Learning Hyperparameters via a Data-Emphasized Variational Objective

Ethan Harvey [1]   Mikhail Petrov [2]   Michael C. Hughes [1]

Abstract

When training large flexible models, practitioners often rely on grid search to select hyperparameters that control over-fitting. This grid search has several disadvantages: the search is computationally expensive, requires carving out a validation set that reduces the available data for training, and requires users to specify candidate values. In this paper, we propose an alternative: directly learning regularization hyperparameters on the full training set via the *evidence lower bound* ("ELBo") objective from variational methods. For deep neural networks with millions of parameters, we recommend a modified ELBo that upweights the influence of the data likelihood relative to the prior. Our proposed technique overcomes all three disadvantages of grid search. In a case study on transfer learning of image classifiers, we show how our method reduces the 88+ hour grid search of past work to under 3 hours while delivering comparable accuracy. We further demonstrate how our approach enables efficient yet accurate approximations of Gaussian processes with *learnable* length-scale kernels.

1. Introduction

When training deep neural networks (DNNs) or other flexible models, a significant amount of computational resources are devoted to tuning hyperparameters that control the trade-off between under- and over-fitting. One widespread example would be setting the strength multiplier of an additive loss term that penalizes the sum-of-squares of weight parameters. This is equivalently known in various communities as a *ridge* penalty (Hastie et al., 2009; Kobak et al., 2020), *L2 regularization* (Murphy, 2022c), or *weight decay* (Krogh & Hertz, 1991; Goodfellow et al., 2016). A common way to tune such hyperparameters is to hold out a dedicated validation set and grid search for the hyperparameter value that performs best on that set (Raschka, 2018; Murphy, 2022b).

While reasonably effective and in widespread use to manage over-fitting, 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 increases computational runtime. Second, the need to carve out a validation set 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 depending on the data and classification task at hand.

We take another approach to hyperparameter selection, inspired by a pragmatic Bayesian perspective. Suppose we model observations $\{y_i\}_{i=1}^N$ via a likelihood $p_\eta(y_{1:N}|\theta)$, where θ is a high-dimensional parameter to be estimated. We also assume a prior $p_\eta(\theta)$. Both prior and likelihood are controlled by hyperparameters η (typically just 1-5 dimensions). To effectively estimate both θ and η from data, we can follow the type-II maximum likelihood recipe: estimate the posterior $p_\eta(\theta|y_{1:N})$ while simultaneously directly learning η to maximize $p_\eta(y_{1:N}) = \int_\theta p_\eta(y_{1:N}, \theta)d\theta$. This latter objective $p_\eta(y_{1:N})$ is known as the *marginal likelihood* or *evidence* (Lotfi et al., 2022). The evidence naturally encodes a notion of Occam's razor, favoring the hyperparameter that leads to the simplest model that fits the data well, while penalizing complex models that over-fit the training data (Jeffreys, 1939; MacKay, 1991; Bishop, 2006). Learning η to maximize evidence (or equivalently, the logarithm of evidence) via gradient ascent avoids all three issues with grid search: we need only one run of gradient ascent (not separate efforts for each candidate η value in a grid), we can use all available data for training without any validation set, and we can explore the full continuous range of 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 this integral directly seems insurmountable even for a specific η, let alone optimizing evidence to select a pre-

---

[1]Department of Computer Science, Tufts University, Medford, USA [2]Department of Mechanical Engineering, Tufts University, Medford, USA. Correspondence to: Ethan Harvey <Ethan.Harvey@tufts.edu>.

Code: https://github.com/tufts-ml/data-emphasized-ELBo

ferred η value. In this work, we use and extend tools from variational Bayesian methods (Jordan et al., 1999; Blei et al., 2017), specifically tractable lower bounds on the evidence, to make *learning* hyperparameters for DNNs and other flexible models possible.

Ultimately, we contribute a general framework for learning hyperparameters that control model complexity in just one run of a gradient-based fitting algorithm, avoiding grid search for these values. We provide two case studies demonstrating success of this framework on distinct models. While our approach uses Bayesian tools, there are huge gains in runtime possible even for non-Bayesian practitioners who use our objective to learn η but then use point-estimated weights θ on downstream tasks instead of a posterior. In our transfer learning case study, we show how our approach delivers competitive accuracy while reducing total time for the grid search recommended for recent state-of-the-art methods (Xuhong et al., 2018; Shwartz-Ziv et al., 2022) from over 88 hours to under 3 hours.

2. Contributions

2.1. Problem setup: ELBo for hyperparameter selection

Consider a generic template of a probabilistic model for observed data $\{y_i\}_{i=1}^N$, governed by high-dimensional parameters $\theta \in \mathbb{R}^D$. We assume a typical factorization structure:

$$p_\eta(y_{1:N}, \theta) = p_\eta(\theta) \cdot \prod_{i=1}^N p_\eta(y_i|\theta). \quad (1)$$

This template is instantiated by specifying a concrete likelihood $p_\eta(y_i|\theta)$ and prior $p_\eta(\theta)$. The subscript of each indicates possible dependence on hyperparameter vector η.

The Bayesian approach to fit this model to data involves estimating the posterior distribution $p_\eta(\theta|y_{1:N})$, yet this is typically intractable. Instead, a popular approach is to pursue an approximate posterior via variational methods (Blei et al., 2017; Jordan et al., 1999). We first select an "easy-to-use" family of distributions $q_r(\theta)$ over the parameter θ. Each concrete value of parameter r defines a specific distribution over θ. We then pose an optimization problem: find the variational parameter r that makes $q_r(\theta)$ as close as possible to the true (intractable) posterior.

A tractable way to do this is to estimate r by maximizing an objective function known as the *evidence lower bound* ("ELBo", (Blei et al., 2017)), denoted as J_{ELBo} and defined for our model p and approximate posterior q as $J_{\text{ELBo}} :=$

$$\mathbb{E}_{q_r(\theta)} \left[\sum_{i=1}^N \log p_\eta(y_i|\theta) \right] - \mathbb{KL}(q_r(\theta) \| p_\eta(\theta)). \quad (2)$$

This objective is a function of data $y_{1:N}$, variational parameters r, and hyperparameters η. Maximizing J_{ELBo} can

be shown equivalent to finding the specific q that is "closest" to the true posterior in the sense of Kullback-Leibler (KL) divergence. Further, as the name of the objective suggests, we can show mathematically that $J_{\text{ELBo}}(y_{1:N}, r, \eta) \leq \log \int_\theta p_\eta(y_{1:N}, \theta) d\theta$. That is, the ELBo is a lower bound on the log marginal likelihood or *evidence*. The evidence is itself a function of η.

The bound relation underlying the ELBo suggests its potential utility for data-driven selection of hyperparameters η. Indeed, past work has used the ELBo to select hyperparameters in several contexts (Ueda & Ghahramani, 2002; Damianou & Lawrence, 2013). Recent theoretical results (Cherief-Abdellatif, 2019) argue that using ELBo for model selection enjoys strong theoretical guarantees on quality, even under misspecification. Unfortunately, practical efforts remain dominated by grid search (Osawa et al., 2019; Shwartz-Ziv et al., 2022) rather than gradient-of-ELBo learning of hyperparameters.

In a similar line of work, Immer et al. (2021) use Laplace's approximation to estimate the marginal likelihood for hyperparameter selection. While effective, this approach relies on expensive approximations of the Hessian, which are at least $\mathcal{O}(D^2 N)$, and an inner loop to learn hyperparameters.

2.2. Contribution: Data-emphasized ELBo

Consider applications of very flexible models where dataset size N is limited relative to the parameter size D. In such cases, where $D \gg N$, classical arguments from statistical learning hardly favor large models due to the risk of overfitting; despite this, well-trained flexible models often enjoy practical success (Sharif Razavian et al., 2014; Xuhong et al., 2018). Our work offers two core contributions to improve hyperparameter tuning in such applications.

First, we point out that straightforward use of the ELBo in the $D \gg N$ regime can be *overly conservative* in hyperparameter selection. Selected hyperparameters often prefer simpler models with far worse performance at the target prediction task. See our own brief demo in Fig. 1, as well as our two later case studies. This finding is corroborated by Blundell et al. (2015), who reportedly tried to learn hyperparameters of Bayesian neural nets via gradients of the ELBo, but found it "not be useful, and yield worse results."

Second, we suggest a modified ELBo objective that can overcome this issue by better emphasizing the data likelihood. Concretely, we suggest this *data-emphasized* ELBo objective $J_{\text{DE ELBo}} :=$

$$\kappa \cdot \mathbb{E}_{q_r(\theta)} \left[\sum_{i=1}^N \log p_\eta(y_i|\theta) \right] - \mathbb{KL}(q_r(\theta) \| p_\eta(\theta)). \quad (3)$$

where we have introduced a scaling factor κ on the likelihood term. When using the standard ELBo ($\kappa = 1$) in
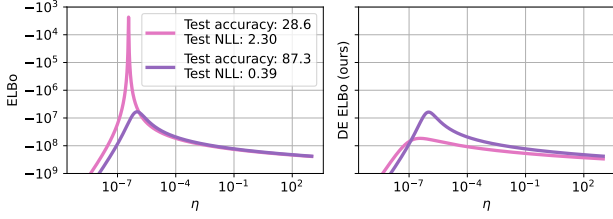
Figure 1: Model selection comparison between the ELBo (left) and our *data-emphasized ELBo* (DE ELBo, right) for two different approximate posteriors q defined in Case Study B: ResNet-50s with D in millions trained on CIFAR-10 with $N = 1000$. Each panel varies hyperparameters $\eta = \lambda = \tau$. **Takeaway: When $D \gg N$, the ELBo prefers simpler q (pink), while our DE ELBo favors q that produce higher test accuracy (purple).**

our target applications, we find the KL term comparing distributions over the high-dimensional random variable θ dominates the overall objective. By setting $\kappa > 1$, our approach emphasizes data fit more and overcomes this imbalance. Concretely, in all our case studies we recommend setting $\kappa = D/N$ to achieve an improved balance between the expected likelihood and KL terms.

