Learning the Regularization Strength for Deep Fine-Tuning via a Data-Emphasized Variational Objective

Ethan Harvey*
Department of Computer Science
Tufts University
ethan.harvey@tufts.edu

Mikhail Petrov*
Department of Mechanical Engineering
Tufts University
mikhail.petrov@tufts.edu

Michael C. Hughes
Department of Computer Science
Tufts University
michael.hughes@tufts.edu

Abstract

A number of popular transfer learning methods rely on grid search to select regularization hyperparameters that control over-fitting. This grid search requirement has several key disadvantages: the search is computationally expensive, requires carving out a validation set that reduces the size of available data for model training, and requires practitioners to specify candidate values. In this paper, we propose an alternative to grid search: directly learning regularization hyperparameters on the full training set via model selection techniques based on the *evidence lower bound* ("ELBo") objective from variational methods. For deep neural networks with millions of parameters, we specifically recommend a modified ELBo that upweights the influence of the data likelihood relative to the prior while remaining a valid bound on the evidence for Bayesian model selection. Our proposed technique overcomes all three disadvantages of grid search. We demonstrate effectiveness on image classification tasks on several datasets, yielding heldout accuracy comparable to existing approaches with far less compute time.

1 Introduction

When fine-tuning deep neural networks (DNNs), a significant amount of computational resources are devoted to tuning hyperparameters that control model complexity to manage tradeoffs between under- and over-fitting on the target task of interest. One widespread example would be tuning the value of the scalar multiplier that controls the strength of an additive loss term computed as the sum-of-squares on weight coefficient values, known in various communities as L2 regularization [33], Ridge penalty [14, 21], or "weight decay" [24, 9]. A common technique for tuning such hyperparameters is to hold out a dedicated validation set and use grid search to find the hyperparameters that perform best on the validation set [36, 32].

While reasonably effective and in widespread use to manage over-fitting in recent transfer learning [45, 39], using grid search for hyperparameter selection has three key drawbacks. First and perhaps most important, the need to train separate models at each possible value in the grid significantly

*Both authors made lead author level contributions
Open-source code: https://github.com/tufts-ml/data-emphasized-ELBo

Workshop on Fine-Tuning in Machine Learning (FITML@NeurIPS 2024)

increases computational runtime and resources. Second, the need to carve out a validation set to assess performance reduces the amount of available data that can inform model training. This can cause under-fitting, especially when available data has limited size. Finally, grid search requires a list of candidate values specified in advance, yet ideal values may vary widely depending on the data and specific classification task at hand.

We take another approach to hyperparameter selection, inspired by a pragmatic Bayesian perspective. Suppose we model observable dataset $\mathcal{D}$ via a likelihood $p(\mathcal{D}|\theta)$, where θ is a high-dimensional parameter to be estimated, with prior distribution $p(\theta|\eta)$ controlled by hyperparameter η (say just 1-5 dimensions). Instead of point estimating a specific θ, η pair, we can instead estimate a posterior $p(\theta|\mathcal{D}, \eta)$ while simultaneously directly learning η to optimize $p(\mathcal{D}|\eta) = \int_\theta p(\mathcal{D}, \theta|\eta) d\theta$. This latter objective $p(\mathcal{D}|\eta)$ is known as the *marginal likelihood* or *evidence* [26]. The evidence naturally encodes a notion of Occam's razor, favoring the hyperparameter setting that leads to the simplest model that fits the data well, while penalizing complex models that over-fit the training data [18, 27, 2]. Learning η to maximize evidence (or equivalently, the logarithm of evidence) via gradient descent avoids all three issues with grid search: we need only one run of gradient descent (not separate efforts for each candidate η value in a grid), we can use all available labeled data for training without any validation set, and we can explore the full continuous range of possible η values rather than a limited discrete set that must be predefined.

While elegant in theory, this vision of selecting hyperparameters via maximizing evidence is difficult in practice for most models of interest due to the intractable high-dimensional integral that defines the evidence. For modern deep image classifiers with millions of parameters, computing the evidence directly seems insurmountable even for a specifc η, let alone optimizing evidence to select a preferred η value. In this work, we use and extend tools from variational Bayesian methods [3, 19], specifically tractable lower bounds on the evidence, to make hyperparameter selection for fine-tuning deep neural image classifiers possible.

Ultimately, we contribute methods that should help practitioners perform cost-effective transfer learning on custom datasets. When available data is plentiful, our experiments suggest our approach is competitive in accuracy while reducing total training time from 16 hours for L2-SP [45] and 150 hours for PTYL [39] (using the grid search ranges recommended by the original authors) to under 3 hours. When available data is limited, e.g., only 5-300 labeled examples per class, our experiments suggest our approach can be particularly effective in improving accuracy and runtime.

2 Background

Problem setup. Consider training a neural network classifier composed of two parts. First, we have a backbone encoder f with weights $w \in \mathbb{R}^D$, which non-linearly maps input vector x_i to a representation vector $z_i \in \mathbb{R}^H$ (which includes an "always one" feature to handle the need for a bias/intercept). The second part is a linear-decision-boundary classifier head with weights $V \in \mathbb{R}^{C \times H}$, which leads to probabilistic predictions over C possible classes. We wish to find values of these parameters that produce good classification decisions on a provided *target task* dataset of N pairs x_i, y_i of features x_i and corresponding class labels $y_i \in \{1, 2, \ldots C\}$. For transfer learning, we assume the backbone weights w have been pretrained to high-quality values μ on a source task.

Deep learning view. Typical approaches to transfer learning in the deep learning tradition (e.g., baselines in [45]) would pursue empirical risk minimization with an L2-penalty on weight magnitudes for regularization, training to minimize the loss function

$$L(w, V) := \frac{1}{N} \left(\sum_{i=1}^{N} \ell(y_i, V f_w(x_i)) + \frac{\alpha}{2} ||w||_2^2 + \frac{\beta}{2} ||\text{vec}(V)||_2^2 \right), \tag{1}$$

where ℓ represents a cross-entropy loss indicating agreement with the true label y_i (one of C categories), while the L2-penalty on weights w, V encourages their magnitude to *decay* toward zero, and this regularization is thus often referred to as "weight decay". The key hyperparameters $\alpha \geq 0, \beta \geq 0$ encode the strength of the L2 penalty, with higher values yielding simpler representations and simpler decision boundaries. Model training would thus consist of solving $w^*, V^* \leftarrow \text{argmin}_{w,V} L(w, V)$ via stochastic gradient descent, given fixed hyperparameters α, β. In turn, the values of α, β would be selected via grid search seeking to optimize ℓ or error on a validation set.

