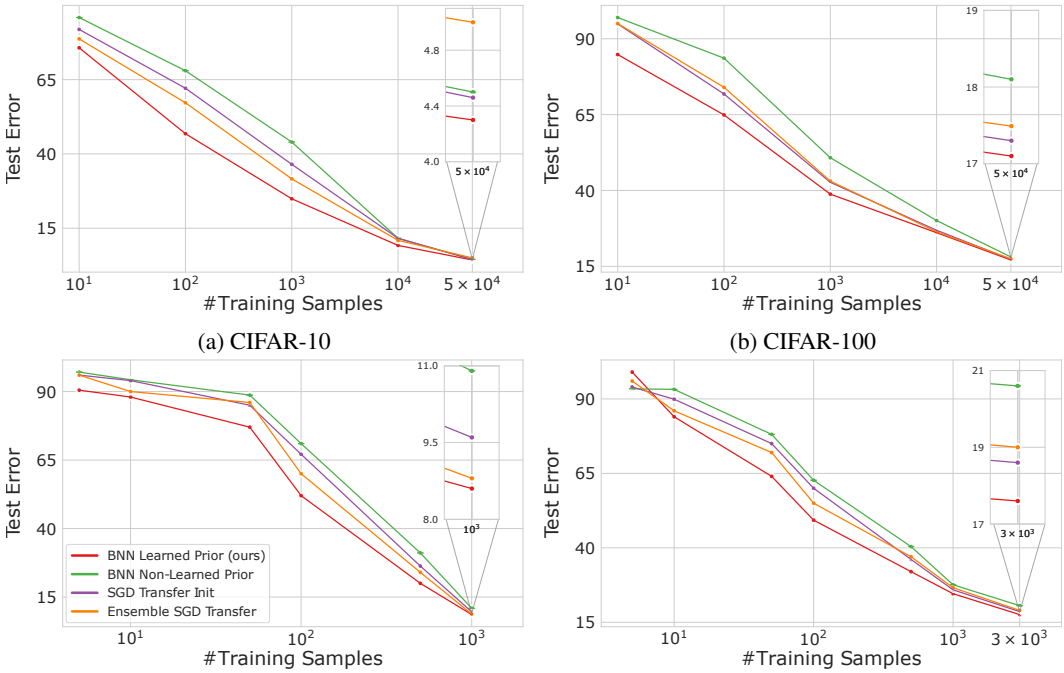
Table 2: CIFAR-10 test error with `torchvision` Resnet-50 prior/initialization.

Model/# samples	10	100	1000	10000	50000
BNN Learned Prior	86.8	64.9	38.8	26.1	17.2
BNN Non-learned Prior	97.0	83.6	50.8	30.1	19.2
SGD Ensemble	93.3	74.6	43.1	26.4	17.4
SGD Transfer Init	94.9	71.8	42.7	26.9	17.8

Table 3: CIFAR-100 test error with `torchvision` Resnet-50 prior/initialization.

B.3 Self-Supervised Pre-Training

In this section, we present additional data from Figures 3 and 4 found in the main body. Tables 10, 11, 12 and 13 correspond to Figure 3, while Table 14 corresponds to Figure 4.



(a) CIFAR-10 (b) CIFAR-100

(c) Oxford Flowers-102 (d) Oxford-IIIT Pets

Figure 6: **Performance comparison.** The test error of a ResNet-50 BNN equipped with supervised pre-trained prior (red) is consistently lower than that of standard SGD-based transfer learning from the same pre-trained checkpoint (purple), an equivalently sized ensemble of SGD-based transfer learning models (orange), or BNN with a mean zero isotropic Gaussian prior (green). The x-axis denotes the number of downstream training samples. Axes on a logarithmic scale.

Model/# samples	5	10	50	100	500	1000
BNN Learned Prior	90.5	88.0	77.2	52.8	20.4	8.5
BNN Non-learned Prior	97.1	94.3	88.7	71.0	35.0	10.7
SGD Ensemble	96.2	90.1	86.1	60.0	24.0	8.9
SGD Transfer Init	96.6	94.3	85.4	68.0	26.3	9.6

Table 4: Oxford-Flowers-102 test error with `torchvision` Resnet-50 prior/initialization.