This objective with varying κ has been used in previous work on *tempering* the variational ELBo (Mandt et al., 2016; Aitchison, 2021; Pitas & Arbel, 2024). This past work was not directly motivated by concerns about hyperparameter tuning as we are. Instead, they sought to improve the posterior's downstream generalization performance or counter misspecification in the probabilistic model. See Sec. 2.3 for a thorough discussion.

Justification. Beyond later empirical success across two case studies, we offer two arguments suggesting this revised objective as suitable for hyperparameter selection. One justification that applies to all models in our framework views this approach as acting as if we are modeling κN *i.i.d.* data instances, and we just happen to observe κ copies of the size-N dataset $y_{1:N}$. Under these assumptions, the ELBo of this replicated data is exactly the DE ELBo of the original data. Another justification applies only when each y_i is a discrete random variable, such as a 1-of-C class label. In such cases, the log likelihood upweighted by $\kappa > 1$ is a log PMF and thus always negative ($\mathbb{E}_q[\log p(y_i|\theta)] \leq 0$). Thus, the function in Eq. (3) is still a valid lower bound on the evidence $\log p(y_{1:N})$. While the bound may be "loose" in an absolute sense, what matters more is which hyperparameter values η it favors.

Demo: Improved selection with $\kappa = D/N$. In Fig. 1, we compare $\kappa = 1$ (left panel) and $\kappa = D/N$ (right) for models defined later in Case Study B: deep neural net classifiers originally trained on ImageNet that are transferred to the CIFAR-10 classification task. In each plot, we compare two possible q, colored *pink* and *purple*, across a range of

η values. The pink version fixes neural net parameters and variational parameters to a solution favored by ELBo, with known test-set accuracy 28.6%. The purple version instead optimizes these parameters to other values via the *data-emphasized ELBo*, with known test-set accuracy of 87.3%. We see the poor-accuracy model is strongly favored by the conventional ELBo ($\kappa=1$), while our DE ELBo ($\kappa=D/N$) prefers the more accurate model.

2.3. Related work adjusting Bayesian objectives

Upweighting data. The work most similar in spirit to ours also considers objectives that upweight the likelihood term of the ELBo, or equivalently downweight the KL term. This line of work (Pitas & Arbel, 2024; Aitchison, 2021; Osawa et al., 2019; Zhang et al., 2018), which refers to such reweighting as *tempering*, pursues variational methods for deep neural net classifiers in order to reap the benefits of modeling uncertainty well. Yet these works miss the opportunity to use the modified ELBo to *learn* hyperparameters efficiently. There is an awareness that it is "favorable to tune regularization" (Zhang et al., 2018), yet often only a few candidate values of prior variances are tried in Osawa et al. (2019, Fig. 8) or Zhang et al. (2018), perhaps due to large costs of each separate run. Aitchison (2021) keep regularization hyperparameters fixed with their tempered ELBo objectives. Pitas & Arbel (2024) claim in their supplement to "optimize the prior variance using the marginal likelihood", yet the exact marginal likelihood of neural net classifiers is intractable and their work lacks reproducible details about how this is done or whether it is effective.

Downweighting data. Other work also under the *tempering* name downweights the likelihood in the ELBo. Mandt et al. (2016) develop a *variational tempering* method which effectively downweights the likelihood in the ELBo in order to pursue the goal of avoiding local optima in the complex posteriors arising in mixture and topic models. That work also does not learn hyperparameters as we do.

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 *power likelihood* (Antoniano-Villalobos & Walker, 2013), *power posterior* (Friel & Pettitt, 2008; Miller & Dunson, 2019), or "Safe" Bayesian learning (Grünwald, 2012; Grünwald & van Ommen, 2017). However, these works focus on setting the likelihood power *smaller than one*, with the stated purpose of counter-acting misspecification. Values of κ larger than one (amplifying influence of data, as we do) are not considered. Work on β-variational autoencoders (Higgins et al., 2017) recommends upweighting the KL term of the ELBo, again effectively diminishing rather than emphasizing data.

Other work. Other work in Bayesian deep learning has recommended *cold posteriors* (Wenzel et al., 2020; Kapoor

et al., 2022), which used multipliers to adjust the "temperature" of the entire log posterior, not just the likelihood.

Bayesian Data Reweighting (Wang et al., 2017) learns instance-specific likelihood weights, some of which can be larger than one and thus emphasize data. However, that work's primary motivation is robustness, seeking to downweight 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 inflates the likelihood of class label data relative to word data in supervised topic modeling (Zhang & Kjellström, 2014). However, they focused on models of much smaller size D that differ from the neural nets we consider, and did not learn hyperparameters.

Naming. To avoid potential confusion over which terms are upweighted or downweighted under past work labeled tempering, we prefer to call our objective the *data-emphasized ELBo*. This name focuses on what term we upweight. It also reminds readers what matters to us is an *objective for hyperparameter estimation*. While users could benefit from downstream use of estimated posteriors, we hope a primary use case is for non-Bayesian practitioners who ultimately just want point-estimated weights with good η. In practice, point-estimated weights often outperform Bayesian posteriors at test-set accuracy (Pitas & Arbel, 2024).

3. Case Study A: Random Fourier features

To motivate this first case study, recall the Gaussian process (GP) (Rasmussen & Williams, 2006b) as a flexible model for regression or classification. A common choice for GPs is to select a *radial basis function* (RBF) kernel $k(x, x') = \nu^2 \exp\left(-\frac{\|x-x'\|_2^2}{2\gamma^2}\right)$, where $\gamma > 0$ determines the length-scale and $\nu > 0$ the output-scale. This GP model allows decision function complexity to elegantly grow with N, the number of training instances. However, a primary downside to GPs is scalability, with classic fitting algorithms scaling cubically with N (Rasmussen & Williams, 2006b). As a possible remedy, we consider a *random Fourier feature* (RFF) (Rahimi & Recht, 2007) approach, which posits an alternative weight-space model that scales *linearly* in N during fitting yet approximates a GP. The approximation quality increases with a user-controlled size parameter R, yet R increases the overall parameter dimension D.

GPs are notoriously sensitive to hyperparameters (Rasmussen & Williams, 2006a). RFFs are also sensitive, yet tuning hyperparameters well has been largely overlooked. Rahimi & Recht (2007) just fix $\gamma = 1, \nu = 1$. When Liu et al. (2020; 2023) used RFFs to build distance-aware neural net classifiers, they recommend $\gamma = 2$ (though released code sets $\gamma = \sqrt{20}$) and tune ν on a heldout validation set via expensive search.

We intend to show how fitting this RFF weight-space model via a data-emphasized ELBo yields effective and affordable hyperparameter learning of γ, ν in the $D \gg N$ regime. Better hyperparameter learning translates to notable gains in downstream prediction quality, better matching the ideal GP on small datasets (as in Fig. 2) but scaling better for large N. Our work here could be used as an efficient (*linear in* N) approximation of GPs on large datasets, or as a drop-in modification of Liu et al.'s distance-aware neural nets to enable even better downstream performance.

3.1. Model A definition

We assume a training set of N pairs x_i, y_i of feature vector $x_i \in \mathbb{R}^H$ and corresponding class label $y_i \in \{1, 2, \dots C\}$. We define a simple linear regression/classification model using RFFs of user-defined size R. First, draw random values that define RFF weights $A \in \mathbb{R}^{H \times R}$ and $b \in \mathbb{R}^R$; once drawn they are fixed throughout training and prediction. Next, use fixed A, b and a cosine function to non-linearly map each x_i to a representation vector $\phi(x_i) \in \mathbb{R}^R$. Formally, these two steps are:

$$A_{h,r} \sim \mathcal{N}(0, 1), \ b_r \sim \text{Unif}(0, 2\pi) \text{ for all } r, h. \quad (4)$$

$$\phi(x_i) = \nu \sqrt{\frac{2}{R}} \cos\left(\frac{1}{\gamma} A^T x_i + b\right). \quad (5)$$

To complete the model, a linear layer with weights $V \in \mathbb{R}^{C \times R}$ maps $\phi(x_i)$ to predicted regression targets or logit-probabilities over C classes.

Contribution: RFF for arbitrary length-scale. Our featurization in Eq. (5) generalizes the classic construction of RFFs by Rahimi & Recht (2007) to any length-scale $\gamma > 0$ and output-scale $\nu > 0$. In the appendix, we prove that our RFF construction allows a Monte Carlo approximation of the RBF kernel. That is, for any pair of feature vectors we have $\phi(x_i)^T \phi(x_j) \approx k(x_i, x_j)$, where k is an RBF kernel whose γ, ν values match those used to construct ϕ in Eq. (5). The quality of this approximation increases with R, typically $100 < R < 10000$. Past work has proven this when $\gamma=1, \nu=1$ (Rahimi & Recht, 2007); however, to the best of our knowledge our use of RFF with an arbitrary γ is potentially novel. We later show how to *learn* γ, ν with substantial practical utility.