Bayesian view. Bayesian interpretation of this neural classification problem would define a joint probabilistic model $p(w, V, y_{1:N})$ decomposed into factors $p(w)p(V)\prod_i p(y_i|w, V)$ defined as:

$$p(w)=\mathcal{N}(w|\mu_p, \lambda\Sigma_p),\ \ p(V)=\mathcal{N}(\text{vec}(V)|0, \tau I),\ \ p(y_i|w, V)=\text{CatPMF}(y_i|\text{SM}(V f_w(x_i))). \quad (2)$$

Here, $\lambda \geq 0, \tau \geq 0$ are hyperparameters controlling over/under-fitting, μ_p, Σ_p represent *a priori* knowledge of the mean and covariance of backbone weights w (see paragraph below), and SM is the softmax function. Note x_i is a known fixed value, not a random variable in the model; we leave such fixed quantities out of probabilistic conditioning notation for simplicity. To fit this model, pursing maximum a-posteriori (MAP) estimation of both w and V recovers the objective in Eq. (1) when we set $\alpha = \frac{1}{\lambda}$, $\beta = \frac{1}{\tau}$, $\mu_p = 0, \Sigma_p = I$ and have ℓ set to $-\log p(y_i|w, V)$.

Need for validation set and grid search. Selecting α, β (or equivalently λ, τ) to directly minimize Eq. (1) on the training set alone is not a coherent way to guard against over-fitting. Regardless of data content or weight parameter values, we would select $\alpha^* = 0, \beta^* = 0$ to minimize L as a function of α, β and thus enforce no penalty on weight magnitudes at all. Carving out a validation set for selecting these hyperparameters is thus critical to avoid over-fitting when point estimating w, V.

Backbone prior mean/covariance. Several recent transfer learning approaches correspond to specific settings of the backbone mean and covariance μ_p, Σ_p. Let vector μ represent a specific setting of pretrained backbone weights w that performs well on a source task. Setting $\mu_p{=}0, \Sigma_p{=}I$ recovers the conventional approach to transfer learning, which we call L2-zero, where regularization pushes backbone weights to zero and the pretrained value μ only informs the initial value of backbone weights w before any SGD [45, 13]. Instead, setting the prior mean as $\mu_p = \mu$ along with $\Sigma_p = I$ recovers the *L2 starting point* (L2-SP) regularization method of [45], also covered in [5]. Further setting Σ_p to the estimated covariance matrix Σ of a Gaussian approximation of the posterior over backbone weights for the source task recovers "Pre-Train Your Loss" (PTYL) [39].

Table 1: Mean and covariance of backbone weights w for several transfer learning approaches.

Method	$p(w)$	Init.
L2-zero	$\mathcal{N}(0, \lambda I)$	μ
L2-SP	$\mathcal{N}(\mu, \lambda I)$	μ
PTYL	$\mathcal{N}(\mu, \lambda\Sigma)$	μ

Need to specify a search space. Selecting α, β (or equivalently λ, τ) via grid search requires practitioners to specify a grid of candidate values spanning a finite range. For the PTYL method, the optimal search space for these key hyperparameters is still unclear. For the same prior and the same datasets, the search space has varied between works: originally, the method creators recommended large values from 1e0 to 1e10 [39, 13]. Later works search far smaller values (1e-5 to 1e-3) [37].

3 Methods

3.1 Variational methods for posterior estimation

For most complex probabilistic models, such as $p(y_{1:N}, w, V)$ defined in Eq. (2), evaluating the posterior distribution probability density function (PDF) $p(w, V|y_{1:N})$ is challenging even for specific w, V value, because $p(w, V|y_{1:N}) \propto \frac{1}{p(y_{1:N})} p(y_{1:N}, w, V)$, and the normalization term is exactly the intractable evidence $p(y_{1:N})$, defined as a challenging high-dimensional integral $p(y_{1:N}) = \int p(y_{1:N}, w, V) dw dV$. This evaluation roadblock also makes sampling from the posterior difficult. As a remedy, variational methods [3] provide a framework to estimate an approximate posterior $q(w, V) \approx p(w, V|y_{1:N})$ that belongs to a simpler parametric family of distributions. For example, we may define q as

$$q(w, V) = q(w)q(V), q(w){=}\mathcal{N}(w|\bar{w}, \bar{\sigma}^2 I), q(V){=}\mathcal{N}(\text{vec}(V)|\text{vec}(\bar{V}), \bar{\sigma}^2 I). \quad (3)$$

We denote the free parameters of q with bars to distinguish them from model parameters. Weights $\bar{w}, \bar{V}$ represent means of the backbone and classifier head. Scalar $\bar{\sigma}^2 > 0$ controls variance around these means. We use one variance parameter for simplicity; future work could allow separate parameters for different classifier parts. Well-known results in variational inference then suggest fitting the parameters of q by maximizing an objective function known as the *evidence lower bound* ("ELBo", [3]), denoted as J_{ELBo} and defined for our model p and approximate posterior q as:

$$J_{\text{ELBo}} := \mathbb{E}_{q(w,V)}\left[\log \sum_{i=1}^{N} p(y_i|w, V)\right] - \mathbb{KL}(q(w)\|p(w|\lambda)) - \mathbb{KL}(q(V)\|p(V|\tau)). \quad (4)$$

This objective is a function of data $y_{1:N}, x_{1:N}$, variational posterior parameters $\bar{w}, \bar{V}, \bar{\sigma}$, and prior hyperparameters λ, τ. Maximizing J_{ELBo} can be shown equivalent to finding the specific q that is "closest" to the true posterior (in the sense of KL divergence). Further, as the name of the objective suggests, we can show mathematically that $J_{\text{ELBo}}(y_{1:N}, \ldots, \lambda, \tau) \leq \log p(y_{1:N}|\lambda, \tau)$. That is, the ELBo is a lower bound on the log evidence (which itself is a function of the chosen λ, τ). This relation suggests a potential utility for data-driven selection of these hyperparameters. Prior work [7] argues that using ELBo for model selection has strong theoretical guarantees, even under misspecification.

Optimizing the ELBo for neural net classifiers. Of the 3 additive terms defining the ELBo objective, both KL terms are the most tractable to evaluate and compute gradients of, as they involve KL divergences of multivariate Gaussians amenable to closed-form expressions. The first term, the expected log likelihood, requires slightly more care. To handle this non-conjugate expectation for neural network classifiers a key insight from [4] suggests applying the "reparameterization trick", a general way to estimate gradients in Monte Carlo fashion [44, 30]. To implement this estimator for this likelihood term, we first draw vectors the same size as w and $\text{vec}(V)$, denoted ϵ^w and ϵ^V, from a standard normal. Then, we transform to samples of w, V from q via well-known transformations (e.g., for the backbone: $w \leftarrow \bar{w} + \bar{\sigma}\epsilon^w$). Averaging over such independent and identically distributed (IID) samples w, V from q allows a Monte Carlo approximation of the expected log likelihood. Gradients follow via automatic differentiation of that Monte Carlo estimator's computational graph. We find using *just one sample* per training step is sufficient and most efficient.

