Variational Inference on the Final-Layer Output of Neural Networks

Anonymous authors
Paper under double-blind review

Abstract

Traditional neural networks are simple to train but they typically produce overconfident predictions. In contrast, Bayesian neural networks provide good uncertainty quantification but optimizing them is time consuming due to the large parameter space. This paper proposes to combine the advantages of both approaches by performing Variational Inference in the Final layer Output space (VIFO), because the output space is much smaller than the parameter space. We use neural networks to learn the mean and the variance of the probabilistic output. Using the Bayesian formulation we incorporate collapsed variational inference with VIFO which significantly improves the performance in practice. On the other hand, like standard, non-Beyesian models, VIFO enjoys simple training and one can use Rademacher complexity to provide risk bounds for the model. Experiments show that VIFO provides a good tradeoff in terms of run time and uncertainty quantification, especially for out of distribution data.

1 Introduction

With the development of training and representation methods for deep learning, models using neural networks provide excellent predictions. However, such models fall behind in terms of uncertainty quantification and their predictions are often overconfident (Guo et al., 2017). Bayesian methods provide a methodology for uncertainty quantification by placing a prior over parameters and computing a posterior given observed data, but the computation required for such methods is often infeasible. Variational inference (VI) is one of the most popular approaches for approximating the Bayesian outcome, e.g., (Blundell et al., 2015; Graves, 2011; Wu et al., 2019). By minimizing the KL divergence between the variational distribution and the true posterior and constructing an evidence lower bound (ELBO), one can find the best approximation to the intractable posterior. However, when applied to deep learning, VI requires sampling to compute the ELBO, and it suffers from both high computational cost and large variance in gradient estimation. Wu et al. (2019) have proposed a deterministic variational inference (DVI) approach to alleviate the latter problem. The idea relies on the central limit theorem, which implies that with sufficiently many hidden neurons, the distribution of the output of each layer forms a multivariate Gaussian distribution. Thus we only need to compute the mean and covariance of the output of each layer. However, DVI still suffers from high computational cost and complex optimization.

Inspired by DVI, we observe that the only aspect that affects the prediction is the distribution of the output of the final layer in the neural network. We therefore propose to perform variational inference in the final-layer *output space* (rather than parameter space), where the posterior mean and diagonal variance are learned by a neural network. We call this method VIFO. Like all Bayesian methods, VIFO induces a distribution over its probabilistic predictions and has the advantage of uncertainty quantification in predictions. At the same time, VIFO has a single set of parameters and thus enjoys simple optimization as in non-Bayesian methods.

We can motivate VIFO from several theoretical perspectives. First, we derive improved priors (or regularizers) for VIFO motivated by collapsed variational inference (Tomczak et al., 2021) and empirical bayes (Wu et al., 2019). The new regularizers greatly improve the performance of VIFO. Second, we show that, for the linear case, with expressive priors VIFO can capture the same predictions as standard variational

inference. On the other hand, with practical priors and deep networks VIFO exhibits limited expressiveness. We propose to overcome this limitation by using ensembles that enable fast training and further improve uncertainty quantification. Third, due to its simplicity, one can derive risk bounds for the model through Rademacher complexity. VIFO was motivated as an effective simplification of VI and DVI, and the ensembles of VIFO can be seen as a Bayesian extension of Deep Ensembles (Lakshminarayanan et al., 2017). We discuss the connections to other Bayesian predictors below.

An experimental evaluation compares VIFO with VI and other state of the art approximation methods and to non-Bayesian neural networks (which we refer to as *base models*). The results show that (1) VIFO is much faster than VI and only slightly slower than base models, and (2) ensembles of VIFO achieve better uncertainty quantification on shifted and out-of-distribution data while preserving the quality of in-distribution predictions. Overall, VIFO provides a good tradeoff in terms of run time and uncertainty quantification especially for out-of-distribution data.

2 VIFO

In this section we describe our VIFO method in detail. We start with a description of the base model. Given a neural network parametrized by weights W and input x, the output layer is $z = f_W(x) \in \mathbb{R}^K$. In classification, K is the number of classes. The probability of being class i is defined as

$$p(y = i|z) = \text{softmax}(z)_i = \frac{\exp z_i}{\sum_j \exp z_j}. \tag{1}$$

In regression, $z = (m, l)$ is a 2-dimensional vector and $K = 2$. We apply a function g on l that maps l to a positive real number. The probability of the output y is:

$$p(y|z) = \mathcal{N}(y|m, l) = \frac{1}{\sqrt{2\pi g(l)}} \exp\left(-\frac{(y-m)^2}{2g(l)}\right). \tag{2}$$

As in other models, the same methodology can be used for any type of prediction likelihood $p(y|z)$. This forms the base model. Traditional, non-Bayesian models, minimize $-\log p(y|z)$ or a regularized variant.