Point estimation view. To fit the RFF model to observed data, an empirical risk minimization strategy would find the value of weights V that minimizes the loss function $L(V) :=$

$$\sum_{i=1}^N \ell(y_i, V\phi(x_i)) + \frac{\tau^{-1}}{2} ||\text{vec}(V)||_2^2, \quad (6)$$

where ℓ is an appropriate loss function for prediction quality (e.g., mean squared error, cross entropy). The sum-of-squares (L2) penalty on V encourages lower magnitude
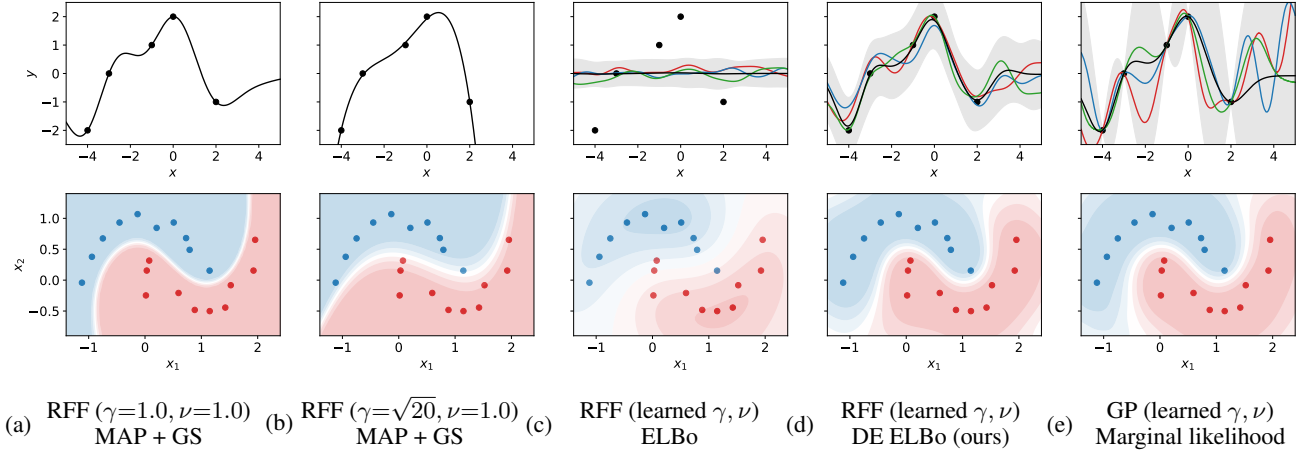
Figure 2: **Case Study A: Demo of hyperparameter sensitivity and selection for RFF models.** The first four columns use the RFF regression/classification model in Sec. 3.1, varying estimation and selection techniques. The last column shows the reference fit of a GP, a gold-standard for this toy data but less scalable. Columns a and b show MAP point estimation, with fixed γ, ν and grid search (GS) for τ (separate GD run for each value). Columns c and d use variational methods, *learning* γ, ν, τ in one run of GD. For the regression dataset, we plot the mean function of y over samples from q. Our DE ELBo objective enables the best approximation of the GP.

weights to avoid overfitting on the high-dimensional features. Ultimately, model quality is impacted by 3 key hyperparameters: length-scale $\gamma > 0$, output-scale $\nu > 0$, and L2-penalty strength $\tau > 0$. None of these can be effectively set using the training loss alone.

Bayesian view. A Bayesian interpretation of the RFF problem assumes a joint probabilistic model $p(V, y_{1:N}) = p(V) \prod_i p(y_i|V)$, with factors for the classification case

$$p(V) = \mathcal{N}(\text{vec}(V)|0, \tau I), \tag{7}$$

$$p(y_i|V) = \text{CatPMF}(y_i|\text{SM}(V\phi(x_i))). \tag{8}$$

For regression, instead of the softmax function SM we'd just model y_i as normally distributed with mean $V\phi(x_i)$. In either case, $\tau > 0$ is a hyperparameter controlling under/overfitting. Maximum a-posteriori (MAP) estimation of V recovers the objective in Eq. (6) when we set ℓ to $-\log p(y_i|V)$. To fit this model into our general framework, we have $\theta = \{V\}$ and $\eta = \{\gamma, \nu, \tau\}$.

3.2. Variational methods for Model A

To apply the general variational recipe described in Sec. 2 to model A, we first select an approximate posterior over parameter V. For simplicity, we chose a Normal with unknown mean and isotropic covariance.

$$q(V) = \mathcal{N}(\text{vec}(V)|\text{vec}(\bar{V}), \bar{\sigma}^2 I). \tag{9}$$

Here, the free parameters that define q are $r = \{\bar{V}, \bar{\sigma}\}$.

3.3. Implementation for Model A

Learning τ. To learn the prior precision τ, we make use of closed-form expressions for the KL divergence between

two Gaussians. We can solve for the optimal τ given other parameters and hyperparameters (see App. D).

Learning ν, γ. To learn other hyperparameters, no such closed-form update exists. We employ first-order gradient descent as the gradient of each ELBo-based objective J is easily computed with respect to ν and γ via automatic differentiation. We reparameterize to handle the positivity constraints with the invertible softplus mapping, so that all gradient descent is done on unconstrained parameters.

Training. For this case study A, our toy datasets are small enough that we perform gradient descent without minibatching. At each step of our ELBo or DE ELBo methods, we update variational parameters r and hyperparameters γ, ν via a gradient step and then update τ to its closed-form optimal value. We run for a specified number of iterations, verifying convergence by manual inspection.

Competitors. We compare to a baseline that performs MAP point-estimation of V, with a separate gradient descent run at each candidate τ configuration in a fixed grid and fix γ, ν to previously recommended values. Given only small toy data, we pick the τ that delivers best train-set likelihood.

3.4. Evaluation for Model A

In Fig. 2, we compare the decision functions of different modeling pipelines on a simple univariate regression dataset inspired by Fig. 2.2 in Rasmussen & Williams (2006b) (top row) and the *two moons* classification task (bottom). The goal here is to illustrate sensitivity to hyperparameters $\eta = \{\gamma, \nu, \tau\}$ and the effective learning of η enabled by our approach. Both dataset sizes are rather small ($N = 5$ on top, $N = 20$ on bottom). For all RFFs, we set $R = 1024$,

so the effective dimension is $D = 1024$.

For reference, the last column of the figure shows an ideal GP with hyperparameters tuned to optimize marginal likelihood (exact in the regression case, approximate for classification via a Dirichlet-based GP (Milios et al., 2018)). The GP is a "gold standard" for these toy tasks, yet the GP does not scale well to larger N. While RFF models are in the $D \gg N$ regime we intend for our DE ELBo, the GP's natural formulation is not.

Results. First, results in columns 1-2 of Fig. 2 show that hyperparameters matter. Past work on RFFs (Liu et al., 2020) recommended $\gamma = \sqrt{20}$, yet we see here that different γ are preferred across the top and bottom datasets.

Next, we find the posteriors and hyperparameters estimated with our DE ELBo (column 4) result in better approximations of the gold-standard GP (column 5). Using the conventional ELBo (column 3), the chosen hyperparameters produce decision functions that *underfit*, because the KL term dominates the unweighted likelihood when $D \gg N$.

4. Case Study B: Transfer learning

For case study B, we investigate transfer learning of image classifiers using informative priors (Xuhong et al., 2018; Shwartz-Ziv et al., 2022).

4.1. Model B definition

Consider training a neural network classifier composed of two parts. First, a backbone encoder f with weights $w \in \mathbb{R}^D$, 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). Second, a linear-decision-boundary classifier head with weights $V \in \mathbb{R}^{C \times H}$, 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 Xuhong et al. (2018)) would pursue empirical risk minimization with an L2-penalty on weight magnitudes for regularization, training to minimize the loss function $L(w, V) :=$

$$\sum_{i=1}^{N} \ell(y_i, V f_w(x_i)) + \frac{\alpha}{2} ||w||_2^2 + \frac{\beta}{2} ||\text{vec}(V)||_2^2 \quad (10)$$

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:

$$p(w) = \mathcal{N}(w|\mu_p, \lambda \Sigma_p), \quad (11)$$

$$p(V) = \mathcal{N}(\text{vec}(V)|0, \tau I), \quad (12)$$

$$p(y_i|w, V) = \text{CatPMF}(y_i|\text{SM}(V f_w(x_i))). \quad (13)$$

Here, $\lambda > 0, \tau > 0$ are hyperparameters controlling under/over-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. Pursing MAP estimation of both w and V recovers the objective in Eq. (10) when we set $\alpha = \frac{1}{\lambda}, \beta = \frac{1}{\tau}, \mu_p = 0, \Sigma_p = I$, and ℓ to $-\log p(y_i|w, V)$. To fit this model into our general framework, we have $\theta = \{w, V\}$ and $\eta = \{\lambda, \tau\}$.

Need for validation set and grid search. Selecting α, β (or equivalently λ, τ) to directly minimize Eq. (10) 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 priors. Several recent transfer learning approaches correspond to specific settings of the mean and covariance μ_p, Σ_p of the backbone prior $p(w)$. Let μ represent a specific pre-trained vector of backbone weights that performs well on a source task. Setting $\mu_p=0, \Sigma_p=I$ recovers a conventional approach to transfer learning, which we call L2-zero, where regularization pushes backbone weights to zero. The pretrained value μ only informs the initial value of backbone weights w before SGD (Xuhong et al., 2018; Harvey et al., 2024). Instead, setting the prior mean as $\mu_p = \mu$ along with $\Sigma_p = I$ recovers *L2 starting point* (L2-SP) regularization (Xuhong et al., 2018), also suggested by Chelba & Acero (2006). Further setting Σ_p to the estimated covariance matrix Σ of a Gaussian approximation of the posterior over

Table 1: Possible priors.