Related work on Bayesian neural nets. Variational inference for neural networks has a long history, dating back to work by Hinton and Van Camp [17] and Graves [10]. Modern efforts include MOdel Priors Extracted from Deterministic DNN (MOPED) [22], which focuses on using informative priors to enable scalable variational inference, not to select regularization hyperparameters.

3.2 Learning key hyperparameters λ, τ with the ELBo

While most works on variational inference concentrate on learning just the posterior over model parameters, it is staightforward to optimize the ELBo objective for prior hyperparameters λ, τ as well. Recall that the term $-\mathbb{KL}(q(w)\|p(w))$ is tractable since both the approximate posterior and prior are multivariate Gaussian. For instance, in our particular model in Eq. (2), the KL divergence between two Gaussians [31] simplifies for the backbone KL term as:

$$-\mathbb{KL}(q(w)\|p(w)) = -\frac{1}{2}\left[\frac{\bar{\sigma}^2}{\lambda}\operatorname{Tr}(\Sigma_p^{-1}) + \frac{1}{\lambda}(\mu_p - \bar{w})^T\Sigma_p^{-1}(\mu_p - \bar{w}) - D + \log\left(\frac{\lambda^D \det(\Sigma_p)}{\bar{\sigma}^{2D}}\right)\right]. \tag{5}$$

A simpler expression is possible for the KL over the classifier head weights V (see App. B).

Closed-form updates. To find an optimal λ value with respect to the J_{ELBo}, notice that of the 3 additive terms in Eq. (4), only the KL term between $q(w)$ and $p(w)$ involves λ. We solve for λ by taking the gradient of the KL term with respect to λ, setting to zero, and solving, with assurances of a local maximum of J_{ELBo} via a second derivative test (see App. C). The gradient is

$$\nabla_\lambda - \mathbb{KL}(q(w)\|p(w)) = -\frac{1}{2}\left[-\frac{\bar{\sigma}^2}{\lambda^2}\operatorname{Tr}(\Sigma_p^{-1}) - \frac{1}{\lambda^2}(\mu_p - \bar{w})^T\Sigma_p^{-1}(\mu_p - \bar{w}) + \frac{D}{\lambda}\right]. \tag{6}$$

Setting $\nabla_\lambda - \mathbb{KL}(q(w)\|p(w)) = 0$ and solving for λ, we get

$$\lambda^* = \frac{1}{D}\Big[\bar{\sigma}^2\operatorname{Tr}(\Sigma_p^{-1}) + (\mu_p - \bar{w})^T\Sigma_p^{-1}(\mu_p - \bar{w})\Big]. \tag{7}$$

Similar updates can be derived for τ (see App. B).

3.3 Inflating dataset size to select complex models with better generalization

In our intended transfer learning settings with only a few 10s or 100s of labeled examples in available training data, classical statistical learning would hardly recommend deep neural network backbones with millions of parameters (recall $w \in \mathbb{R}^D$, with D very large). However, the trend in deep learning has suggested that careful fine-tuning of very large models can deliver strong performance that generalizes well [38, 45]. We find that straightforward use of the ELBo in this regime (when $D \gg N$)

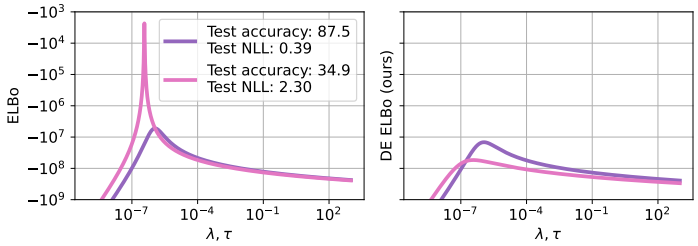

Figure 1: Model selection comparison between the ELBo (left) and our *data-emphasized ELBo* (DE ELBo) (right) for two ResNet-50s trained on CIFAR-10 $n = 1000$. For both models, we fix the estimated posterior q and vary λ, τ. **Takeaway: Without enough training data or with too many model parameters, the ELBo has a preference for simpler models.**

is overly conservative in hyperparameter selection, preferring simpler models with much-worse performance at the target classification task (see demo in Fig. 1, described further below). When N is much smaller than D, the KL terms dominate Eq. (4), overpowering the likelihood signal.

In order to make an objective that emphasizes the need for strong data fit in this overparameterized regime, we modify the usual ELBo by introducing a scaling factor $\kappa \geq 1$ on the likelihood term,

$$J_{\text{DE-ELBo}} := \kappa \mathbb{E}_{q(w,V)} \left[\log \sum_{i=1}^{N} p(y_i | w, V) \right] - \mathbb{KL}(q(w,V) \| p(w,V)) \tag{8}$$

When $D \gg N$, we recommend setting $\kappa = D/N$ to achieve an improved balance between likelihood and KL terms. In our applications, with D in millions and $N \geq 100$, this yields $\kappa \gg 1$. Of course, $\kappa = 1$ recovers the standard ELBo. We call this objective the *data-emphasized ELBo* (DE ELBo). We use it with $\kappa = D/N$ for all training steps, as well as ultimate hyperparameter/model selection.

Justification. Beyond later empirical success, we offer two arguments suggesting this revised objective as suitable for hyperparameter selection. One justification is that the function in Eq. (8) maintains a valid lower bound on the evidence $\log p(y_{1:N})$ of our observed data of size N, because as long as each y_i is a discrete random variable (in our case, a 1-of-C class label), the term we are inflating is a log PMF and thus is always negative ($\mathbb{E}_q[\log p(y_i | w, V)] \leq 0$). While the bound may be "loose" in an absolute sense, what matters more is which λ, τ values are favored. Another justification views this approach as acting as if we are modeling κN IID data instances, and we just happen to observe κ copies of the size-N dataset $y_{1:N}$ with known features $x_{1:N}$.

Demonstrating the value of $\kappa = D/N$ for improved selection. In Fig. 1, we compare the cases of $\kappa = 1$ (left panel) and $\kappa = D/N$ (right) on CIFAR-10 for L2-SP. In each plot, we compare two possible q, "A" (in pink) and "B" (in purple), across a range of λ, τ values. Version A sets $\bar{w}_A, \bar{V}_A, \bar{\sigma}_A$ to a solution favored by ELBo, with known test-set accuracy 34.9%. Version B sets $\bar{w}_B, \bar{V}_B, \bar{\sigma}_B$ to values favored by our *data-emphasized ELBo*, with known test-set accuracy of 87.5%. We see the poor-accuracy version A model is strongly favored by the conventional ELBo, while the more accurate version B model is favored by our recommended DE ELBo with $\kappa = D/N$.