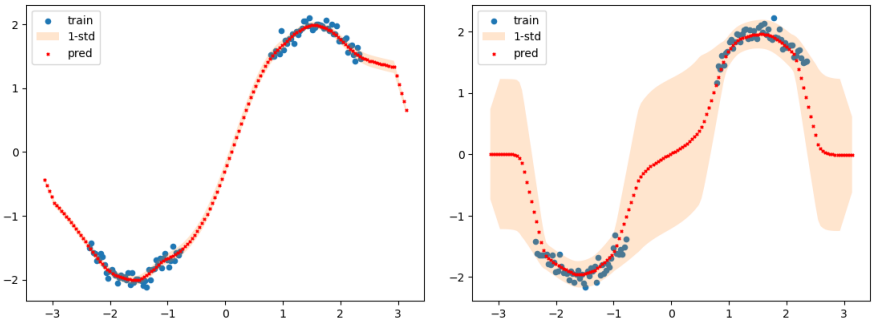
By fixing the weights W, base models map x to z deterministically, while Bayesian methods seek to map x to a distribution over z. Variational inference puts a distribution over W and marginalizes out to get a distribution over z. As shown by Wu et al. (2019), by the central limit theorem, with a sufficiently wide neural network the marginal distribution of z is Gaussian. VIFO pursues this in a direct manner. It has two sets of weights, W_1 and W_2 (with shared components), to model the mean and variance of z. That is, $\mu_q(x) = f_{W_1}(x)$, $\sigma_q(x) = g(f_{W_2}(x))$, where $g : \mathbb{R} \to \mathbb{R}^+$ maps the output to positive real numbers as the variance is positive. Thus, $q(z|x) = \mathcal{N}(z|\mu_q(x), \text{diag}(\sigma_q^2(x)))$, where $\mu_q(x), \sigma_q^2(x)$ are vectors of the corresponding dimension. We will call $q(z|x)$ the *variational output distribution*. Given z, y is generated from the likelihood $p(y|z)$.

Remark 2.1. VIFO in regression is different from the existing models known as the mean-variance estimator (Kabir et al., 2018; Khosravi et al., 2011; Kendall & Gal, 2017). Instead, mean-variance estimators are the base models that VIFO can be applied on. Applying VIFO on these models, we will have *four* outputs, two of which are the means of m and l, and the other two are the variances of m and l. The variances of m and l are from the variational output distribution. Like all Bayesian methods VIFO computes a distribution over distributions which is lacking in non-Bayesian predictions.

The standard Bayesian approach puts a prior on the weights W. Instead, since VIFO models the distribution over z, we put a prior over z. We consider two options, a conditional prior $p(z|x)$ and a simpler prior $p(z)$. Both of these choices yield a valid ELBO using the same steps:

$$\log p(y|x) \geq \mathbb{E}_{q(z|x)}\left[\log \frac{p(y, z|x)}{q(z|x)}\right] = \mathbb{E}_{q(z|x)}[\log p(y|z)] - \text{KL}(q(z|x)||p(z|x)). \tag{3}$$

The approach has some similarity to Dirichlet-based models (Sensoy et al., 2018; Charpentier et al., 2020; Bengs et al., 2022). However, we perform inference on the output layer whereas, as discussed by Bengs et al.



(a) without auxiliary training (b) with auxiliary training

Figure 1: Predictive distribution of VIFO using an MLP. Blue points are training data generated from a sinusoidal function, red points are the predicted mean, shaded area indicates the 1 standard deviation. More details are in Appendix D.1.

(2022), these models implicitly perform variational inference on the prediction. In particular, in that work z is interpreted as a vector in the simplex and $q(z|x)$ and $p(z)$ are Dirichlet distributions, whereas when using VIFO for classification z has a Gaussian distribution and $p(y|z)$ is on the simplex. In other words, we model and regularize different distributions. We discuss related work in more details below.

Eq. (3) is defined for every (x, y). For a dataset $\mathcal{D} = \{(x, y)\}$, we optimize W_1 and W_2 such that

$$\sum_{(x,y)\in\mathcal{D}} \Big\{\mathbb{E}_{q(z|x)}[\log p(y|z)] - \mathrm{KL}(q(z|x)||p(z|x))\Big\}$$

is maximized. We regard the negation of the first term $\mathbb{E}_{q(z|x)}[-\log p(y|z)]$ as the loss term and treat $\mathrm{KL}(q(z|x)\|p(z|x))$ as a regularizer.

2.1 Auxiliary Training

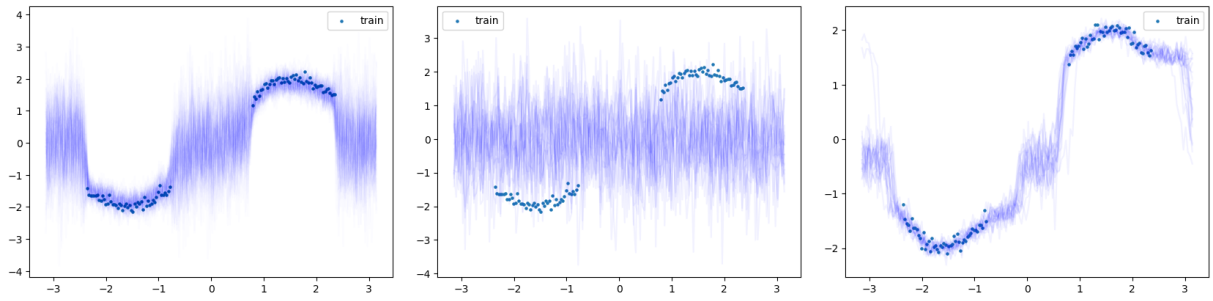
As in prior work (Sun et al., 2019), to improve the uncertainty quantification we introduce auxiliary input x_{aux} and include $\mathrm{KL}(q(z|x_{\text{aux}})\|p(z|x_{\text{aux}}))$ as an additional regularization term. We include corresponding coefficients η and η_{aux} on the regularizers, as is often done in variational approximations (e.g., (Higgins et al., 2017; Jankowiak et al., 2020; Wenzel et al., 2020; Wei et al., 2021; Wei & Khardon, 2022).). Then, viewed as a regularized loss minimization, the optimization objective for VIFO becomes:

$$\min_{W_1,W_2} \sum_{(x,y)\in\mathcal{D}} \Big\{\mathbb{E}_{q(z|x)}[-\log p(y|z)] + \eta\mathrm{KL}(q(z|x)\|p(z|x)) + \eta_{\text{aux}} \sum_{x_{\text{aux}}} \mathrm{KL}(q(z|x_{\text{aux}}), \|p(z|x_{\text{aux}}))\Big\}. \quad (4)$$

Generally the loss term is intractable, so we use Monte Carlo samples to approximate it. In practice, since auxiliary data is not available, we uniformly sample $x^{(i)}_{\text{aux}} \sim \mathrm{Unif}[x^{(i)}_{\min} - \frac{d}{2}, x^{(i)}_{\max} + \frac{d}{2}]$ where $d = x^{(i)}_{\max} - x^{(i)}_{\min}$. Figure 1 shows an example where a MLP is used to learn a complex function over 1 dimensional input space, illustrating that such regularization can improve uncertainty quantification in the area where the data is missing.