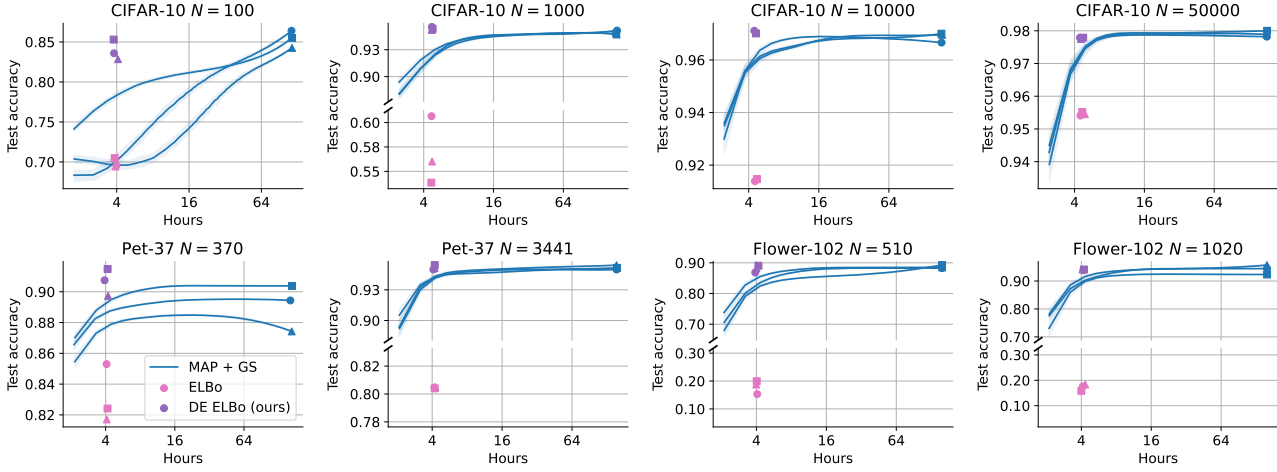| Method	$p(w)$	Init.
L2-zero	$\mathcal{N}(0, \lambda I)$	μ
L2-SP	$\mathcal{N}(\mu, \lambda I)$	μ
PTYL	$\mathcal{N}(\mu, \lambda \Sigma)$	μ

Figure 3: Test-set accuracy on CIFAR-10, Pet-37, and Flower-102 over total runtime for L2-SP with *MAP + grid search (GS)* and our *data-emphasized ELBo* (DE ELBo) using ConvNeXt-Tiny (see ViT-B/16 in Fig. G.1 and ResNet-50 in Fig. G.2). 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.

backbone weights for the source task recovers the "Pre-Train Your Loss" (PTYL) method (Shwartz-Ziv et al., 2022).

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: PTYL's creators recommended large values from 1e0 to 1e10 (Shwartz-Ziv et al., 2022). Later works search far smaller values (1e-5 to 1e-3) (Rudner et al., 2025).

4.2. Variational methods for Model B

To apply the general variational recipe described in Sec. 2 to model B, we first select an approximate posterior over parameters w, V. For simplicity, we chose a factorized Normal with unknown means and isotropic covariance.

$$q(w, V) = q(w)q(V), \tag{14}$$

$$q(w) = \mathcal{N}(w|\bar{w}, \bar{\sigma}^2 I), \tag{15}$$

$$q(V) = \mathcal{N}(\text{vec}(V)|\text{vec}(\bar{V}), \bar{\sigma}^2 I). \tag{16}$$

Here, the free parameters that define q are $r = \{\bar{w}, \bar{V}, \bar{\sigma}\}$.

To fit to data, we estimate r and η values that optimize the ELBo or the DE ELBo. To evaluate the KL term in each expression, we use the closed-form available because both prior and q are Gaussian. To evaluate the expected log likelihood term, we use Monte Carlo averaging of S samples from q (Xu et al., 2019; Mohamed et al., 2020). We further use the *reparameterization trick* to obtain gradient

estimates of $\nabla_r J$ (Blundell et al., 2015). We find using *just one sample* per training step is sufficient and most efficient.

4.3. Implementation for Model B

Learning λ, τ. To learn the prior precisions λ, τ, we make use of the closed-form expression for the KL term in our ELBo objectives. For example, setting $\nabla_\lambda J = 0$ and solving for λ, we get

$$\lambda^* = \frac{1}{D}\left[\bar{\sigma}^2 \text{Tr}(\Sigma_p^{-1}) + (\mu_p - \bar{w})^T \Sigma_p^{-1}(\mu_p - \bar{w})\right] \tag{17}$$

with assurances of a local maximum of $\nabla_\lambda J$ via a second derivative test (see App. E). Similar updates can be derived for τ (see App. D).

Training. For this case study B, we use stochastic gradient descent (SGD) with batches of 128 to fit to large datasets. At each step of the algorithm, we update variational parameters r via a gradient step, and then update λ, τ to its closed-form optimal values as in Eq. (17). 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).

Estimating ELBo. Given a fit q, we estimate the expected log likelihood term of the ELBo or DE ELBo for final selection decisions by averaging over 10 samples from q.

Evaluating accuracy. To evaluate classification accuracy for a q, we find that just plugging in the estimated posterior mean $\bar{w}, \bar{V}$ to make predictions on heldout images achieves similar accuracy to averaging over 10 posterior samples

Table 2: Computational time comparison between methods for transfer learning with informative priors using *MAP + grid search (GS)* and our *data-emphasized ELBo* (DE ELBo) for CIFAR-10 $N = 50000$. See App. C.2 for search space details. Runtime measured on four Intel Xeon Gold 6226R CPUs (2.90 GHz) and one NVIDIA A100 GPU (40 GB).

Model	Method	lr search space	λ, τ search space	ResNet-50		ViT-B/16		ConvNeXt-Tiny	
				Avg. SGD runtime	Total GS time	Avg. SGD runtime	Total GS time	Avg. SGD runtime	Total GS time
L2-SP	MAP + GS	4	36	36 mins. 37 secs.	88 hrs. 29 mins.	2 hrs. 16 mins.	328 hrs. 56 mins.	1 hrs. 15 mins.	181 hrs. 53 mins.
	DE ELBo	4	n/a	32 mins. 44 secs.	2 hrs. 10 mins.	2 hrs. 22 mins.	9 hrs. 29 mins.	1 hrs. 6 mins.	4 hrs. 2 mins.
PTYL	MAP + GS	4	60	37 mins. 2 secs.	148 hrs. 43 mins.	n/a	n/a	n/a	n/a
	DE ELBo	4	n/a	35 mins. 55 secs.	2 hrs. 23 mins.	n/a	n/a	n/a	n/a

Table 3: Accuracy on CIFAR-10, Pet-37, and Flower-102 test sets for different probabilistic models, methods, and backbones. We report mean (min-max) over 3 separately-sampled training sets. The *MAP + grid search (GS)* baseline requires 24 different SGD runs for L2-zero, 144 for L2-SP, and 240 for PTYL, while our *data-emphasized ELBo* (DE ELBo) requires 4 different SGD runs.

Model	Method	CIFAR-10				Pet-37		Flower-102	
		$N = 100$ (10/cl.)	1000 (100/cl.)	10000 (1k/cl.)	50000 (5k/cl.)	370 (10/cl.)	3441(93/cl.)	510 (5/cl.)	1020 (10/cl.)
					ResNet-50				
L2-zero	MAP + GS	67.7 (66.0-69.2)	87.7 (87.2-88.2)	94.6 (93.8-95.1)	97.0 (96.9-97.1)	87.6 (86.7-88.4)	93.1 (92.9-93.4)	86.4 (86.0-86.7)	92.8 (92.4-93.2)
	DE ELBo	62.0 (60.7-63.4)	87.4 (87.0-87.8)	91.3 (90.9-91.7)	93.0 (92.9-93.4)	83.3 (83.2-83.3)	91.6 (91.6-91.7)	81.2 (81.0-81.4)	89.9 (89.3-90.5)
L2-SP	MAP + GS	69.0 (65.9-72.3)	88.7 (88.3-89.4)	95.4 (95.3-95.6)	97.3 (97.2-97.4)	88.0 (87.1-88.8)	92.9 (92.6-93.1)	86.1 (85.3-86.7)	93.2 (93.1-93.3)
	DE ELBo	69.9 (67.4-71.9)	88.5 (86.9-89.5)	95.1 (95.0-95.2)	96.8 (96.7-96.9)	87.5 (87.2-87.8)	92.9 (92.7-93.0)	84.5 (84.5-84.6)	91.8 (91.6-92.1)
PTYL	MAP + GS	70.1 (69.2-71.4)	89.8 (89.5-90.3)	95.6 (95.5-95.8)	97.0 (96.8-97.2)	87.9 (87.5-88.2)	93.0 (92.8-93.2)	86.3 (85.8-86.6)	92.9 (92.6-93.1)
	DE ELBo	70.0 (67.9-72.1)	89.2 (89.0-89.5)	95.1 (94.9-95.4)	96.9 (96.8-97.0)	87.5 (87.3-87.8)	92.9 (92.7-93.0)	84.6 (84.5-84.6)	91.8 (91.7-92.0)
					ViT-B/16				
L2-SP	MAP + GS	86.5 (85.3-87.7)	93.8 (93.3-94.0)	97.2 (97.1-97.2)	98.2 (98.2-98.2)	88.1 (87.1-89.4)	93.3 (93.2-93.5)	85.8 (84.9-86.6)	91.4 (88.3-93.0)
	DE ELBo	88.5 (87.1-89.8)	94.1 (94.0-94.2)	97.4 (97.3-97.6)	98.2 (98.1-98.3)	88.9 (88.3-89.5)	93.6 (93.5-93.7)	80.5 (78.0-84.0)	89.2 (88.5-89.6)
					ConNeXt-Tiny				
L2-SP	MAP + GS	85.4 (84.3-86.4)	94.1 (93.9-94.3)	96.9 (96.7-97.0)	97.9 (97.8-98.0)	89.1 (87.4-90.4)	94.3 (94.2-94.5)	88.7 (88.3-89.3)	94.2 (92.3-95.7)
	DE ELBo	83.9 (82.9-85.3)	94.5 (94.4-94.6)	97.1 (97.0-97.1)	97.8 (97.8-97.8)	90.7 (89.7-91.5)	94.3 (94.2-94.5)	88.0 (86.8-89.0)	94.1 (93.9-94.2)

without the added computational cost.

Competitors. We compare to a baseline that simply performs MAP point-estimation of w, V. This executes a separate SGD run at each candidate λ, τ configuration in a fixed grid (see App. C.2). We select the best according to the validation-set likelihood. This "grid search" baseline is representative of cutting-edge work in transfer learning (Shwartz-Ziv et al., 2022; Harvey et al., 2024).