Related work adjusting Bayesian objectives. Other approaches have recognized value in raising the likelihood to a power in the context of general Bayesian modeling, under the vocabulary terms of a tempered or *power likelihood* [1], *power posterior* [8, 29], or "Safe" Bayesian learning [11, 12]. However, throughout *all* these previously cited cases, the desired power is meant to be *smaller than one*, with the stated purpose of counter-acting misspecification. Values of κ larger than one (amplifying influence of data) are not even considered in these previous works, and their applications are far from our focus on transfer learning for deep neural networks. Others have recommended upweighting the KL term (not the likelihood), as in β-variational autoencoders [16], again using numerical values that diminish rather than emphasize the likelihood.

Bayesian Data Reweighting [41] learns instance-specific likelihood weights, some of which can be larger than one. However, its primary motivation is robustness and the ability to turn down the influence of observations that do not match assumptions. The authors discourage letting observations be "arbitrarily up- or down-weighted". In work with similar spirit to ours, Power sLDA artificially inflates the likelihood of class labels relative to words in supervised topic modeling applications [46].

However, they did not pursue hyperparameter selection or DNNs, as we do here, instead they focused on different models with only a few thousand parameters. Work in Bayesian deep learning has used multipliers to adjust the "temperature" of the entire log posterior (not just the likelihood), in a line of work known as *cold posteriors* [42, 20].

3.4 Overall algorithm and implementation

Ultimately, our fitting algorithm proceeds by doing stochastic gradient descent (SGD) for $\bar{w}, \bar{V}$, and $\bar{\sigma}$. We parameterize the standard deviation $\bar{\sigma} = \log(1 + \exp(\rho))$ with free parameter $\rho \in \mathbb{R}$, which ensures $\bar{\sigma}(\rho)$ is always positive during gradient descent. Before each gradient-based update step to these parameters of q, we perform closed-form updates of λ, τ as in Eq. (7); this update is the same regardless of κ. We run until a specified maximum number of iterations is reached.

Ultimately, our proposed ELBo-based method can deliver an estimated posterior q and learned hyperparameters λ, τ from one run of gradient descent. We compare to a baseline that simply performs MAP point-estimation of w, V, with a separate SGD run at each candidate λ, τ configuration in a fixed grid (see App. A.2). This "grid search" baseline is representative of cutting-edge work in transfer learning [39, 13]. In all experiments, we select ResNet-50 [15] as the backbone f_w.

Each run of our method and the baseline depends on the adequate selection of learning rate. All runs search over 4 candidate values and select the best according to either the DE ELBo (ours) or validation-set likelihood (baseline).

Both our method and the baseline can be implemented with any of the 3 settings of the backbone prior in Tab. 1. For the PTYL method [39], we use released code from Harvey et al. [13], which fixes a key issue in the original implementation so that the learned low-rank covariance Σ_p is properly scaled. We use the Woodbury matrix identity [43], trace properties, and the matrix determinant lemma to compute the trace of the inverse, squared Mahalanobis distance, and log determinant of low-rank covariance matrix Σ_p for the KL term. See App. D for details.

4 Results

Across several probabilistic modeling priors for transfer learning and 2 datasets, our findings are:

Our *data-emphasized ELBo* makes learning the regularization strength possible without compromising task accuracy. As demonstrated in Fig. 1, in regimes where the parameter dimension D is much larger than N, the ELBo has a preference for simpler models favored by the prior, potentially at the expense of downstream task accuracy. By setting $\kappa = D/N$, we upweight the likelihood terms to have reasonable importance relative to the KL terms on the ELBo. This results in an objective that can simultaneously learn backbone and classifier head weights $\bar{w}, \bar{V}$ and key hyperparameters λ, τ that are effective at downstream tasks.

The runtime of our DE ELBO is affordable, and avoids the extreme runtime costs of grid search. In Tab. 2, we show that an individual SGD run of our DE ELBO has comparable cost to one SGD run of standard MAP estimation. However, the cumulative cost of grid search needed to select λ, τ for the MAP baseline is far higher than our approach: for PTYL the recommended grid search costs over 149 hours; our approach delivers in under 3 hours.

Our *data-emphasized ELBo* achieves heldout accuracy comparable to existing approaches with far less compute time. In Fig. 2, we compare accuracy on CIFAR-10 [23] and Oxford-IIIT Pet [35] test sets over training time for L2-SP with MAP + GS and our DE ELBo. Our DE ELBo achieves comparable heldout accuracy with far less compute time. When training data is limited in size (CIFAR-10 at $N = 100$ in Fig. 2), our approach can perform even better than grid search.

In Tab. 3 and 4, we report the final accuracy on CIFAR-10 and Oxford-IIIT Pet test sets using the entire grid search compared to our DE ELBo. See App. E for NLL results.

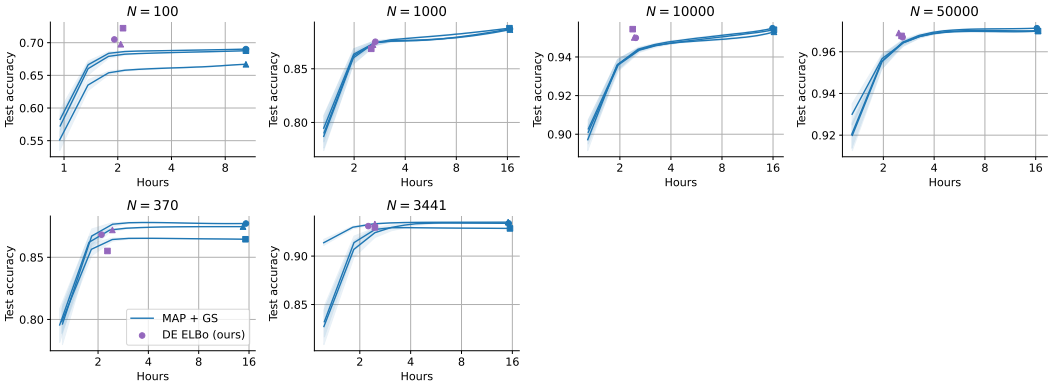

Figure 2: Test-set accuracy on CIFAR-10 (top row) and Oxford-IIIT Pet (bottom row) over training time for L2-SP with *MAP + grid search (GS)* and our *data-emphasized ELBo* (DE ELBo). We run each method on 3 separate training sets of size N (3 different marker styles). **Takeaway: Our DE ELBo achieves as good or better performance at small dataset sizes and similar performance at large dataset sizes with far less compute time.** To make the blue curves, we did the full grid search once (markers). Then, at each given shorter compute time, we subsampled a fraction of all hyperparameter configurations with that runtime and chose the best via validation NLL. Averaging this over 500 subsamples at each runtime created each blue line.

Table 2: Computational time comparison between methods for transfer learning with informative priors using grid search to find the hyperparameters that perform best on the validation set then retraining with the selected hyperparameters on the combined set of all N images (merged train and validation) and using our *data-emphasized ELBo* (DE ELBo) to learn λ, τ and select the initial learning rate (lr) on the training set for CIFAR-10 at $N = 50000$. See App. A.2 for search space details. Runtime measured on one NVIDIA A100 40GB PCIe GPU.