B.4 Evaluating uncertainty

In Figure 8, we present the test negative log likelihood (NLL) for both the Bayesian methods and SGD-based methods for 4 datasets (CIFAR-10, CIFAR-100, Oxford Flowers-102 and Oxford-IIIT Pets). In general, we can see a similar trend for all the datasets where Bayesian transfer learning outperforms all other methods. In Figure 9, we present the reliability diagrams for CIFAR-10 and CIFAR-100 for ResNet-18. A perfectly calibrated network would express no difference between accuracy and confidence, represented by a dashed black line. Under-confident predictions are those below this line, whereas overconfident predictions are those above. We can see that Bayesian inference with learned priors are the best calibrated among the methods we consider. The error bars are computed on 5 runs, and the radius is one standard error.

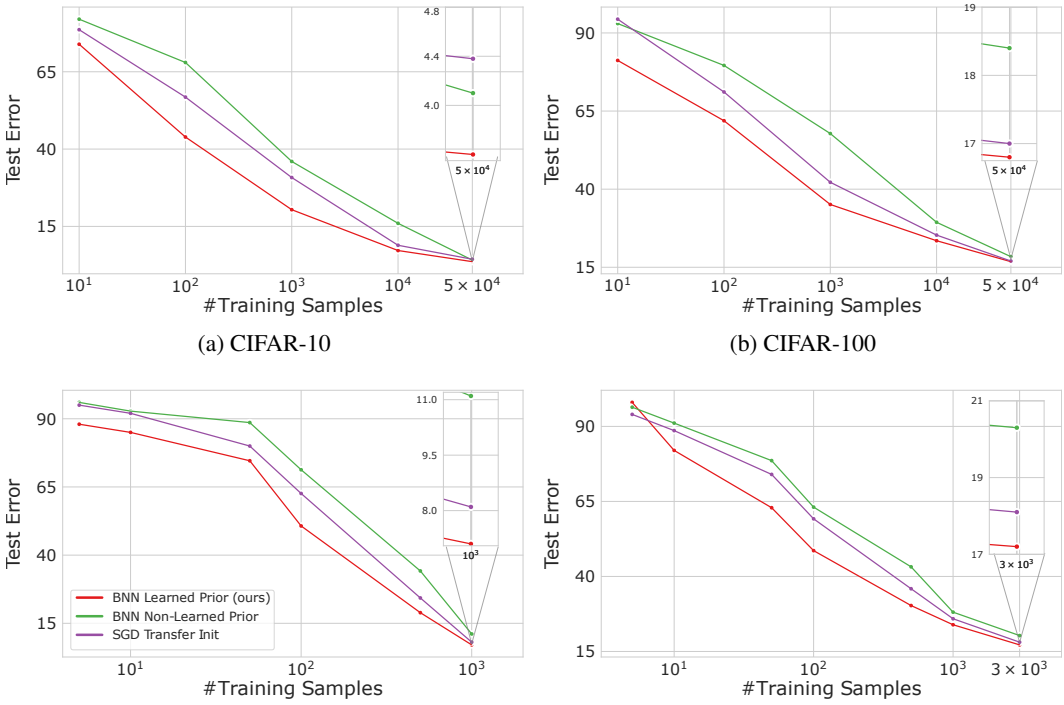
C Semantic Segmentation

C.1 Implementation Details

For our semantic segmentation evaluations, we use the DeepLabv3+ [4] model with ResNet-50 and ResNet-101 backbone architectures. We train our models using Pascal VOC 2012 and Cityscapes trainsets and evaluate on their *val* sets. We utilize the same hyperparameters employed in Chen et al.

Model/# samples	5	10	50	100	500	1000	3000
BNN Learned Prior	98.2	84.0	64.7	49.3	32.0	24.6	17.6
BNN Non-learned Prior	93.4	93.2	78.1	62.7	45.4	38.6	25.2
SGD Ensemble	96.6	86.3	72.1	55.0	37.9	26.6	19.3
SGD Transfer Init	94.5	89.9	75.4	61.1	36.1	25.9	18.6

Table 5: Oxford-IIIT Pets test error with `torchvision` Resnet-50 prior/initialization.



(a) CIFAR-10 (b) CIFAR-100

(c) Oxford Flowers-102 (d) Oxford-IIIT Pets

Figure 7: **Performance comparison.** The test error of a ResNet-101 BNN equipped with supervised pre-trained prior (red) is consistently lower than that of standard SGD-based transfer learning from the same pre-trained checkpoint (purple), or BNN with a mean zero isotropic Gaussian prior (green). The x-axis denotes the number of downstream training samples. Axes on a logarithmic scale.