2.2 Collapsed VIFO

Bayesian methods are often sensitive to the choice of prior parameters. To overcome this, Wu et al. (2019) used empirical Bayes (EB) to select the value of the prior parameters, and Tomczak et al. (2021) proposed collapsed variational inference, which defined a hierarchical model and performed inference on the prior parameters as well. Empirical Bayes can be regarded as a special case of collapsed variational inference. We show how this idea is applicable in VIFO. In addition to z, we model the prior mean μ_p and variance σ_p^2 as Bayesian parameters. Now the prior becomes $p(z|\mu_p, \sigma_p^2)p(\mu_p, \sigma_p^2)$ and the variational distribution is



(a) vifo-mean (b) vi-naive (c) vi-mean

Figure 2: Induced predictions by learned prior distribution for different methods. Note that VI has a prior over weights and VIFO has a prior over z. For each method we sample values from the prior and calculate predictions y based on the sampled values. We then plot the y values. As we can see, vi-naive induces a uniform prior that does not capture the data distribution, vi-mean has an increased variance in areas where data is missing and vifo-mean does so to a larger extent. Details of the setup for this experiment are given in Appendix D.1.

$q(z|x)q(\mu_p, \sigma_p^2)$. Then the objective becomes:

$$\begin{aligned}\log p(y|x) &\geq \mathbb{E}_{q(z|x)q(\mu_p,\sigma_p^2)}\left[\log \frac{p(y,z,\mu_p,\sigma_p^2|x)}{q(z|x)q(\mu_p,\sigma_p^2)}\right] \\ &= \mathbb{E}_{q(z|x)}[\log p(y|z)] - \mathbb{E}_{q(\mu_p,\sigma_p^2)}[\mathrm{KL}(q(z|x)||p(z|\mu_p,\sigma_p^2))] - \mathrm{KL}(q(\mu_p,\sigma_p^2)||p(\mu_p,\sigma_p^2)). \end{aligned} \tag{5}$$

Similar to equation 4, we treat the first term as a loss and the other two terms as a regularizer along with a coefficient η and aggregate over all data. Since the loss does not contain μ_p and σ_p^2, we can get the optimal $q^*(\mu_p, \sigma_p^2)$ by optimizing the regularizer and the choice of η will not affect $q^*(\mu_p, \sigma_p^2)$. Then we can plug in the value of q^* into Eq equation 5. We next show how to compute $q^*(\mu_p, \sigma_p^2)$ and the final collapsed variational inference objective. The derivations are similar to the ones by Tomczak et al. (2021) but they are applied on z not on W. Recall that K is the dimension of z.

Learn mean, fix variance Let $p(z|\mu_p) = \mathcal{N}(z|\mu_p, \gamma I)$, $p(\mu_p) = \mathcal{N}(\mu_p|0, \alpha I)$. Then $q^*(\mu_p|x)$ is

$$\arg\min_{q(\mu_p)} \mathbb{E}_{q(\mu_p)}[\mathrm{KL}(q(z|x)||p(z|\mu_p))] + \mathrm{KL}(q(\mu_p)||p(\mu_p)),$$

and the optimal $q^*(\mu_p|x)$ can be computed as:

$$\log q^*(\mu_p|x) \propto -\frac{(\mu_q(x)-\mu_p)^\top(\mu_q(x)-\mu_p)}{2\gamma} - \frac{\mu_p^\top \mu_p}{2\alpha},$$

and $q^*(\mu_p|x) = \mathcal{N}(\mu_p|\frac{\alpha}{\alpha+\gamma}\mu_q(x), \frac{\alpha\gamma}{\alpha+\gamma})$. Notice that, unlike the prior, $q^*(\mu_p)$ depends on x. If we put q^* back in the regularizer of equation 5, the regularizer becomes:

$$\frac{1}{2\gamma}\left[1^\top \sigma_q^2(x) + \frac{\gamma}{\gamma+\alpha}\mu_q(x)^\top \mu_q(x)\right] - \frac{1}{2}1^\top \log \sigma_q^2(x) + \frac{K}{2}\log(\gamma+\alpha) - \frac{K}{2}. \tag{6}$$

As in Tomczak et al. (2021), equation 6 puts a factor $\frac{\gamma}{\gamma+\alpha} < 1$ in front of $\mu_q(x)^\top \mu_q(x)$, which weakens the regularization on $\mu_q(x)$. We refer to this method as "vifo-mean".

Figure 2, shows the learned prior for vifo-mean and VI for the same example as in Figure 1. We observe that vifo-mean allows diverse prior distribution and captures the data distribution.