Model	Method	Avg. SGD runtime	lr search space	λ, τ search space	Total GS time
L2-SP	MAP + GS	39 mins. 11 secs.	4	6	16 hrs. 15 mins.
	DE ELBo	39 mins. 0 secs.	4	n/a	2 hrs. 36 mins.
PTYL	MAP + GS	37 mins. 15 secs.	4	60	149 hrs. 36 mins.
	DE ELBo	40 mins. 0 secs.	4	n/a	2 hrs. 39 mins.

Table 3: Accuracy on CIFAR-10 test set for different probabilistic models and methods. We report mean (min-max) over 3 separately-sampled training sets. For each separately-sampled training set and training set size, the *MAP + grid search (GS)* baseline requires 24 different SGD runs for L2-zero and L2-SP, and 240 for PTYL. Our *data-emphasized ELBo* (DE ELBo) requires 4 different SGD runs (one for each initial learning rate) and learns optimal λ, τ values. See App. A.2 for hyperparameter search space details.

Model	Method	$N = 100$ (10/cl.)	1000 (100/cl.)	10000 (1k/cl.)	50000 (5k/cl.)
L2-zero	MAP + GS	67.7 (66.0-68.6)	87.8 (87.5-88.4)	95.0 (94.4-95.5)	97.2 (97.1-97.2)
	DE ELBo	60.9 (58.9-63.1)	87.2 (87.0-87.4)	91.2 (90.7-92.0)	93.2 (93.0-93.3)
L2-SP	MAP + GS	68.1 (66.7-68.9)	87.3 (87.2-87.3)	95.3 (95.1-95.7)	97.1 (97.0-97.1)
	DE ELBo	70.6 (68.7-72.7)	87.2 (86.8-87.4)	95.0 (94.8-95.2)	96.8 (96.7-96.9)
PTYL	MAP + GS	67.5 (65.7-68.4)	87.9 (86.9-89.2)	95.2 (95.0-95.4)	97.3 (97.3-97.3)
	DE ELBo	70.6 (68.7-72.6)	87.2 (86.9-87.6)	95.1 (94.9-95.4)	96.9 (96.8-96.9)

5 Discussion and Conclusion

We proposed an alternative to grid search: directly learning regularization hyperparameters on the full training set via model selection techniques based on the ELBo. We showed that a modified ELBo that upweights the influence of the data likelihood relative to the prior improves model selection with limited training data or an overparameterized model. We included results on CIFAR-10 and Oxford-IIIT Pet at several data sizes that showed our DE ELBo achieves heldout accuracy comparable to existing approaches with far less compute time.

Table 4: Accuracy on Oxford-IIIT Pet test set for different probabilistic models and methods. We report mean (min-max) over 3 separately-sampled training sets. For each separately-sampled training set and training set size, the *MAP + grid search (GS)* baseline requires 24 different SGD runs for L2-zero and L2-SP, and 240 for PTYL. Our *data-emphasized ELBo* (DE ELBo) requires 4 different SGD runs (one for each initial learning rate) and learns optimal λ, τ values. See App. A.2 for hyperparameter search space details.

Model	**Method**	$N =$ **370 (10/cl.)**	**3441(93/cl.)**
L2-zero	MAP + GS	87.2 (86.6-87.5)	93.2 (93.0-93.4)
	DE ELBo	81.8 (80.3-83.6)	92.6 (92.1-92.4)
L2-SP	MAP + GS	87.2 (86.6-87.6)	93.2 (93.0-93.4)
	DE ELBo	86.6 (85.7-87.3)	93.2 (93.0-93.3)
PTYL	MAP + GS	87.1 (86.6-87.5)	92.2 (90.7-93.2)
	DE ELBo	86.6 (85.7-87.3)	93.2 (93.0-93.3)

Learning the regularization strength lets practitioners focus on other aspects that could improve accuracy more. For Oxford-IIIT Pet at $N = 370$, we found that L2-zero with an initialization pre-trained with supervised learning on ImageNet resulted in a gain of 32.4 percentage points over its self-supervised counterpart (see App. F). Our *data-emphasized ELBo* reduces compute by learning the regularization strength which enables practitioners to focus on finding pre-trained weights that generalize better and using data augmentation strategies to improve performance.

What informative prior should I use for PTYL? When using PTYL, we found that μ, Σ values obtained from supervised pre-training (minimizing cross-entropy on the source task) lead to better transfer learning (better target task accuracy) than using SimCLR [6] self-supervised learning on the source task. Consistent with Špendl and Pirc [40]'s findings on the Oxford Flowers-102 dataset [34] without hyperparameter tuning, we find that Shwartz-Ziv et al. [39]'s supervised prior performs better than their self-supervised prior on CIFAR-10 and Oxford-IIIT Pet (see App. F).

Limitations. We acknowledge our proposed approach still requires a (much smaller scale) grid search to select a learning rate. Additionally, our experiments were limited to two datasets and one backbone architecture, and only look at transfer learning rather than other possible classifier tasks. We hope that more comprehensive experiments in future work can further validate our approach.

Outlook. Our proposed approach saves practitioners time by learning an optimal regularization strength without need for expensive grid search. We hope our data-emphasized ELBo for efficient hyperparameter tuning may eventually prove useful across a wide array of classifier tasks beyond transfer learning, such as semi-supervised learning, few-shot learning, continual learning, and beyond.

Acknowledgments

Authors EH and MCH gratefully acknowledge support in part from the Alzheimer's Drug Discovery Foundation and the National Institutes of Health (grant # 1R01NS134859-01). MCH is also supported in part by the U.S. National Science Foundation (NSF) via grant IIS # 2338962. We are thankful for computing infrastructure support provided by Research Technology Services at Tufts University, with hardware funded in part by NSF award OAC CC* # 2018149.

References

[1] Isadora Antoniano-Villalobos and Stephen G. Walker. Bayesian Nonparametric Inference for the Power Likelihood. *Journal of Computational and Graphical Statistics*, 22(4):801–813, 2013. URL http://www.tandfonline.com/doi/abs/10.1080/10618600.2012.728511.

[2] Christopher M. Bishop. *Pattern Recognition and Machine Learning*, chapter 3.4 Bayesian Model Comparison. Springer, 2006.

[3] David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[4] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight Uncertainty in Neural Networks. In *International Conference on Machine Learning (ICML)*, 2015.

[5] Ciprian Chelba and Alex Acero. Adaptation of Maximum Entropy Capitalizer: Little Data Can Help a Lot. *Computer Speech & Language*, 20(4):382–399, 2006.

[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 (ICML)*, 2020.

[7] Badr-Eddine Cherief-Abdellatif. Consistency of ELBO maximization for model selection. In *Proceedings of The 1st Symposium on Advances in Approximate Bayesian Inference*, 2019. URL https://proceedings.mlr.press/v96/cherief-abdellatif19a.html.