Both our method and the MAP baseline can be implemented with any of the 3 settings of the backbone prior in Tab. 1. For the PTYL method (Shwartz-Ziv et al., 2022), we use released code from Harvey et al. (2024), 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 (Woodbury, 1950), 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.

4.4. Results for Model B

Across several possible priors for transfer learning, backbone architectures, and datasets, our findings are:

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 148 hours; our approach delivers in under 3 hours.

Our DE ELBo achieves heldout accuracy comparable to existing approaches with far less compute time. In Fig. 3, we compare test set accuracy on CIFAR-10 (Krizhevsky et al., 2010), Oxford-IIIT Pet (Parkhi et al., 2012), and Oxford Flower (Nilsback & Zisserman, 2008) images 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 (Pet-37 $N = 370$ in Fig. 3), our approach can perform even better than grid search. In Tab. 3, we report test-set accuracies for both expensive grid search and our (faster) DE ELBo. See App. H for more results.

5. Discussion and Conclusion

We proposed an alternative to grid search: directly learning hyperparameters that control model complexity 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. Experiments for transfer learning examine 3 architectures (ResNet, ViT, and ConvNexT) and 3 distinct datasets (CIFAR-10, Pet-37, and Flower-102), each at several sizes. Results showed our DE ELBo achieves heldout accuracy comparable to existing approaches with far less compute time.

Learning hyperparameters lets practitioners focus on other aspects that could improve accuracy more. Our DE ELBo reduces compute time by directly learning hyperparameters. With dozens of hours of saved time, practitioners can focus on other efforts that improve target task performance, such as better data augmentation or trying other pretrained weights. For example, on Pet-37 $N = 370$, we found that L2-zero using weights pre-trained with supervised learning on ImageNet resulted in a gain of 32.4 percentage points in accuracy over an alternative *self-supervised* pretraining (see Tab. H.1).

Our DE ELBo would benefit from further exploration of how κ values impact model performance. Beyond saving practioners valuable time, we hope our work sparks interest in theoretical understanding of how modified ELBos can form a compelling basis for hyperparameter learning.

Acknowledgments

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. Our work has the potential to reduce the energy consumption required for machine learning research and applications.

References

Aitchison, L. A statistical theory of cold posteriors in deep neural networks. In *International Conference on Learning Representations (ICLR)*, 2021.

Antoniano-Villalobos, I. and Walker, S. G. 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.

Bishop, C. M. *Pattern Recognition and Machine Learning*, chapter 3.4 Bayesian Model Comparison. Springer, 2006.

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

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight Uncertainty in Neural Networks. In *International Conference on Machine Learning (ICML)*, 2015.

Burda, Y., Grosse, R., and Salakhutdinov, R. Importance Weighted Autoencoders. In *International Conference on Learning Representations (ICLR)*, 2016.

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

Cherief-Abdellatif, B.-E. 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.

Damianou, A. and Lawrence, N. Deep Gaussian Processes. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2013. URL http://proceedings.mlr.press/v31/damianou13a.html.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations (ICLR)*, 2021.

Friel, N. and Pettitt, A. N. 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.

Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*, chapter 7 Regularization for Deep Learning. MIT Press, 2016. URL https://www.deeplearningbook.org/.

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

Grünwald, P. The Safe Bayesian: learning the learning rate via the mixability gap. In *International Conference on Algorithmic Learning Theory*, pp. 169–183. Springer, 2012.

Grünwald, P. and van Ommen, T. 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.

Harvey, E., Petrov, M., and Hughes, M. C. 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.

Hastie, T., Tibshirani, R., and Friedman, J. *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/.

He, K., Zhang, X., Ren, S., and Sun, J. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. In *International Conference on Learning Representations (ICLR)*, 2017.

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

Immer, A., Bauer, M., Fortuin, V., Rätsch, G., and Emtiyaz, K. M. Scalable Marginal Likelihood Estimation for Model Selection in Deep Learning. In *International Conference on Machine Learning (ICML)*, 2021.

Jeffreys, H. *The Theory of Probability*. The Clarendon Press, Oxford, 1939.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. An Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37:183–233, 1999.

Kapoor, S., Maddox, W. J., Izmailov, P., and Wilson, A. G. On Uncertainty, Tempering, and Data Augmentation in Bayesian Classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Kobak, D., Lomond, J., and Sanchez, B. 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.

Krishnan, R., Subedar, M., and Tickoo, O. 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.

Krizhevsky, A., Nair, V., and Hinton, G. CIFAR-10 (Canadian Institute for Advanced Research). 2010. URL http://www.cs.toronto.edu/~kriz/cifar.html.

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

Liu, J., Lin, Z., Padhy, S., Tran, D., Bedrax Weiss, T., and Lakshminarayanan, B. Simple and Principled Uncertainty Estimation with Deterministic Deep Learning via Distance Awareness. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Liu, J. Z., Padhy, S., Ren, J., Lin, Z., Wen, Y., Jerfel, G., Nado, Z., Snoek, J., Tran, D., and Lakshminarayanan, B. A Simple Approach to Improve Single-Model Deep Uncertainty via Distance-Awareness. *Journal of Machine Learning Research (JMLR)*, 24(42):1–63, 2023.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

Loshchilov, I. and Hutter, F. SGDR: Stochastic Gradient Descent with Warm Restarts. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2016.

Lotfi, S., Izmailov, P., Benton, G., Goldblum, M., and Wilson, A. G. Bayesian Model Selection, the Marginal Likelihood, and Generalization. In *International Conference on Machine Learning (ICML)*, 2022.

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

Maddox, W. J., Izmailov, P., Garipov, T., Vetrov, D. P., and Wilson, A. G. A Simple Baseline for Bayesian Uncertainty in Deep Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Mandt, S., McInerney, J., Abrol, F., Ranganath, R., and Blei, D. Variational tempering. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2016.

Milios, D., Camoriano, R., Michiardi, P., Rosasco, L., and Filippone, M. Dirichlet-based gaussian processes for large-scale calibrated classification. *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

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

Mohamed, S., Rosca, M., Figurnov, M., and Mnih, A. 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.

Murphy, K. S. *Probabilistic Machine Learning: An Introduction*, chapter 6.2.3 Example: KL divergence between two Gaussians. MIT Press, 2022a.

Murphy, K. S. *Probabilistic Machine Learning: An Introduction*, chapter 4.5.4 Picking the regularizer using a validation set. MIT Press, 2022b.

Murphy, K. S. *Probabilistic Machine Learning: An Introduction*, chapter 4.5 Regularization. MIT Press, 2022c.

Nilsback, M.-E. and Zisserman, A. Automated Flower Classification over a Large Number of Classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing (ICVGIP)*, 2008.

Osawa, K., Swaroop, S., Khan, M. E. E., Jain, A., Eschenhagen, R., Turner, R. E., and Yokota, R. Practical Deep Learning with Bayesian Principles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Parkhi, O. M., Vedaldi, A., Zisserman, A., and Jawahar, C. Cats And Dogs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

Pitas, K. and Arbel, J. The Fine Print on Tempered Posteriors. In *Asian Conference on Machine Learning*, 2024.

Rahimi, A. and Recht, B. Random Features for Large-Scale Kernel Machines. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2007.

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

Rasmussen, C. E. and Williams, C. K. *Gaussian Processes for Machine Learning*, chapter 2.3 Varying the Hyperparameters. The MIT Press, 2006a.

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

Rudner, T. G. J., Pan, X., Li, Y. L., Shwartz-Ziv, R., and Wilson, A. G. 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)*, 2025. URL https://openreview.net/forum?id=37fM2QEBSE.

Sharif Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. 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.

Shwartz-Ziv, R., Goldblum, M., Souri, H., Kapoor, S., Zhu, C., LeCun, Y., and Wilson, A. G. 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.

Ueda, N. and Ghahramani, Z. Bayesian Model Search for Mixture Models Based on Optimizing Variational Bounds. *Neural Networks*, 15(1):1223–1241, 2002.

Wang, Y., Kucukelbir, A., and Blei, D. M. Robust Probabilistic Modeling with Bayesian Data Reweighting. In *International Conference on Machine Learning (ICML)*, 2017. URL https://proceedings.mlr.press/v70/wang17g/wang17g.pdf.

Wenzel, F., Roth, K., Veeling, B. S., Świątkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. How Good is the Bayes Posterior in Deep Neural Networks Really? In *International Conference on Machine Learning (ICML)*, 2020.

Woodbury, M. A. *Inverting Modified Matrices*. Department of Statistics, Princeton University, 1950.

Xu, M., Quiroz, M., Kohn, R., and Sisson, S. A. Variance Reduction Properties of the Reparameterization Trick. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

Xuhong, L., Grandvalet, Y., and Davoine, F. Explicit Inductive Bias for Transfer Learning with Convolutional Networks. In *International Conference on Machine Learning (ICML)*, 2018.

Zhang, C. and Kjellström, H. How to supervise topic models. In *ECCV Workshop on Graphical Models in Computer Vision*, 2014.

Zhang, G., Sun, S., Duvenaud, D., and Grosse, R. Noisy Natural Gradient as Variational Inference. In *International Conference on Machine Learning (ICML)*, 2018.

Appendix of
Learning Hyperparameters via a Data-Emphasized Variational Objective

A. Lenth-Scale and Output-Scale

RFFs are typically used with an RBF kernel where $\gamma = 1, \nu = 1$. Below we show that our RFF construction allows a Monte Carlo approximation of the RBF kernel for any $\gamma > 0, \nu > 0$. Our proof extends base case material found in the blogpost *Random Fourier Features*: https://gregorygundersen.com/blog/2019/12/23/random-fourier-features/.