Other Regularizers The same approach can be used for a joint prior $p(z|\mu_p, \sigma_p^2) = \mathcal{N}(z|\mu_p, \sigma_p^2)$, $p(\mu_p) = \mathcal{N}(\mu_p|0, \frac{1}{t}\sigma_p^2)$, $p(\sigma_p^2) = \mathcal{IG}(\sigma_p^2|\alpha, \beta)$, where $\mathcal{IG}$ is inverse Gamma, yielding a method we call "vifo-mv". Similarly, the hierarchical prior in empirical Bayes models the variance but not the mean $p(\sigma_p^2) = \mathcal{IG}(\sigma_p^2|\alpha, \beta)$, $p(z|\sigma_p^2) = \mathcal{N}(z|0, \sigma_p^2)$ and yields "vifo-eb". Derivations are given in Appendix B.

3 Expressiveness of VIFO

VIFO is inspired by DVI and it highly reduces the computational cost. In this section we explore whether VIFO can produce exactly the same predictive distribution as VI. We show that this is the case for linear models but that for deep models VIFO is less powerful. We first introduce the setting of linear models. Let the parameter be θ, then the model is:

$$y|x,\theta \sim p(y|\theta^\top x). \tag{7}$$

For example, $p(y|\theta^\top x) = \mathcal{N}(y|\theta^\top x, \frac{1}{\beta})$ where β is a constant for Bayesian linear regression; and $p(y = 1|\theta^\top x) = \frac{1}{1+\exp(-\theta^\top x)}$ for Bayesian binary classification.

For simplicity, we assume $\theta \in \mathbb{R}^d$, where d is the dimension of x, and then the output dimension $K = 1$. The standard approach specifies the prior of θ to be $p(\theta) = \mathcal{N}(\theta|m_0, S_0)$, and uses $q(\theta) = \mathcal{N}(\theta|m, S)$. Then the ELBO objective, with a dataset $X_N = (x_1, x_2, \ldots, x_N) \in \mathbb{R}^{d\times N}$ and $Y_N = (y_1, y_2, \ldots, y_N) \in \mathbb{R}^N$, is

$$\begin{aligned}
&\sum_{i=1}^N \mathbb{E}_{q(\theta)}[\log p(y_i|\theta^\top x_i)] - \mathrm{KL}(q(\theta)||p(\theta)) \\
=&\sum_{i=1}^N \mathbb{E}_{q(\theta)}[\log p(y_i|\theta^\top x_i)] - \frac{1}{2}[\mathrm{tr}(S_0^{-1}S) - \log|S_0^{-1}S|] - \frac{1}{2}(m-m_0)^\top S_0^{-1}(m-m_0) + \frac{d}{2}.
\end{aligned} \tag{8}$$

As the following theorem shows, if we use a conditional correlated prior and a variational posterior that correlates data points, then in the linear case VIFO can recover the ELBO and VI solution. We defer the proof and discussion of $K > 1$ to Appendix A.2.

Theorem 3.1. *Let $q(z|x) = \mathcal{N}(z|w^\top x, x^\top V x)$ be the variational predictive distribution of VIFO, where w and V are the parameters to be optimized, and let $p(z|X_N) = \mathcal{N}(z|m_0^\top X_N, X_N^\top S_0 X_N)$ and $q(z|X_N) = \mathcal{N}(z|w^\top X_N, X_N^\top V X_N)$ be a correlated and data-specific prior and posterior (which means that for different data x, we have a different prior/posterior over z). Then the VIFO objective is equivalent to the ELBO objective implying identical predictive distributions.*

However, as the next theorem shows, for the non-linear case we cannot produce the variational output distribution $q(z|x)$ as if it is marginalized over the posterior on W.

Theorem 3.2. *Given a neural network f_W parametrized by W and a mean-field Gaussian distribution $q(W)$ over W, there may not exist a set of parameters $\tilde{W}$ such that for all input x we have $\mathbb{E}_{q(W)}[f_W(x)] = f_{\tilde{W}}(x)$.*

The proof is given in Appendix A.2. The significance of these results is twofold. On the one hand, we see from Theorem 3.2 and the conditions of Theorem 3.1 that the representation is more limited, i.e., efficiency comes at some cost. On the other hand, Theorem 3.1 shows the connection of VIFO to VI, which gives a better perspective on the approximation it provides. Moreover, this facilitates the use of existing improvements in VI for VIFO such as collaposed VI applied to VIFO.

In practice, a correlated and data-specific prior $p(z|x)$ is complex, and tuning its hyperparameters would be challenging. Hence, for a practical algorithm we propose to use a simple prior $p(z)$ independent of x. In addition, to reduce computational complexity, we do not learn a full covariance matrix and focus on the diagonal approximation. These aspects limit expressive power but enable fast training of VIFO and hence also ensembles of VIFO.

4 Rademacher Complexity of VIFO

In this section we provide generalization bounds for VIFO through Rademacher Complexity. We need to make the following assumptions.

Assumption 4.1. $\log p(y|z)$ is L_0-Lipschitz in z, i.e., $|\log p(y|z) - \log p(y|z')| \le L_0\|z - z'\|_2$.

Assumption 4.2. The link function g is L_1-Lipschitz.

We show in the Appendix A.1 that these assumptions hold for classification and with a smoothed loss for regression.

Recall that the Rademacher complexity of a set of vectors $A \subseteq \mathbb{R}^N$ is defined as $R(A) = \frac{1}{N} E_{\sigma \sim \{-1,1\}^N}[\sup_{a \in A} \sum_i \sigma_i a_i]$. The Rademacher complexity of the set of loss values induced by functions $f \in \mathcal{F}$ over a dataset S has been used to derive generalization bounds for learning of the class $\mathcal{F}$. We need the following technical lemma, proved in Appendix A.1, that generalizes well known Lipschitz based bounds Shalev-Shwartz & Ben-David (2014) to multi-input functions.