[4] with a few modifications: our learning rate schedule is the *poly* policy [3] with initial value 0.01 for Pascal VOC 2012 and 0.1 for Cityscapes, our crop size is 513×513 for Pascal VOC 2012 and 768×768 for Cityscapes. Our *output stride* is 16 for both training and evaluating, and we perform random scale data augmentation during training. In our SGD and SGLD optimization, we employ Nesterov momentum, and we set the momentum coefficient to 0.9. We use batches of 16 images and weight decay with a coefficient of 10^{-4}. In addition, for the SGLD optimizer, the temperature is set to 2×10^{-8}. We train all models for 30k iterations.

C.2 MAP Estimation

We also compare MAP estimates (SGD) on the loss induced by our learned priors to the baseline with only weight decay. Table 15 and 16 contain the results using ResNet-50 and ResNet-101 backbones, respectively. On both architectures and both datasets, learned priors boost performance on these experiments without using Bayesian inference at all.

D Alternative Methods for Re-Scaling the Prior

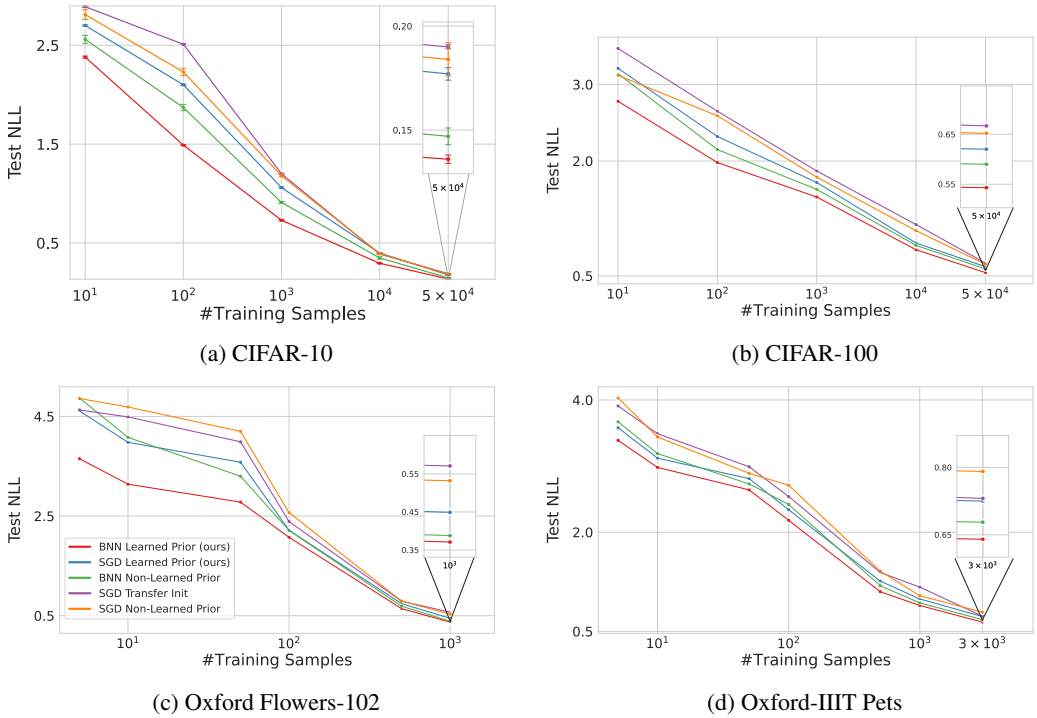
As discussed previously, we tune a coefficient to re-scale our learned prior by maximizing the validation log-likelihood over a grid of coefficient values. Our objective in this section is to examine

Model/# samples	10	100	1000	10000	50000
BNN Learned Prior	73.9	43.9	20.4	7.2	3.7
BNN Non-learned Prior	82.1	68.7	36.3	16.4	4.1
SGD Transfer Init	78.6	56.8	30.8	8.9	4.3

Table 6: CIFAR-10 test error with `torchvision` Resnet-101 prior/initialization.