[8] Nial Friel and Anthony N. Pettitt. Marginal Likelihood Estimation via Power Posteriors. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 70(3):589–607, 2008. URL https://academic.oup.com/jrsssb/article/70/3/589/7109555.

[9] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*, chapter 7 Regularization for Deep Learning. MIT Press, 2016. URL https://www.deeplearningbook.org/.

[10] Alex Graves. Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2011.

[11] Peter Grünwald. The Safe Bayesian: learning the learning rate via the mixability gap. In *International Conference on Algorithmic Learning Theory*, pages 169–183. Springer, 2012.

[12] Peter Grünwald and Thijs van Ommen. Inconsistency of Bayesian Inference for Misspecified Linear Models, and a Proposal for Repairing It. *Bayesian Analysis*, 12(4):1069–1103, 2017. URL https://projecteuclid.org/journals/bayesian-analysis/volume-12/issue-4/Inconsistency-of-Bayesian-Inference-for-Misspecified-Linear-Models-and-a/10.1214/17-BA1085.full.

[13] Ethan Harvey, Mikhail Petrov, and Michael C. Hughes. Transfer Learning with Informative Priors: Simple Baselines Better than Previously Reported. *Transactions on Machine Learning Research (TMLR)*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=BbvSU02jLg. Reproducibility Certification.

[14] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*, chapter Sec. 3.4 Shrinkage Methods. Springer Series in Statistics. Springer, second edition, 2009. URL https://web.stanford.edu/~hastie/ElemStatLearn/.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[16] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *International Conference on Learning Representations (ICLR)*, 2017.

[17] Geoffrey E. Hinton and Drew Van Camp. Keeping Neural Networks Simple by Minimizing the Description Length of the Weights. In *Proceedings of the sixth annual conference on Computational Learning Theory*, 1993.

[18] Harold Jeffreys. *The Theory of Probability*. The Clarendon Press, Oxford, 1939.

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

[20] Sanyam Kapoor, Wesley J. Maddox, Pavel Izmailov, and Andrew G. Wilson. On Uncertainty, Tempering, and Data Augmentation in Bayesian Classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[21] Dmitry Kobak, Jonathan Lomond, and Benoit Sanchez. The Optimal Ridge Penalty for Real-world High-dimensional Data Can Be Zero or Negative due to the Implicit Ridge Regularization. *Journal of Machine Learning Research (JMLR)*, 21(169):1–16, 2020.

[22] Ranganath Krishnan, Mahesh Subedar, and Omesh Tickoo. Efficient Priors for Scalable Variational Inference in Bayesian Deep Neural Networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, 2019.

[23] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. CIFAR-10 (Canadian Institute for Advanced Research). 2010. URL http://www.cs.toronto.edu/~kriz/cifar.html.

[24] Anders Krogh and John A. Hertz. A Simple Weight Decay Can Improve Generalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1991.

[25] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic Gradient Descent with Warm Restarts. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

[26] Sanae Lotfi, Pavel Izmailov, Gregory Benton, Micah Goldblum, and Andrew Gordon Wilson. Bayesian Model Selection, the Marginal Likelihood, and Generalization. In *International Conference on Machine Learning (ICML)*, 2022.

[27] David MacKay. Bayesian Model Comparison and Backprop Nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1991.

[28] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A Simple Baseline for Bayesian Uncertainty in Deep Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[29] Jeffrey W. Miller and David B. Dunson. Robust Bayesian Inference via Coarsening. *Journal of the American Statistical Association*, 2019. URL http://arxiv.org/abs/1506.06101.

[30] Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte Carlo Gradient Estimation in Machine Learning. *Journal of Machine Learning Research (JMLR)*, 21(132), 2020. URL http://jmlr.org/papers/v21/19-346.html.

[31] Kevin S. Murphy. *Probabilistic Machine Learning: An Introduction*, chapter 6.2.3 Example: KL divergence between two Gaussians. MIT Press, 2022.

[32] Kevin S. Murphy. *Probabilistic Machine Learning: An Introduction*, chapter 4.5.4 Picking the regularizer using a validation set. MIT Press, 2022.

[33] Kevin S. Murphy. *Probabilistic Machine Learning: An Introduction*, chapter 4.5 Regularization. MIT Press, 2022.

[34] 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 (ICVGIP)*, 2008.

[35] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. Cats And Dogs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[36] Sebastian Raschka. Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning. *arXiv preprint arXiv:1811.12808*, 2018. URL http://arxiv.org/abs/1811.12808.

[37] Tim G. J. Rudner, Xiang Pan, Yucen Lily Li, Ravid Shwartz-Ziv, and Andrew Gordon Wilson. Fine-Tuning with Uncertainty-Aware Priors Makes Vision and Language Foundation Models More Reliable. In *ICML Workshop on Structured Probabilistic Inference & Generative Modeling (SPIGM@ICML)*, 2024. URL https://openreview.net/forum?id=37fM2QEBSE.

[38] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN Features off-the-shelf: an Astounding Baseline for Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2014.

[39] Ravid Shwartz-Ziv, Micah Goldblum, Hossein Souri, Sanyam Kapoor, Chen Zhu, Yann LeCun, and Andrew G. Wilson. Pre-Train Your Loss: Easy Bayesian Transfer Learning with Informative Priors. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/b1e7f61f40d68b2177857bfcb195a507-Paper-Conference.pdf.

[40] Martin Špendl and Klementina Pirc. [Re] Easy Bayesian Transfer Learning with Informative Priors. In *ML Reproducibility Challenge 2022*, 2023. URL https://openreview.net/forum?id=JpaQ8GFOVu.

[41] Yixin Wang, Alp Kucukelbir, and David M Blei. Robust Probabilistic Modeling with Bayesian Data Reweighting. In *International Conference on Machine Learning (ICML)*, 2017. URL https://proceedings.mlr.press/v70/wang17g/wang17g.pdf.

[42] Florian Wenzel, Kevin Roth, Bastiaan S. Veeling, Jakub Świątkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin. How Good is the Bayes Posterior in Deep Neural Networks Really? In *International Conference on Machine Learning (ICML)*, 2020.

[43] Max A. Woodbury. *Inverting Modified Matrices*. Department of Statistics, Princeton University, 1950.

[44] Ming Xu, Matias Quiroz, Robert Kohn, and Scott A. Sisson. Variance Reduction Properties of the Reparameterization Trick. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[45] Li Xuhong, Yves Grandvalet, and Franck Davoine. Explicit Inductive Bias for Transfer Learning with Convolutional Networks. In *International Conference on Machine Learning (ICML)*, 2018.

[46] Cheng Zhang and Hedvig Kjellström. How to supervise topic models. In *ECCV Workshop on Graphical Models in Computer Vision*, 2014.

A Classification

A.1 Dataset details

We inlcude experiments on CIFAR-10 [23] and Oxford-IIIT Pet [35]. For both datasets, we enforce classes are uniformly distributed in the training data.