Lemma 4.3. *Consider an L-Lipschitz function $\phi : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, i.e. $\phi(a_1, b_1) - \phi(a_2, b_2) \leq L(|a_1 - a_2| + |b_1 - b_2|)$. For $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^N$, let $\phi(\boldsymbol{a}, \boldsymbol{b})$ denote the vector $(\phi(a_1, b_1), \ldots, \phi(a_N, b_N))$. Let $\phi(A \times B)$ denote $\{\phi(\boldsymbol{a}, \boldsymbol{b}) : \boldsymbol{a} \in A, \boldsymbol{b} \in B\}$, then*

$$R(\phi(A \times B)) \leq L(R(A) + R(B)). \tag{9}$$

Applying the previous lemma sequentially over multiple dimensions we obtain:

Corollary 4.4. *Consider an L-Lipschitz function $\phi : \mathbb{R}^d \to \mathbb{R}$, i.e., for any $x, x' \in \mathbb{R}^d$, $\phi(x) - \phi(x') \leq L\|x - x'\|_1$. Let $\phi(A^d) = \{\phi(a_{1:d,i}) : \boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_d \in A \subset \mathbb{R}^N\}$, then $R(\phi(A^d)) \leq LdR(A)$.*

With the assumptions and technical lemma, we derive the main result:

Theorem 4.5. *Let $\mathcal{H}$ be the set of functions that can be represented with neural networks with parameter space $\mathcal{W}$, $\mathcal{H} = \{f_W(\cdot) | W \in \mathcal{W}\}$. VIFO has two components, so the VIFO hypothesis class is $\mathcal{H} \times \mathcal{H} = \{(f_{W_1}(\cdot), f_{W_2}(\cdot)) \ | W = (W_1, W_2), W_1, W_2 \in \mathcal{W}\}$. Let l be the loss function for VIFO, $l(W, (x, y)) = E_{q_W(z|x)}[-\log p(y|z)]$. Then the Rademacher complexity of VIFO is bounded as $R(l \circ (\mathcal{H} \times \mathcal{H}) \circ S) \leq 2(L_0 \max\{1, L_1\}K) \cdot R(\mathcal{H} \circ S)$, where K is the dimension of z and S is training dataset.*

Proof. We show that the loss is Lipschitz in $f_{W_1}(x)$ and $f_{W_2}(x)$. Fix any x, and W, W'. We denote the mean and standard deviation of $q_W(z|x)$ by μ and s and the same for $q_{W'}(z|x)$. We use $\cdot$ for Hadamard product.

$$\begin{aligned}
&\mathbb{E}_{q_W(z|x)}[-\log p(y|z)] - \mathbb{E}_{q_{W'}(z|x)}[-\log p(y|z)] \\
=&\mathbb{E}_{\epsilon \sim \mathcal{N}(0,I)}[\log p(y|\mu' + \epsilon \cdot s') - \log p(y|\mu + \epsilon \cdot s)] \\
\leq&\mathbb{E}_{\epsilon \sim \mathcal{N}(0,I)}\left[L_0 \|(\mu - \mu') + \epsilon \cdot (s - s')\|_2\right] \qquad \text{(Lipschitz)} \\
\leq&L_0 \|\mu - \mu'\|_2 + L_0 \mathbb{E}_\epsilon \left[\sqrt{\|\epsilon \cdot (s - s')\|_2^2}\right] \\
\leq&L_0 \|\mu - \mu'\|_2 + L_0 \sqrt{\mathbb{E}_\epsilon[\|\epsilon \cdot (s - s')\|_2^2]} \qquad \text{(Jensen's Ineq)} \\
=&L_0(\|\mu - \mu'\|_2 + \|s - s'\|_2) \\
\leq&L_0(\|\mu - \mu'\|_1 + \|s - s'\|_1).
\end{aligned}$$

For the 6th line note that $\mathbb{E}_\epsilon[\|\epsilon \cdot (s - s')\|_2^2] = \mathbb{E}_\epsilon[\sum_i \epsilon_i^2 (s_i - s_i')^2] = \sum_i \mathbb{E}_{\epsilon_i \sim \mathcal{N}(0,1)}[\epsilon_i^2 (s_i - s_i)^2] = \sum_i (s_i - s_i')^2 = \|s - s'\|_2^2$. The loss function is Lipschitz in μ, which is exactly $f_{W_1}(x)$. Further, s is L_1-Lipschitz in the logit $f_{W_2}(x)$, thus, the loss function is $(L_0 \max\{1, L_1\})$-Lipschitz in the concatenation of $f_{W_1}(x)$ and $f_{W_2}(x)$, each of which is of dimension K. The theorem now follows from Corollary 4.4. □

Hence the Rademacher complexity for VIFO is bounded through the Rademacher complexity of deterministic neural networks. This shows one advantage of VIFO which is more amenable to analysis than standard VI due to its simplicity. Risk bounds for VI have been recently developed (e.g., (Germain et al., 2016; Sheth & Khardon, 2017)) but they require different proof techniques. The Rademacher complexity for neural networks is $O\left(\frac{B_W B_x}{\sqrt{N}}\right)$ (Golowich et al., 2018), where B_W bounds the norm of the weights and B_x bounds the input. The Rademacher complexity of VIFO is of the same order.