Recall that $x \in \mathbb{R}^H$, $A \in \mathbb{R}^{H \times R}$ and $b \in \mathbb{R}^R$. Fill b so each entry is $b_r \sim \text{Unif}(0, 2\pi)$. Fill A so each entry is $A_{hr} \sim \mathcal{N}(0, 1)$.

$$\phi(x)^T \phi(x') = \frac{\nu^2}{R} \sum_{r=1}^{R} 2\cos\left(\frac{1}{\gamma} A_{:,r}^T x + b_r\right) \cos\left(\frac{1}{\gamma} A_{:,r}^T x' + b_r\right) \tag{18}$$

$$= \frac{\nu^2}{R} \sum_{r=1}^{R} \cos\left(\frac{1}{\gamma} A_{:,r}^T (x + x') + 2b_r\right) + \cos\left(\frac{1}{\gamma} A_{:,r}^T (x - x')\right) \quad \text{Sum of angles formula} \tag{19}$$

$$= \frac{\nu^2}{R} \sum_{r=1}^{R} \cos\left(\frac{1}{\gamma} A_{:,r}^T (x - x')\right) \quad \begin{array}{l}\text{Expectation of } \cos \text{ over its period is} \\ \text{zero}\end{array} \tag{20}$$

$$= \frac{\nu^2}{R} \sum_{r=1}^{R} \cos\left(\frac{1}{\gamma} A_{:,r}^T (x - x')\right) + i\sin\left(\frac{1}{\gamma} A_{:,r}^T (x - x')\right) \quad \begin{array}{l}\text{Add imaginary part, which is zero} \\ \text{in expectation, as } \mathbb{E}[\sin(a)] = 0 \\ \text{for any zero-mean Gaussian r.v. } a\end{array} \tag{21}$$

$$= \frac{\nu^2}{R} \sum_{r=1}^{R} \exp\left(i\frac{1}{\gamma} A_{:,r}^T (x - x')\right) \quad \text{Euler's formula} \tag{22}$$

The above sum over R samples is an unbiased estimate of the expectation of a complex exponential with respect to a vector $\omega \in \mathbb{R}^H$ where each entry is drawn $\omega_h \sim \mathcal{N}(0, \frac{1}{\gamma^2})$ for all h. Using the reparametrization trick, we can sample a draw ω as $\frac{1}{\gamma} A_{:,r}$ where $A_{h,r} \sim \mathcal{N}(0, 1)$ for all h. Then we have:

$$\frac{\nu^2}{R} \sum_{r=1}^{R} \exp\left(i\frac{1}{\gamma} A_{:,r}^T (x - x')\right) \approx \nu^2 \mathbb{E}_{\mathcal{N}(\omega|0, \frac{1}{\gamma^2} I_H)}[\exp(i\omega^T (x - x'))]. \tag{23}$$

In the remainder, we define a vector $\Delta = x - x'$ for simplicity, where $\Delta \in \mathbb{R}^H$. Next, we show that this expectation of a complex exponential is equivalent to the RBF kernel $k(\Delta)$ whose lengthscale γ and outputscale ν values match those used

to construct ϕ.

$$\nu^2 \mathbb{E}_{p(\omega)}[\exp(i\omega^T\Delta)] \qquad\qquad p(\omega) = \mathcal{N}(\omega|0, \tfrac{1}{\gamma^2}I_H) \qquad (24)$$

$$= \nu^2 \int p(\omega) \exp\left(i\omega^T\Delta\right) d\omega \qquad\qquad \text{Expectation as integral} \qquad (25)$$

$$= \nu^2 \int \left(\frac{\gamma^2}{2\pi}\right)^{H/2} \exp\left(-\frac{\gamma^2\omega^T\omega}{2}\right) \exp\left(i\omega^T\Delta\right) d\omega \qquad\qquad \text{Definition of } p(\omega) \qquad (26)$$

$$= \nu^2 \left(\frac{\gamma^2}{2\pi}\right)^{H/2} \int \exp\left(-\frac{\gamma^2\omega^T\omega}{2} + i\omega^T\Delta\right) d\omega \qquad\qquad \text{Product rule of exponents} \qquad (27)$$

$$= \nu^2 \left(\frac{\gamma^2}{2\pi}\right)^{H/2} \int \exp\left(-\frac{\gamma^2\omega^T\omega}{2} + i\omega^T\Delta + \frac{\Delta^T\Delta}{2\gamma^2} - \frac{\Delta^T\Delta}{2\gamma^2}\right) d\omega \qquad \text{Add and subtract same term} \quad (28)$$

$$= \nu^2 \left(\frac{\gamma^2}{2\pi}\right)^{H/2} \exp\left(-\frac{\Delta^T\Delta}{2\gamma^2}\right) \int \exp\left(-\frac{\gamma^2}{2}\left(\omega - \frac{i}{\gamma^2}\Delta\right)^T \left(\omega - \frac{i}{\gamma^2}\Delta\right)\right) d\omega \quad \text{Completing the square} \quad (29)$$

$$= \nu^2 \left(\frac{\gamma^2}{2\pi}\right)^{H/2} \exp\left(-\frac{\Delta^T\Delta}{2\gamma^2}\right) \left(\frac{2\pi}{\gamma^2}\right)^{H/2} \qquad\qquad \begin{array}{l}\text{Translation invariance of} \\ \text{Gaussian integral}\end{array} \qquad (30)$$

$$= \nu^2 \exp\left(-\frac{\Delta^T\Delta}{2\gamma^2}\right) \qquad\qquad (31)$$

$$= k(\Delta) \qquad\qquad \text{By definition of RBF kernel} \quad (32)$$

B. Importance Weighted Evidence

There are two possible explanations for why our *data-emphasized ELBo* favors q that produce higher test accuracy: 1) our *data-emphasized* objective addresses looseness in the ELBo and 2) our *data-emphasized* objective addresses a misspecified prior. To demonstrate the latter is true, we use importance weighted evidence (IWE) (Burda et al., 2016) to approximate the logarithm of evidence. IWE is a better estimator of the logarithm of evidence than the ELBo and a should approach the logarithm of evidence as the number of samples goes to infinity.

$$\log p_\eta(y_{1:N}) = \log \mathbb{E}_{q_r(\theta)}\left[\frac{p_\eta(y_{1:N}|\theta)p_\eta(\theta)}{q_r(\theta)}\right] \geq \mathbb{E}_{q_r(\theta)}\left[\log \frac{p_\eta(y_{1:N}|\theta)p_\eta(\theta)}{q_r(\theta)}\right] = J_{\text{ELBo}} \qquad (33)$$

$$J_{\text{IWE}} := \log \mathbb{E}_{q_r(\theta)}\left[\exp\left(\sum_{i=1}^N \log p_\eta(y_i|\theta) + \log p_\eta(\theta) - \log q_r(\theta)\right)\right] \qquad (34)$$

However, just like the ELBo, when $D \gg N$ the IWE prefers simpler q, while our *data-emphasized IWE* (DE IWE) favors q that produce higher test accuracy. This suggests that our *data-emphasized* objective addresses a misspecified prior rather than looseness in the ELBo.

$$J_{\text{DE IWE}} := \log \mathbb{E}_{q_r(\theta)}\left[\exp\left(\kappa \cdot \sum_{i=1}^N \log p_\eta(y_i|\theta) + \log p_\eta(\theta) - \log q_r(\theta)\right)\right] \qquad (35)$$

C. Classification

C.1. Dataset details

We include experiments on CIFAR-10 (Krizhevsky et al., 2010), Oxford-IIIT Pet (Parkhi et al., 2012), and Oxford Flower (Nilsback & Zisserman, 2008). For all experiments, we randomly select N training images, stratifying by class to ensure balanced class frequencies.

We use the same preprocessing steps for all 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, randomly crop images to 224×224, and randomly flip images horizontal with probability 0.5. At test time, we resize the images to 256×256 pixels and center crop to 224×224.
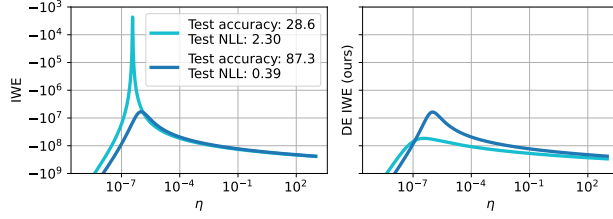
Figure B.1: Model selection comparison between the importance weighted evidence (IWE, left) and our *data-emphasized IWE* (DE IWE, right) for two different approximate posteriors q defined in Case Study B: ResNet-50s with D in millions trained on CIFAR-10 with $N = 1000$. Each panel varies hyperparameters $\eta = \lambda = \tau$. **Takeaway: When $D \gg N$, the IWE prefers simpler q (cyan), while our DE IWE favors q that produce higher test accuracy (blue).**

C.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 (Loshchilov & Hutter, 2016).

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

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 $\{0.1, 0.01, 0.001, 0.0001\}$ and learn optimal λ, τ values.