Model/# samples	10	100	1000	10000	50000
BNN Learned Prior	81.2	61.9	35.1	23.5	16.8
BNN Non-learned Prior	93.5	79.6	58.8	32.0	20.2
SGD Transfer Init	94.4	71.1	44.2	25.9	17.0

Table 7: CIFAR-100 test error with `torchvision` Resnet-101 prior/initialization.



(a) CIFAR-10

(b) CIFAR-100

(c) Oxford Flowers-102

(d) Oxford-IIIT Pets

Figure 8: **Predictive likelihood.** The horizontal axis denotes the number of downstream CIFAR-10 training samples on a logarithmic scale. All experiments performed with ResNet-50 backbone.

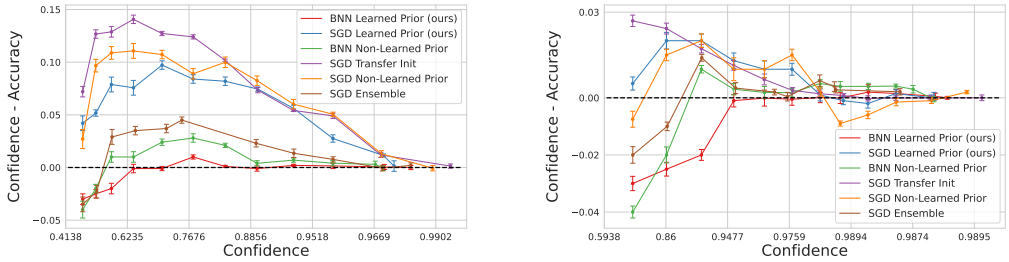
three alternative methods for tuning coefficients to re-scale the prior. First, we consider maximizing the Laplace approximation to the marginal likelihood, which does not require holding out validation data [22]. Marginal likelihood maximization can be performed online together with MAP estimation in deep learning. Specifically, we try tuning a single coefficient and also per-layer coefficients for re-scaling the prior's covariance matrix in this fashion. A third alternative is to instead tune two coefficients: one for the diagonal component of the covariance matrix, and one for the low-rank component. We use a grid-search in this case to tune the two coefficients. Table 17 contains the results for these three different methods as well as the method advocated in the main body of our draft using the SimCLR Resnet-50 backbone and SGHMC for downstream fine-tuning. We observe that on most datasets, tuning a single covariance re-scaling hyperparameter using marginal likelihood does not improve the results reported in the paper. Using different coefficients for each layer results in slight improvements on some datasets. On CIFAR-10, Oxford Flowers-102, and Oxford-IIIT Pets, test errors decreased by $0.03\% - 0.08\%$. Finally, on most datasets, re-scaling the diagonal and low-rank components of the covariance matrix independently results in slightly better performance. For example, the test errors on CIFAR-100, Oxford Flowers-102, and Oxford-IIIT Pets decrease

Model/# samples	5	10	50	100	500	1000
BNN Learned Prior	88.0	81.0	74.6	50.7	18.9	7.1
BNN Non-learned Prior	96.0	92.8	88.6	75.1	41.1	21.7
SGD Transfer Init	95.0	90.2	80.8	59.2	22.3	8.6

Table 8: Oxford-Flowers-102 test error with `torchvision` Resnet-101 prior/initialization.

Model/# samples	5	10	50	100	500	1000	3000
BNN Learned Prior	96.1	82.4	62.9	48.6	30.3	23.9	17.2
BNN Non-learned Prior	96.4	91.1	78.6	67.7	56.9	38.2	26.2
SGD Transfer Init	94.0	88.6	74.3	59.2	35.5	25.0	18.1

Table 9: Oxford-IIIT Pets test error with `torchvision` Resnet-101 prior/initialization.



(a) CIFAR-100

(b) CIFAR-10

Figure 9: **Reliability diagrams** - Bayesian transfer learning is capable of significantly improving calibration over standard transfer training (SGD Transfer Init) as well as non-learned prior BNN. Error bars are computed over 5 runs.

by $0.03\% - 0.11\%$. We note that these alternative methods bear an additional computational and implementational cost.

E Limitations

Limitations. Bayesian model averaging provides particularly compelling results, especially in terms of calibration and data efficiency, but does incur some additional computational cost. On the other hand, our SGLD based sampling procedures outperform deep ensembles, which have comparable computational costs. Moreover, our informative priors with MAP optimization still provide a clear performance boost over standard transfer learning, with negligible overhead. In terms of applications, we have limited our considerations to vision settings, for a focused exposition.