5 Related Work

VIFO shares some aspects with the model of Kendall & Gal (2017), where both use neural networks to output the mean and covariance of the last layer. However Kendall & Gal (2017) use the cross entropy

loss, $-\log \mathbb{E}_{q(z|x)} p(y|z)$ instead of our loss in Eq. (4), they use dropout for epistemic uncertainty, and their objective has no explicit regularization. Hence unlike VIFO their formulation does not correspond to a standard ELBO. Sharma et al. (2023) model the distribution of the last layer by adding random noise as input and do not give an explicit form of the output distribution. Dirichlet-based methods (Sensoy et al., 2018; Charpentier et al., 2020; Bengs et al., 2022), discussed above, implicitly perform variational inference on the prediction layer, where the network output provides parameters of a Dirichlet distribution. Like VIFO they provide Bayesian predictions in a single pass over the network, but their relation to the standard variational inference in parameter space is non obvious. On the other hand, VIFO is a single pass method clearly related to VI in parameter space which enables the benefits of collapsed variational inference. Thus VIFO can be seen to bridge between Dirichlet methods and VI. Another related line of work (Sun et al., 2019; Tran et al., 2022) performs variational inference in function space. However, they focus on choosing a better prior in weight space which is induced from Gaussian Process priors on function space, whereas VIFO directly induces a simple prior on function space.

VIFO differs from other existing variational inference methods as well. Last-layer variational inference (Brosse et al., 2020) performs variational inference on the *parameters* of the last layer, while we perform variational inference on the *output* of the last layer. The last layer usually contains more parameters than the output (which has constant size). Thus, Last-layer VI is much closer to VI and VIFO regularizes and optimizes in a different space. The local reparametrization trick (Tomczak et al., 2020; Oleksiienko et al., 2022) performs two forward passes with the mean and variance to sample the output for each layer, while we only require one pass and sample the output of the last layer for prediction.

Various alternative Bayesian techniques have been proposed. One direction is to get samples from the true posterior, as in Markov chain Monte Carlo methods Wenzel et al. (2020); Izmailov et al. (2021). Expectation propagation aims to minimize the reverse KL divergence to the true posterior Teh et al. (2015); Li et al. (2015). These Bayesian methods, including variational inference, often suffer from high computational cost and therefore hybrid methods were proposed. Stochastic weight averaging Gaussian Maddox et al. (2019) forms a Gaussian distribution over parameters from the stochastic gradient descent trajectory in the base model. Dropout Gal & Ghahramani (2016) randomly sets weights 0 to capture uncertainty in the model. Deep ensembles Lakshminarayanan et al. (2017) use ensembles of base models learned with random initialization and shuffling of data points and then average the predictions. These methods implicitly perform approximate inference. In addition to these methods, there are also non-Bayesian methods to calibrate overconfident predictions, for example, temperature scaling (Guo et al., 2017) introduces a temperature parameter to anneal the predictive distribution to avoid high confidence. VIFO strikes a balance between simplicity and modelling power to enable simple training and Bayesian uncertainty quantification. On the one hand, VIFO can be seen as a simplification of VI. On the other hand, it can be seen as an extension of the base model. From this perspective, the use of ensembles of VIFO, which extend the ensembles of Lakshminarayanan et al. (2017), are highly motivated as a practical algorithm. As shown below, ensembles of VIFO are indeed very effective in practice.

6 Experiments

In this section, we compare the empirical performance of VIFO with VI and hybrid methods that use the base model, as well as repulsive ensembles ("repulsive", (D'Angelo & Fortuin, 2021)), the Dirichlet-based model ("dir", (Sensoy et al., 2018)) and dropout (Gal & Ghahramani, 2016). VI candidates include the VI algorithm ("vi-naive"(Blundell et al., 2015)) with fixed prior parameters, and other variations from collapsed variational inference (Tomczak et al., 2021) and empirical Bayes (Wu et al., 2019). Non-Bayesian and hybrid methods include the base model ("sgd", because it uses stochastic gradient descent as optimizer), stochastic weight averaging ("swa", which uses the average of the sgd trajectory on the base model as the final weights) from Izmailov et al. (2018) and SWA-Gaussian ("swag", which uses the sgd trajectory to form a Gaussian distribution over the neural network weight space) from Maddox et al. (2019). We use ensembles of the base models which are known as deep ensembles (Lakshminarayanan et al., 2017), and the ensembles of SWAG models, which are the multiSWAG model of Wilson & Izmailov (2020), both of which are considered strong baselines for uncertainty quantification (Ovadia et al., 2019). Our main goal is to show:

Table 1: Running time for training 1 epoch with batch size 512, AlexNet

dataset	CIFAR10	CIFAR100	SVHN	STL10
size	50000	50000	73257	500
VI	8.51 ± 0.41	8.27 ± 0.40	11.56 ± 0.39	1.75 ± 0.41
VIFO	2.18 ± 0.39	2.17 ± 0.43	2.72 ± 0.38	1.16 ± 0.40
base	1.97 ± 0.41	1.99 ± 0.43	2.46 ± 0.40	1.12 ± 0.38

- VIFO is much faster than VI and only slightly slower than base models;
- Ensembles of VIFO preserve the quality of in-distribution predictions;
- Ensembles of VIFO achieve better uncertainty quantification on shifted and out-of-distribution (OOD) data than all baselines.

For our main experiments, we pick four large datasets, CIFAR10, CIFAR100, SVHN, STL10, together with two types of neural networks, AlexNet (Krizhevsky et al., 2012) and PreResNet20 (He et al., 2016). The regularization parameter η is fixed to 0.1 for both VIFO and VI, as this choice yields better performance compared with the standard choice $\eta = 1$. Empirically we observe that using collapsed variational inference in VI does not improve the performance. This is because Tomczak et al. (2021) used $\eta = 1$ to obtain their results whereas we use $\eta = 0.1$ that yields better performance and is a much stronger baseline. In addition, vifo-mean and vifo-mv perform better than other variants of VIFO. Thus, we only list these variants in our main paper and provide full results for other variants for VIFO and VI in the appendix. For each method we run 5 independent runs and report means and standard deviations in results. Complete details for the setup and hyperparameters are given in Appendix D.2.