D. Learning λ, τ

In our particular model in Eq. (11), the KL divergence between two Gaussians (Murphy, 2022a) simplifies for the backbone KL term as:

$$-\mathbb{KL}(q(w)\|p(w)) = -\frac{1}{2}\left[\frac{\bar{\sigma}^2}{\lambda}\text{Tr}(\Sigma_p^{-1}) + \frac{1}{\lambda}(\mu_p - \bar{w})^T\Sigma_p^{-1}(\mu_p - \bar{w}) - D + \log\left(\frac{\lambda^D\det(\Sigma_p)}{\bar{\sigma}^{2D}}\right)\right]. \tag{36}$$

Closed-form updates. To find an optimal λ value with respect to the J_{ELBo}, notice that of the 3 additive terms in Eq. (3), 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. E). The gradient is

$$\nabla_\lambda -\mathbb{KL}(q(w)\|p(w)) = -\frac{1}{2}\left[-\frac{\bar{\sigma}^2}{\lambda^2}\text{Tr}(\Sigma_p^{-1}) - \frac{1}{\lambda^2}(\mu_p - \bar{w})^T\Sigma_p^{-1}(\mu_p - \bar{w}) + \frac{D}{\lambda}\right]. \tag{37}$$

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

$$\lambda^* = \frac{1}{D}\left[\bar{\sigma}^2\text{Tr}(\Sigma_p^{-1}) + (\mu_p - \bar{w})^T\Sigma_p^{-1}(\mu_p - \bar{w})\right]. \tag{38}$$

In our particular model in Eq. (11), the KL divergence between two Gaussians (Murphy, 2022a) simplifies for the classifier head KL term as:

$$-\mathbb{KL}(q(V)\|p(V)) = -\frac{1}{2}\left[\frac{\bar{\sigma}^2}{\tau}D + \frac{1}{\tau}\|\text{vec}(\bar{V})\|_2^2 - D + \log\left(\frac{\tau^D}{\bar{\sigma}^{2D}}\right)\right]. \tag{39}$$

Closed-form updates To find an optimal τ value with respect to the J_{ELBo}, notice that of the 3 additive terms in Eq. (3), 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. E). The gradient is

$$\nabla_\tau - \mathbb{KL}(q(V)\|p(V)) = -\frac{1}{2}\left[-\frac{\bar{\sigma}^2}{\tau^2}D - \frac{1}{\tau^2}\|\text{vec}(\bar{V})\|_2^2 + \frac{D}{\tau}\right]. \tag{40}$$

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

$$\tau^* = \bar{\sigma}^2 + \frac{1}{D}\|\text{vec}(\bar{V})\|_2^2. \tag{41}$$

E. Second Derivative Test

The second derivative is

$$\nabla_\lambda^2 - \mathbb{KL}(q(w)\|p(w)) = -\frac{1}{2}\left[\frac{2\bar{\sigma}^2}{\lambda^3}\text{Tr}(\Sigma_p^{-1}) + \frac{2}{\lambda^3}(\mu_p - \bar{w})^T\Sigma_p^{-1}(\mu_p - \bar{w}) - \frac{D}{\lambda^2}\right] \tag{42}$$

$$= -\frac{1}{2}\left[\frac{2D}{\lambda^3}\frac{1}{D}\left(\bar{\sigma}^2\text{Tr}(\Sigma_p^{-1}) + (\mu_p - \bar{w})^T\Sigma_p^{-1}(\mu_p - \bar{w})\right) - \frac{D}{\lambda^2}\right] \tag{43}$$

$$= -\frac{1}{2}\left[\frac{2D}{\lambda^3}\lambda^* - \frac{D}{\lambda^2}\right]. \tag{44}$$

Plugging in λ^* and simplifying, we get

$$\nabla_\lambda^2 - \mathbb{KL}(q(w)\|p(w|\lambda^*)) = -\frac{D}{2}\frac{1}{\lambda^{*2}} \tag{45}$$

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

The second derivative is

$$\nabla_\tau^2 - \mathbb{KL}(q(V)\|p(V)) = -\frac{1}{2}\left[\frac{2\bar{\sigma}^2}{\tau^3}D + \frac{2}{\tau^3}\|\text{vec}(\bar{V})\|_2^2 - \frac{D}{\tau^2}\right] \tag{46}$$

$$= -\frac{1}{2}\left[\frac{2D}{\tau^3}\left(\bar{\sigma}^2 + \frac{1}{D}\|\text{vec}(\bar{V})\|_2^2\right) - \frac{D}{\tau^2}\right] \tag{47}$$

$$= -\frac{1}{2}\left[\frac{2D}{\tau^3}\tau^* - \frac{D}{\tau^2}\right]. \tag{48}$$

Plugging in τ^* and simplifying, we get

$$\nabla_\tau^2 - \mathbb{KL}(q(V)\|p(V|\tau^*)) = -\frac{D}{2}\frac{1}{\tau^{*2}}. \tag{49}$$

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

F. Low-Rank Σ_p

The PTYL method (Shwartz-Ziv et al., 2022) uses Stochastic Weight Averaging-Gaussian (SWAG) (Maddox et al., 2019) 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 (Woodbury, 1950), 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. Like in the PTYL method, the squared Mahalanobis distance needs to be re-evaluated every iteration of gradient descent.

F.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.

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

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.

F.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.

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

F.3. Log determinant

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

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

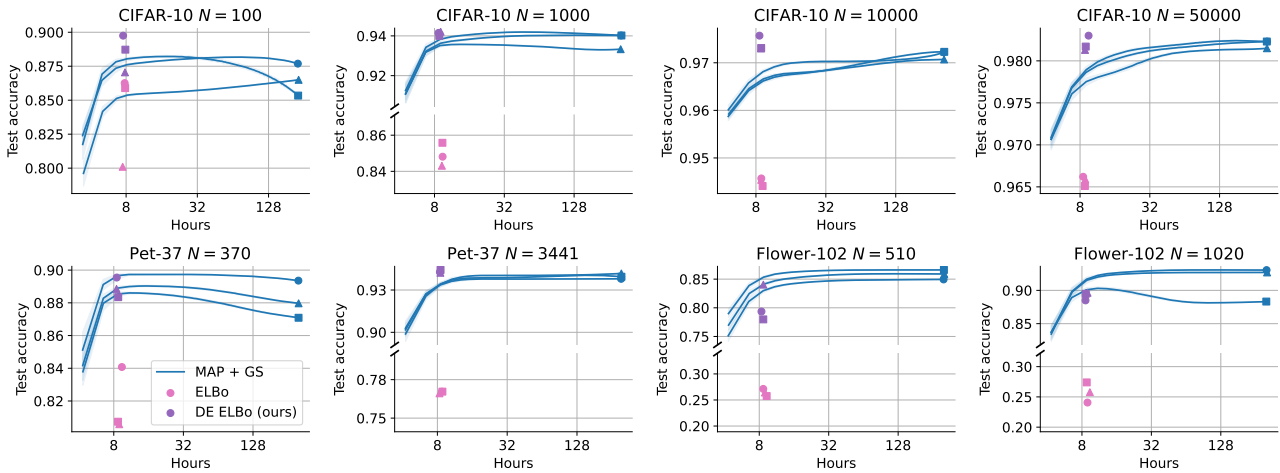# G. Computational Time Comparison

Figure G.1: Test-set accuracy on CIFAR-10, Pet-37, and Flower-102 over total runtime for L2-SP with *MAP + grid search (GS)* and our *data-emphasized ELBo* (DE ELBo) using ViT-B/16. 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.
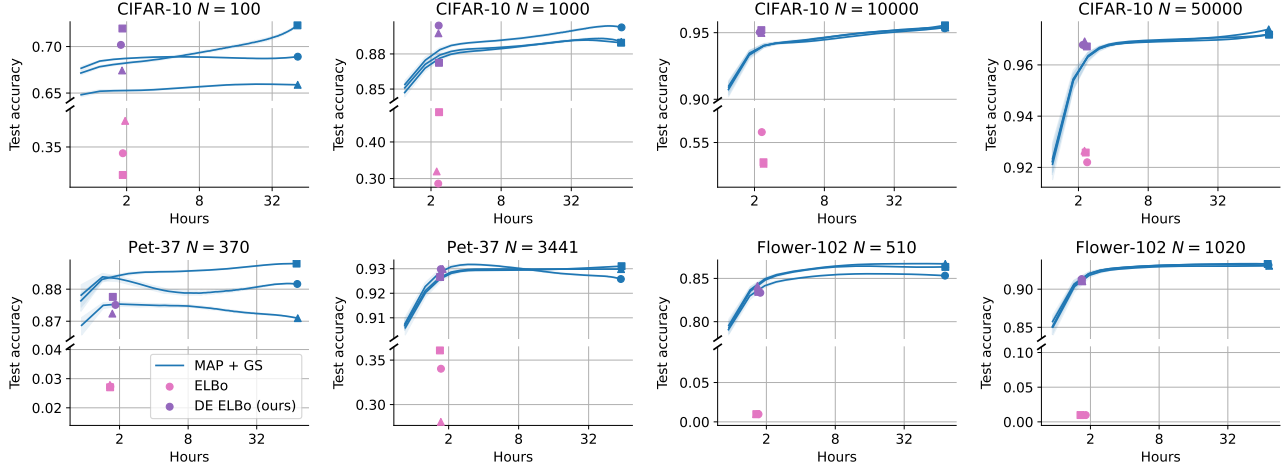
Figure G.2: Test-set accuracy on CIFAR-10, Pet-37, and Flower-102 over total runtime for L2-SP with *MAP + grid search (GS)* and our *data-emphasized ELBo* (DE ELBo) using ResNet-50. We run each method on 3 separate training sets of size N (3 different marker styles). **Takeaway: Our DE ELBo achieves 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 H.1: Accuracy on CIFAR-10, Pet-37, and Flower-102 test sets for different probabilistic models, methods, and backbones. 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, 144 for 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. C.2 for hyperparameter search space details.