Directions for future research. Pre-training has been particularly transformative in natural language processing, where informative priors could be used to great effect. More broadly studying loss-surface alignment between tasks could also be informative for understanding how to build models that provide good domain generalization and robustness to spurious correlations.

Model/# samples	10	100	1000	10000	50000
BNN Learned Prior	73.8 ± 3.2	43.1 ± 3.1	21.6 ± 1.6	7.8 ± 0.5	4.0 ± 0.1
SGD Learned Prior	75.9 ± 2.1	49.3 ± 1.9	26.8 ± 1.1	8.9 ± 0.3	4.2 ± 0.1
BNN LP Single-Chain	74.0 ± 2.7	48.4 ± 2.8	25.4 ± 1.3	8.3 ± 0.5	4.2 ± 0.2
BNN LP Laplace	86.9 ± 3.1	66.4 ± 2.2	37.1 ± 2.1	11.2 ± 0.6	4.6 ± 0.2
BNN Non-learned Prior	86.1 ± 3.9	68.0 ± 3.8	44.9 ± 1.7	11.6 ± 0.6	4.4 ± 0.1
SGD Ensemble	81.3 ± 2.2	57.0 ± 3.1	32.3 ± 1.4	9.4 ± 0.5	4.6 ± 0.1
SGD Transfer Init	83.6 ± 1.8	61.0 ± 2.1	36.8 ± 1.0	10.9 ± 0.4	4.4 ± 0.1

Table 10: CIFAR-10 test error with SimCLR (SSL) Resnet-50 prior/initialization.

Model/# samples	10	100	1000	10000	50000
BNN Learned Prior	81.9 ± 3.7	63.8 ± 2.4	37.3 ± 1.2	24.2 ± 0.7	16.4 ± 0.4
SGD Learned Prior	90.9 ± 2.8	69.7 ± 1.9	41.1 ± 1.1	26.3 ± 0.5	17.3 ± 0.4
BNN LP Laplace	90.1 ± 3.1	73.6 ± 2.5	42.8 ± 1.9	27.9 ± 1.2	18.1 ± 0.8
BNN LP Single-Chain	84.4 ± 3.8	69.6 ± 2.1	$38.2 \pm 2.$	25.1 ± 0.7	16.8 ± 0.5
BNN Non-learned Prior	97.0 ± 3.9	84.0 ± 2.8	50.8 ± 1.8	28.9 ± 0.8	18.8 ± 0.6
SGD Ensemble	89.2 ± 2.0	70.9 ± 1.7	44.0 ± 1.2	26.6 ± 0.7	17.2 ± 0.4
SGD Transfer Init	93.0 ± 2.2	73.2 ± 1.8	45.5 ± 0.9	27.8 ± 0.6	17.9 ± 0.3

Table 11: CIFAR-100 test error with SimCLR (SSL) Resnet-50 prior/initialization.

Model/# samples	5	10	50	100	500	1000
BNN Learned Prior	89.8 ± 3.1	80.3 ± 1.7	67.3 ± 1.8	48.9 ± 1.4	19.7 ± 0.7	7.8 ± 0.2
SGD Learned Prior	90.2 ± 2.0	81.0 ± 1.3	67.0 ± 1.1	49.9 ± 1.1	20.6 ± 0.6	8.2 ± 0.1
BNN LP Laplace	97.4 ± 2.3	93.1 ± 2.1	89.9 ± 1.4	69.8 ± 1.4	32.1 ± 0.8	11.9 ± 0.2
BNN LP Single-Chain	90.3 ± 2.8	80.8 ± 1.8	67.4 ± 1.1	49.5 ± 1.7	20.1 ± 0.9	8.1 ± 0.2
BNN Non-learned Prior	97.1 ± 2.7	94.3 ± 2.0	88.7 ± 2.1	71.0 ± 2.0	35.0 ± 0.9	10.9 ± 0.3
SGD Ensemble	93.8 ± 2.0	91.2 ± 1.6	84.3 ± 1.2	56.8 ± 1.0	21.9 ± 0.3	8.5 ± 0.2
SGD Transfer Init	96.0 ± 1.8	93.0 ± 1.8	82.0 ± 1.3	66.0 ± 1.1	24.1 ± 0.6	8.7 ± 0.1

Table 12: Oxford-Flowers-102 test error with SimCLR (SSL) Resnet-50 prior/initialization.