6.1 Run Time

Ignoring the data preprocessing time, we compare the run time of training 1 epoch of VI, VIFO and the base model. In Table 1 we show the mean and standard deviation of 10 runs of these methods. Different regularizers do not affect run time, so we only show that of vi-naive for VI and vifo-mean for VIFO. As shown in Table 1, VIFO is much faster than VI and is slightly slower than the base model.

These differences are dominated by sampling and forward passes in the network. The base model only needs 1 forward pass without sampling per batch. VIFO needs 1 forward pass and M samples of size K per batch. VI needs M samples of the parameter space and M forward passes per batch. The same facts apply for predictions on test data, where the advantage can be important for real time applications.

6.2 Ensembles of VIFO

Theorem 3.2 points out the limit of expressiveness of VIFO. To overcome this, we use ensembles of VIFO, which independently train multiple VIFO models and average their predictions. Section 6.1 establishes fast training of VIFO, allowing us to train each VIFO model simultaneously while still maintaining the running time advantage of VIFO. Table 2 shows that with ensembles, VIFO achieves much better log loss than when using a single model. Notice that Table 2 lists the results for VIFO with auxiliary training and the same phenomenon occurs for VIFO without auxiliary training as well. This indicates that ensembles of VIFO are much more expressive than a single VIFO. In the following experiments, we use ensembles of VIFO. For a fair comparison, we use ensembles for all other methods except for VI (which is very time-consuming) and repulsive ensembles (which are themselves ensembles).

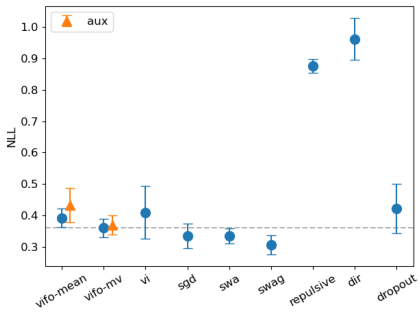
6.3 In-distribution Performance

In this section we use log loss and accuracy to measure the performance for in-distribution data.

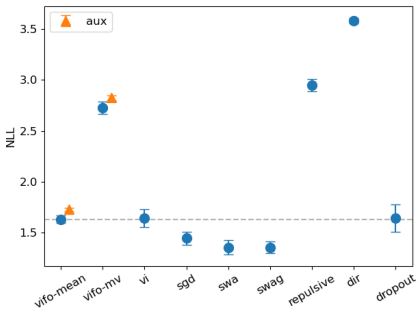
Fig. 3 and Fig. E.1 in Appendix compare main methods in terms of log loss. First, we observe that repulsive ensembles and the Dirichlet method have much worse log loss than all other methods and they tend to give

Table 2: Test log loss of single VIFO and ensembles of VIFO.

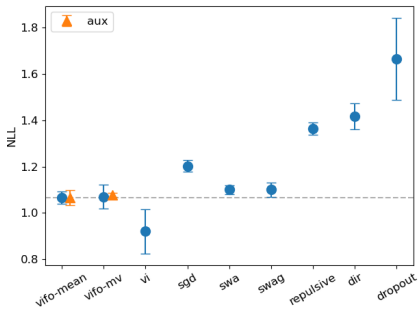
	vifo-mean		vifo-mv	
	single	ensemble	single	ensemble
CIFAR10	0.527 ± 0.015	0.345 ± 0.003	0.626 ± 0.010	0.324 ± 0.001
CIFAR100	2.253 ± 0.032	1.688 ± 0.006	2.688 ± 0.029	1.725 ± 0.003
STL10	1.333 ± 0.065	1.055 ± 0.008	1.531 ± 0.019	1.123 ± 0.008
SVHN	0.509 ± 0.029	0.351 ± 0.005	0.520 ± 0.027	0.298 ± 0.009



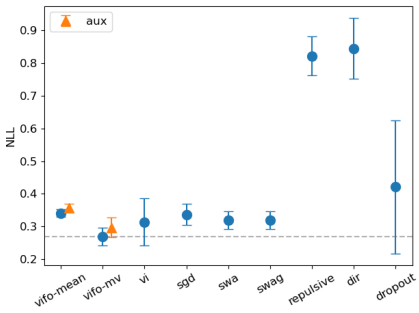
(a) CIFAR10



(b) CIFAR100



(c) STL10



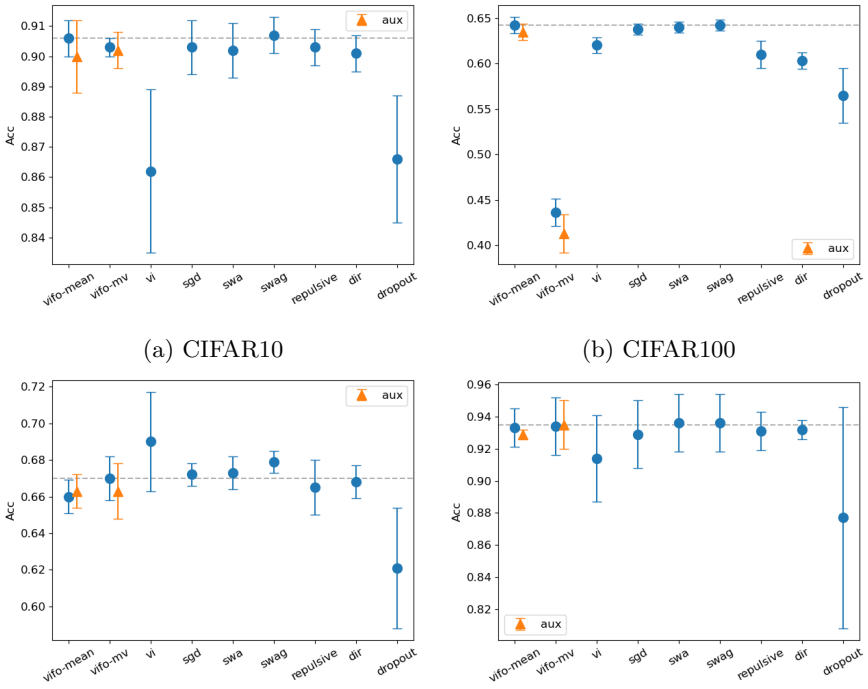
(d) SVHN

Figure 3: Test log loss of image datasets on PreResNet20. Dashed lines indicate the best version of VIFO. The error bar is three times of the standard deviation for better visualization and same for other figures.