Model	Method	CIFAR-10				Pet-37		Flower-102	
		$N = 100$ (10/cl.)	1000 (100/cl.)	10000 (1k/cl.)	50000 (5k/cl.)	370 (10/cl.)	3441(93/cl.)	510 (5/cl.)	1020 (10/cl.)
ResNet-50									
Linear probing	MAP + GS	60.1 (59.3-61.2)	74.5 (74.2-74.8)	81.4 (81.3-81.4)	83.2 (83.1-83.2)	85.9 (85.2-86.5)	90.8 (90.8-90.8)	80.6 (79.5-81.5)	87.9 (87.9-88.1)
	DE ELBo	58.4 (57.8-58.9)	73.6 (73.2-74.1)	81.6 (81.5-81.8)	83.2 (83.1-83.3)	85.6 (85.0-85.9)	90.7 (90.6-90.9)	80.8 (79.6-81.6)	88.0 (87.8-88.2)
L2-zero	MAP + GS	67.7 (66.0-69.2)	87.7 (87.2-88.2)	94.6 (93.8-95.1)	97.0 (96.9-97.1)	87.6 (86.7-88.4)	93.1 (92.9-93.4)	86.4 (86.0-86.7)	92.8 (92.4-93.2)
	DE ELBo	62.0 (60.7-63.4)	87.4 (87.0-87.8)	91.3 (90.9-91.7)	93.0 (92.9-93.4)	83.3 (83.2-83.3)	91.6 (91.6-91.7)	81.2 (81.0-81.4)	89.9 (89.3-90.5)
L2-SP	MAP + GS	69.0 (65.9-72.3)	88.7 (88.3-89.4)	95.4 (95.3-95.6)	97.3 (97.2-97.4)	88.0 (87.1-88.8)	92.9 (92.6-93.1)	86.1 (85.3-86.7)	93.2 (93.1-93.3)
	DE ELBo	69.9 (67.4-71.9)	88.5 (86.9-89.5)	95.1 (95.0-95.2)	96.8 (96.7-96.9)	87.5 (87.2-87.8)	92.9 (92.7-93.0)	84.5 (84.5-84.6)	91.8 (91.6-92.1)
PTYL (SSL)	MAP + GS	57.5 (56.1-58.6)	78.4 (77.8-79.0)	90.6 (90.1-90.8)	96.6 (96.5-96.6)	58.3 (55.6-60.7)	86.3 (85.9-86.8)	81.9 (80.9-82.8)	89.7 (89.2-90.1)
	DE ELBo	60.2 (59.5-60.7)	78.1 (77.5-78.8)	90.6 (90.3-90.8)	96.7 (96.6-96.7)	56.6 (56.1-57.2)	80.1 (80.0-80.3)	76.8 (76.5-77.2)	84.9 (84.6-85.1)
PTYL	MAP + GS	70.1 (69.2-71.4)	89.8 (89.5-90.3)	95.6 (95.5-95.8)	97.0 (96.8-97.2)	87.9 (87.5-88.2)	93.0 (92.8-93.2)	86.3 (85.8-86.6)	92.9 (92.6-93.1)
	DE ELBo	70.0 (67.9-72.1)	89.2 (89.0-89.5)	95.1 (94.9-95.4)	96.9 (96.8-97.0)	87.5 (87.3-87.8)	92.9 (92.7-93.0)	84.6 (84.5-84.6)	91.8 (91.7-92.0)
ViT-B/16									
L2-SP	MAP + GS	86.5 (85.3-87.7)	93.8 (93.3-94.0)	97.2 (97.1-97.2)	98.2 (98.2-98.2)	88.1 (87.1-89.4)	93.3 (93.2-93.5)	85.8 (84.9-86.6)	91.4 (88.3-93.0)
	DE ELBo	88.5 (87.1-89.8)	94.1 (94.0-94.2)	97.4 (97.3-97.6)	98.2 (98.1-98.3)	88.9 (88.3-89.5)	93.6 (93.5-93.7)	80.5 (78.0-84.0)	89.2 (88.5-89.6)
ConNeXt-Tiny									
L2-SP	MAP + GS	85.4 (84.3-86.4)	94.1 (93.9-94.3)	96.9 (96.7-97.0)	97.9 (97.8-98.0)	89.1 (87.4-90.4)	94.3 (94.2-94.5)	88.7 (88.3-89.3)	94.2 (92.3-95.7)
	DE ELBo	83.9 (82.9-85.3)	94.5 (94.4-94.6)	97.1 (97.0-97.1)	97.8 (97.8-97.8)	90.7 (89.7-91.5)	94.3 (94.2-94.5)	88.0 (86.8-89.0)	94.1 (93.9-94.2)

Table H.2: NLL on CIFAR-10, Pet-37, and Flower-102 test sets for different probabilistic models, methods, and backbones. 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, 144 for 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. C.2 for hyperparameter search space details.

Model	Method	CIFAR-10 $N = 100$ (10/cl.)	1000 (100/cl.)	10000 (1k/cl.)	50000 (5k/cl.)	Pet-37 370 (10/cl.)	3441(93/cl.)	Flower-102 510 (5/cl.)	1020 (10/cl.)
					ResNet-50				
Linear probing	MAP + GS	1.22 (1.20-1.25)	0.75 (0.75-0.76)	0.55 (0.54-0.55)	0.48 (0.48-0.48)	0.46 (0.42-0.52)	0.30 (0.30-0.30)	0.83 (0.81-0.86)	0.53 (0.53-0.53)
	DE ELBo	1.89 (1.87-1.90)	1.06 (1.03-1.09)	0.55 (0.54-0.55)	0.48 (0.48-0.48)	0.51 (0.46-0.55)	0.39 (0.39-0.39)	0.83 (0.81-0.86)	0.53 (0.53-0.53)
L2-zero	MAP + GS	0.97 (0.94-1.03)	0.41 (0.39-0.44)	0.19 (0.19-0.20)	0.10 (0.10-0.10)	0.42 (0.37-0.47)	0.24 (0.24-0.25)	0.65 (0.63-0.68)	0.35 (0.32-0.38)
	DE ELBo	2.71 (2.51-2.85)	0.53 (0.50-0.55)	0.28 (0.27-0.29)	0.28 (0.27-0.29)	0.81 (0.78-0.84)	0.27 (0.26-0.28)	0.86 (0.79-0.91)	0.45 (0.42-0.46)
L2-SP	MAP + GS	0.94 (0.85-1.02)	0.40 (0.39-0.41)	0.15 (0.15-0.16)	0.09 (0.09-0.09)	0.41 (0.36-0.45)	0.27 (0.24-0.30)	0.65 (0.63-0.68)	0.33 (0.32-0.35)
	DE ELBo	1.01 (0.92-1.12)	0.44 (0.42-0.46)	0.24 (0.23-0.24)	0.12 (0.12-0.12)	0.51 (0.49-0.53)	0.24 (0.24-0.24)	0.71 (0.67-0.74)	0.38 (0.38-0.39)
PTYL (SSL)	MAP + GS	1.34 (1.20-1.41)	0.50 (0.49-0.52)	0.23 (0.22-0.24)	0.11 (0.11-0.11)	1.64 (1.55-1.68)	0.54 (0.50-0.56)	0.84 (0.81-0.86)	0.49 (0.46-0.51)
	DE ELBo	1.36 (1.33-1.43)	0.76 (0.74-0.78)	0.29 (0.28-0.31)	0.12 (0.12-0.12)	2.07 (2.04-2.08)	0.68 (0.67-0.69)	1.10 (1.05-1.15)	0.72 (0.70-0.74)
PTYL	MAP + GS	0.90 (0.87-0.94)	0.36 (0.34-0.37)	0.15 (0.14-0.16)	0.10 (0.10-0.10)	0.42 (0.37-0.46)	0.26 (0.21-0.31)	0.65 (0.63-0.68)	0.33 (0.32-0.34)
	DE ELBo	1.00 (0.92-1.08)	0.46 (0.46-0.47)	0.23 (0.22-0.24)	0.12 (0.12-0.12)	0.51 (0.49-0.53)	0.24 (0.24-0.24)	0.71 (0.67-0.74)	0.38 (0.38-0.39)
					ViT-B/16				
L2-SP	MAP + GS	0.46 (0.41-0.51)	0.25 (0.20-0.28)	0.10 (0.10-0.10)	0.06 (0.06-0.06)	0.42 (0.38-0.47)	0.25 (0.24-0.29)	0.69 (0.64-0.72)	0.42 (0.34-0.54)
	DE ELBo	0.36 (0.32-0.41)	0.24 (0.23-0.26)	0.12 (0.12-0.13)	0.07 (0.07-0.07)	0.40 (0.38-0.42)	0.30 (0.30-0.31)	0.90 (0.73-1.03)	0.51 (0.48-0.55)
					ConNeXt-Tiny				
L2-SP	MAP + GS	0.46 (0.44-0.50)	0.22 (0.19-0.24)	0.11 (0.10-0.12)	0.07 (0.07-0.07)	0.38 (0.30-0.46)	0.18 (0.18-0.18)	0.54 (0.51-0.56)	0.28 (0.23-0.38)
	DE ELBo	0.54 (0.48-0.59)	0.23 (0.22-0.24)	0.12 (0.11-0.12)	0.07 (0.07-0.07)	0.32 (0.27-0.36)	0.22 (0.22-0.22)	0.58 (0.53-0.64)	0.30 (0.29-0.31)

H. Results

H.1. Changes to Training Performance from Running/Computed Mean and Variance in Batch Normalization Layers

During training, batch normalization layers keep running estimates of its computed mean and variance, which are then used for normalization during evaluation. We find that when using batch normalization, the ELBo and accuracy on the train set can change between `train()` and `eval()` mode (see Fig. H.3). We recommend using the computed mean and variance to evaluate the ELBo on the training set for model selection since this mode is used for normalization during evaluation.
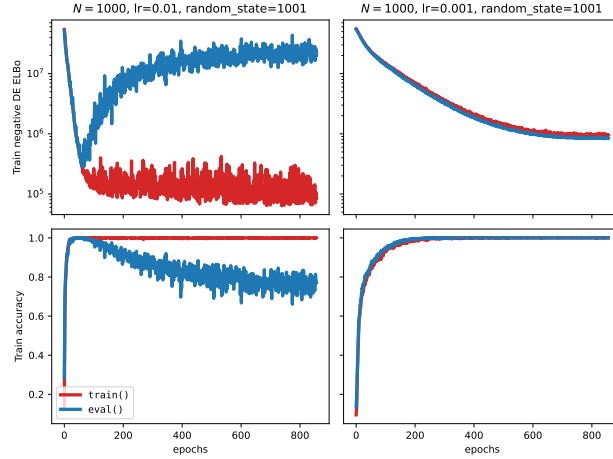

Figure H.3: Negative *data-emphasized ELBo* (DE ELBo) and accuracy on train set for two ResNet-50s trained on CIFAR-10 $N = 1000$. We use *just one sample* every training step to approximate the expected log likelihood. **Takeaway: We find that when using batch normalization, the ELBo and accuracy on the train set can change between** `train()` **and** `eval()` **mode.**