We use the same preprocessing steps for both datasets. For each distinct training set size N, we compute the mean and standard deviation of each channel to normalize images. During fine-tuning we resize the images to 256×256 pixels, perform random cropping to 224×224, and perform horizontal flips. At test time, we resize the images to 256×256 pixels and center crop to 224×224.

A.2 Classifier details

We use SGD with a Nesterov momentum parameter of 0.9 and batch size of 128 for optimization. We train for 6,000 steps using a cosine annealing learning rate [25].

For MAP + GS, we select the initial learning rate from $\{0.1, 0.01, 0.001, 0.0001\}$. For L2-zero and L2-SP we select $\frac{\alpha}{N}, \frac{\beta}{N}$ from $\{0.01, 0.001, 0.0001, \text{1e-5}, \text{1e-6}, 0.0\}$. For PTYL, we select λ from 10 logarithmically spaced values between 1e0 to 1e9 and $\frac{1}{\tau N}$ from $\{0.01, 0.001, 0.0001, \text{1e-5}, \text{1e-6}, 0.0\}$.

While tuning hyperparameters, we hold out 1/5 of the training set for validation, ensuring balanced class frequencies between sets.

After selecting the optimal hyperparameters from the validation set NLL, we retrain the model using the selected hyperparameters on the combined set of all N images (merging training and validation). All results report performance on the task in question's predefined test set.

For DE ELBo, we select the initial learning rate from $\{0.1, 0.01, 0.001, 0.0001\}$ with the highest training set J_{ELBo}. For hyperparameter/model selection (the last epoch), we average over 10 samples to get a Monte Carlo approximation of the expectation.

B Learning key hyperparameters λ, τ with the ELBo

In our particular model in Eq. (2), the KL divergence between two Gaussians [31] simplifies for the classifier head KL term as:

$$-\mathbb{KL}(q(V)\|p(V)) = -\frac{1}{2}\left[\frac{\bar{\sigma}^2}{\tau}D + \frac{1}{\tau}||\text{vec}(\bar{V})||_2^2 - D + \log\left(\frac{\tau^D}{\bar{\sigma}^{2D}}\right)\right]. \tag{9}$$

Closed-form updates To find an optimal τ value with respect to the J_{ELBo}, notice that of the 3 additive terms in Eq. (4), only the KL term between $q(V)$ and $p(V)$ involves τ. We solve for τ by taking the gradient of the KL term with respect to τ, setting to zero, and solving, with assurances of a local maximum of J_{ELBo} via a second derivative test (see App. C). The gradient is

$$\nabla_\tau - \mathbb{KL}(q(V)\|p(V)) = -\frac{1}{2}\left[-\frac{\bar{\sigma}^2}{\tau^2}D - \frac{1}{\tau^2}||\text{vec}(\bar{V})||_2^2 + \frac{D}{\tau}\right]. \tag{10}$$

Setting $\nabla_\tau - \mathbb{KL}(q(V)\|p(V)) = 0$ and solving for τ, we get

$$\tau^* = \bar{\sigma}^2 + \frac{1}{D}||\text{vec}(\bar{V})||_2^2. \tag{11}$$

C Second Derivative Test

The second derivative is

$$\begin{aligned}
\nabla_\lambda^2 - \mathbb{KL}(q(w)\|p(w)) &= -\frac{1}{2}\left[\frac{2\bar{\sigma}^2}{\lambda^3}\,\text{Tr}(\Sigma_p^{-1}) + \frac{2}{\lambda^3}(\mu_p - \bar{w})^T\Sigma_p^{-1}(\mu_p - \bar{w}) - \frac{D}{\lambda^2}\right] && (12)\\
&= -\frac{1}{2}\left[\frac{2D}{\lambda^3}\frac{1}{D}\left(\bar{\sigma}^2\,\text{Tr}(\Sigma_p^{-1}) + (\mu_p - \bar{w})^T\Sigma_p^{-1}(\mu_p - \bar{w})\right) - \frac{D}{\lambda^2}\right] && (13)\\
&= -\frac{1}{2}\left[\frac{2D}{\lambda^3}\lambda^* - \frac{D}{\lambda^2}\right]. && (14)
\end{aligned}$$

Plugging in λ^* and simplifying, we get

$$\nabla_\lambda^2 - \mathbb{KL}(q(w)\|p(w|\lambda^*)) = -\frac{D}{2}\frac{1}{\lambda^{*2}} \tag{15}$$

This expression is always negative, indicating that λ^* is a local maximum of J_{ELBo}.

The second derivative is

$$\begin{aligned}
\nabla_\tau^2 - \mathbb{KL}(q(V)\|p(V)) &= -\frac{1}{2}\left[\frac{2\bar{\sigma}^2}{\tau^3}D + \frac{2}{\tau^3}||\text{vec}(\bar{V})||_2^2 - \frac{D}{\tau^2}\right] && (16)\\
&= -\frac{1}{2}\left[\frac{2D}{\tau^3}\left(\bar{\sigma}^2 + \frac{1}{D}||\text{vec}(\bar{V})||_2^2\right) - \frac{D}{\tau^2}\right] && (17)\\
&= -\frac{1}{2}\left[\frac{2D}{\tau^3}\tau^* - \frac{D}{\tau^2}\right]. && (18)
\end{aligned}$$

Plugging in τ^* and simplifying, we get

$$\nabla_\tau^2 - \mathbb{KL}(q(V)\|p(V|\tau^*)) = -\frac{D}{2}\frac{1}{\tau^{*2}}. \tag{19}$$

This expression is always negative, indicating that τ^* is a local maximum of J_{ELBo}.

D Low-Rank Σ_p

The PTYL method [39] uses Stochastic Weight Averaging-Gaussian (SWAG) [28] to approximate the posterior distribution $p(w|\mathcal{D}_S)$ of the source data $\mathcal{D}_S$ with a Gaussian distribution $\mathcal{N}(\mu, \Sigma)$ where μ is the learned mean and $\Sigma = \frac{1}{2}(\Sigma_{\text{diag}} + \Sigma_{\text{LR}})$ is a representation of a covariance matrix with both

diagonal and *low-rank* components. The LR covariance has the form $\Sigma_{\text{LR}} = \frac{1}{K-1} QQ^T$, where $Q \in \mathbb{R}^{D \times K}$.

We use the Woodbury matrix identity [43], trace properties, and the matrix determinant lemma to compute the trace of the inverse, squared Mahalanobis distance, and log determinant of the low-rank covariance matrix for the KL term.

The trace and log determinant of the low-rank covariance matrix can be calculated once and used during training. Just like in PTYL method, the squared Mahalanobis distance needs to be re-evaluated every iteration of gradient descent.