Model/# samples	5	10	50	100	500	1000	3000
BNN Learned Prior	89.4 ± 2.6	80.9 ± 1.9	62.0 ± 1.8	47.3 ± 1.6	30.7 ± 1.1	21.4 ± 0.9	17.1 ± 0.7
SGD Learned Prior	91.7 ± 1.9	84.7 ± 1.7	66.1 ± 1.9	51.2 ± 1.3	34.1 ± 0.8	23.9 ± 0.6	18.2 ± 0.6
BNN LP Laplace	94.1 ± 3.2	91.1 ± 2.7	75.9 ± 2.7	60.8 ± 1.7	40.1 ± 1.4	26.8 ± 1.4	21.3 ± 0.3
BNN LP Single-Chain	91.0 ± 2.7	84.9 ± 2.5	65.2 ± 2.9	49.4 ± 1.9	31.9 ± 1.8	23.7 ± 1.6	18.1 ± 0.6
BNN Non-learned Prior	93.4 ± 2.9	93.2 ± 2.1	78.1 ± 2.2	62.7 ± 2.1	45.4 ± 1.1	38.6 ± 1.1	25.2 ± 0.8
SGD Ensemble	92.2 ± 2.1	88.1 ± 1.4	68.3 ± 2.1	55.7 ± 1.5	35.0 ± 1.2	24.5 ± 1.0	18.1 ± 0.5
SGD Transfer Init	94.9 ± 2.1	88.5 ± 1.6	72.0 ± 1.3	55.7 ± 1.1	36.3 ± 1.0	25.0 ± 0.7	18.9 ± 0.3

Table 13: Oxford-IIIT Pets test error with SimCLR (SSL) Resnet-50 prior/initialization.

Model/# samples	10	100	1000	10000	50000
BNN Learned Prior	81.8 ± 2.0	59.9 ± 1.7	43.0 ± 1.3	18.7 ± 0.8	8.8 ± 0.4
SGD Learned Prior	81.6 ± 1.4	63.9 ± 1.5	48.0 ± 1.2	20.1 ± 0.8	9.8 ± 0.3
BNN Non-learned Prior	86.9 ± 2.3	78.5 ± 1.8	59.0 ± 1.3	32.0 ± 1.0	11.3 ± 0.4
SGD Transfer Init	83.1 ± 1.6	69.0 ± 1.3	52.0 ± 1.1	21.3 ± 0.6	10.3 ± 0.2

Table 14: Classification error with self-supervised pre-learning - CIFAR-10.1

Table 15: **MAP estimation for semantic segmentation (ResNet-50 backbone).** Comparing the performance (Mean-IoU) of SGD with learned priors (both `torchvision` supervised and SimCLR SSL) to a non-learned prior (weight decay with `torchvision` initialization) over DeepLabv3+ backbone parameters for downstream segmentation with SGD rather than Bayesian inference. Evaluations are conducted on *val* sets.

Dataset	**Non-Learned Prior MAP**	**Supervised Prior MAP**	**SSL Prior MAP**
PASCAL VOC 2012	73.17	73.27	**73.48**
Cityscapes	75.52	75.57	**76.15**

Table 16: **MAP estimation for semantic segmentation (ResNet-101 backbone).** Comparing the performance (Mean-IoU) of SGD with `torchvision` learned priors to a non-learned prior (weight decay with `torchvision` initialization) over DeepLabv3+ backbone parameters for downstream segmentation with SGD rather than Bayesian inference. Evaluations are conducted on *val* sets.

Dataset	Non-Learned Prior MAP	Supervised Prior MAP
PASCAL VOC 2012	75.53	**75.89**
Cityscapes	77.04	**77.27**

Table 17: **Alternative methods for prior re-scaling (SimCLR ResNet-50 backbone)** - test error corresponding to different methods for re-scaling the prior.

Dataset	CIFAR-100	CIFAR-10	Oxford Flowers-102	Oxford-IIIT Pets
Single Coefficient - Grid Search	17.14	4.02	7.85	17.12
Single Coefficient - Marginal Likelihood	17.13	4.02	7.87	17.06
Per-Layer - Marginal Likelihood	17.14	3.99	7.79	17.05
Separate Scales for Low-Rank and Diagonal	17.11	4.01	7.81	17.03