D.1 Trace of the inverse

We compute the trace of the inverse of the low-rank covariance matrix using the Woodbury matrix identity and trace properties.

$$\begin{aligned}
\operatorname{Tr}(\Sigma_p^{-1}) &= \operatorname{Tr}((A + UCV)^{-1}) \\
&= \operatorname{Tr}(A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}) && \text{Woodbury matrix identity} \\
&= \operatorname{Tr}(A^{-1}) - \operatorname{Tr}(A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}) && \operatorname{Tr}(A+B) = \operatorname{Tr}(A) + \operatorname{Tr}(B) \\
&= \operatorname{Tr}(A^{-1}) - \operatorname{Tr}((C^{-1} + VA^{-1}U)^{-1}VA^{-1}A^{-1}U) && \operatorname{Tr}(AB) = \operatorname{Tr}(BA)
\end{aligned}$$

where $A = \frac{1}{2}\Sigma_{\text{diag}}$, $C = I_K$, $U = \frac{1}{\sqrt{2K-2}}Q$, and $V = \frac{1}{\sqrt{2K-2}}Q^T$. The last trace property, lets us compute the trace of the inverse of the low-rank covariance matrix without having to store a $D \times D$ covariance matrix.

D.2 Squared Mahalanobis distance

We compute the squared Mahalanobis distance $(\mu_p - \bar{w})^T \Sigma_p^{-1} (\mu_p - \bar{w})$ by distributing the mean difference vector into the Woodbury matrix identity.

$$\begin{aligned}
\Sigma_p^{-1} &= (A + UCV)^{-1} \\
&= (A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1}) && \text{Woodbury matrix identity}
\end{aligned}$$

D.3 Log determinant

We compute the log determinant of the low-rank covariance matrix using the matrix determinant lemma.

$$\begin{aligned}
\log\det(\Sigma_p) &= \log\det(A + UV) \\
&= \log(\det(I_K + VA^{-1}U)\det(A)) && \text{Matrix determinant lemma}
\end{aligned}$$

E NLL Results

Table 5: NLL on CIFAR-10 test set for different probabilistic models and methods. We report mean (min-max) over 3 separately-sampled training sets. For each separately-sampled training set and training set size, the *MAP + grid search (GS)* baseline requires 24 different SGD runs for L2-zero and L2-SP, and 240 for PTYL. Our *data-emphasized ELBo* (DE ELBo) requires 4 different SGD runs (one for each initial learning rate) and learns optimal λ, τ values. See App. A.2 for hyperparameter search space details.

Model	**Method**	$N =$ **100 (10/cl.)**	**1000 (100/cl.)**	**10000 (1k/cl.)**	**50000 (5k/cl.)**
L2-zero	MAP + GS	1.03 (0.96-1.10)	0.42 (0.37-0.51)	0.18 (0.16-0.19)	0.10 (0.10-0.10)
	DE ELBo	2.99 (2.67-3.17)	0.54 (0.53-0.54)	0.27 (0.25-0.29)	0.27 (0.26-0.29)
L2-SP	MAP + GS	0.97 (0.94-1.01)	0.40 (0.38-0.44)	0.15 (0.14-0.16)	0.09 (0.09-0.09)
	DE ELBo	1.07 (0.97-1.17)	0.40 (0.39-0.42)	0.26 (0.25-0.27)	0.12 (0.11-0.12)
PTYL	MAP + GS	0.98 (0.95-1.03)	0.41 (0.37-0.46)	0.16 (0.16-0.17)	0.09 (0.09-0.09)
	DE ELBo	1.06 (0.98-1.17)	0.40 (0.39-0.42)	0.26 (0.25-0.28)	0.12 (0.11-0.12)

Table 6: NLL on Oxford-IIIT Pet test set for different probabilistic models and methods. We report mean (min-max) over 3 separately-sampled training sets. For each separately-sampled training set and training set size, the *MAP + grid search (GS)* baseline requires 24 different SGD runs for L2-zero and L2-SP, and 240 for PTYL. Our *data-emphasized ELBo* (DE ELBo) requires 4 different SGD runs (one for each initial learning rate) and learns optimal λ, τ values. See App. A.2 for hyperparameter search space details.

Model	**Method**	$N =$ **370 (10/cl.)**	**3441(93/cl.)**
L2-zero	MAP + GS	0.44 (0.42-0.48)	0.25 (0.24-0.26)
	DE ELBo	0.91 (0.84-1.01)	0.27 (0.26-0.28)
L2-SP	MAP + GS	0.44 (0.42-0.48)	0.24 (0.24-0.25)
	DE ELBo	0.57 (0.55-0.61)	0.24 (0.24-0.24)
PTYL	MAP + GS	0.45 (0.42-0.48)	0.29 (0.26-0.34)
	DE ELBo	0.57 (0.55-0.61)	0.24 (0.24-0.24)

F Supervised or Self-Supervised Priors

Table 7: Accuracy on CIFAR-10 test set for different probabilistic models and methods. We report mean (min-max) over 3 separately-sampled training sets.

Model	**Method**	$N =$ **100 (10/cl.)**	**1000 (100/cl.)**	**10000 (1k/cl.)**	**50000 (5k/cl.)**
PTYL	MAP + GS	67.5 (65.7-68.4)	87.9 (86.9-89.2)	95.2 (95.0-95.4)	97.3 (97.3-97.3)
PTYL (SSL)	MAP + GS	58.7 (56.4-60.6)	83.5 (83.3-83.9)	93.8 (93.4-94.1)	96.9 (96.7-97.0)

Table 8: NLL on CIFAR-10 test set for different probabilistic models and methods. We report mean (min-max) over 3 separately-sampled training sets.

Model	**Method**	$N =$ **100 (10/cl.)**	**1000 (100/cl.)**	**10000 (1k/cl.)**	**50000 (5k/cl.)**
PTYL	MAP + GS	0.98 (0.95-1.03)	0.41 (0.37-0.46)	0.16 (0.16-0.17)	0.09 (0.09-0.09)
PTYL (SSL)	MAP + GS	1.31 (1.20-1.44)	0.58 (0.56-0.58)	0.22 (0.21-0.23)	0.10 (0.10-0.11)

Table 9: Accuracy on Oxford-IIIT Pet test set for different probabilistic models and methods. We report mean (min-max) over 3 separately-sampled training sets.

Model	**Method**	$N =$ **370 (10/cl.)**	**3441(93/cl.)**
PTYL	MAP + GS	87.1 (86.6-87.5)	92.2 (90.7-93.2)
PTYL (SSL)	MAP + GS	57.4 (56.2-58.2)	86.7 (85.0-87.8)

Table 10: NLL on Oxford-IIIT Pet test set for different probabilistic models and methods. We report mean (min-max) over 3 separately-sampled training sets.

Model	**Method**	$N =$ **370 (10/cl.)**	**3441(93/cl.)**
PTYL	MAP + GS	0.45 (0.42-0.48)	0.29 (0.26-0.34)
PTYL (SSL)	MAP + GS	1.69 (1.64-1.72)	0.52 (0.52-0.53)