underconfident predictions. Second, we observe that using auxiliary training slightly increases the log loss but the increase is negligible. Later we can see that auxiliary training improves the uncertainty quantification for out-of-distribution data. We observe that VIFO is competitive with all methods in terms of log loss, with relatively small differences between the top group of methods in each case. Fig. 4 and Fig. E.2 show accuracy on test data in the same experiments, revealing that in many cases VIFO outperforms VI and it is competitive with all methods. Finally, there is no clear winner between vifo-mean and vifo-mv; vifo-mv provides a small advantage overall but might be more sensitive as illustrated by the performance on CIFAR100 with PreResNet20.

6.4 Uncertainty Quantification

In this section we examine whether VIFO can capture the uncertainty in predictions for shifted and OOD data. We measure performance using ECE, Entropy and AUC for detecting OOD data. These represent a comprehensive set of measures from the literature. For datasets, for uncertainty under data shift, STL10 and CIFAR10 can be treated as a shifted dataset for each other, as the figure size of STL10 is different from CIFAR10, and STL10 shares some classes with CIFAR10 so the labels are meaningful. For uncertainty under



(a) CIFAR10 (b) CIFAR100

(c) STL10 (d) SVHN

Figure 4: Test accuracy of image datasets on PreResNet20. Dashed lines indicate the best version of VIFO.

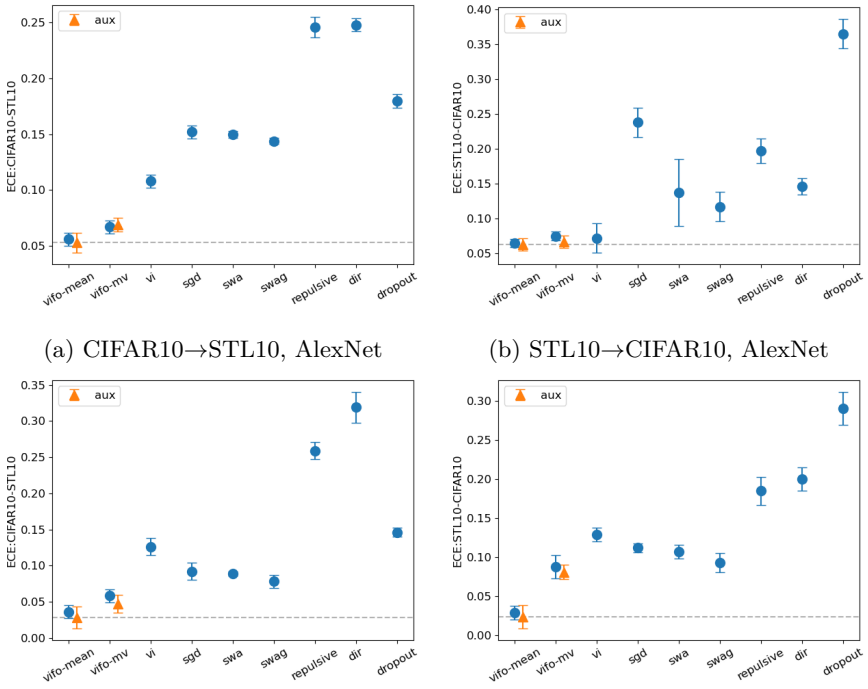
OOD data, we choose the SVHN dataset as an OOD dataset for CIFAR10 and STL10, as SVHN contains images of digits and the labels of SVHN are not meaningful in the context of CIFAR10.

Expected Calibration Error (ECE) ECE (Naeini et al., 2015; Ovadia et al., 2019) is often used to measure the uncertainty quantification under data shift. We separate data into bins of the same size according to the confidence level, calculate the difference between the accuracy and the averaged confidence in each bin and then average the absolute differences among all bins. Better calibrated models have lower ECE. ECE has its faults (for example the trivial classifier has zero ECE) but it is nonetheless informative. We selected the number of bins to be 20.

Fig. 5 shows the ECE of each method under data shift. As we can see, both vifo-mean and vifo-mv achieve the best performance compared to all other methods.

Entropy Entropy (Ovadia et al., 2019) of the categorical predictive distribution is used to measure the uncertainty quantification for out-of-distribution (OOD) data as the labels for OOD data are meaningless. We want our model to be as uncertain as possible and this implies high entropy and low confidence (the maximum probability assigned to any class) in the predictive distribution. We summarize the averaged entropy for the entire dataset in Fig. 6 and Fig. E.3. We can see that both vifo-mean and vifo-mv are better than all other methods except repulsive ensembles and the Dirichlet method. However, as observed in Fig. 5 and Fig. 3, repulsive ensembles and the Dirichlet method have poor performance in terms of log loss due to underconfident predictions. Hence they achieve high entropy by sacrificing in distribution performance whereas VIFO performs well. Further, we observe from Fig. 6 that auxiliary training greatly improve the performance of VIFO on PreResNet20. Auxiliary training only has a small impact on VIFO with AlexNet (see Fig. E.3) but VIFO already performs well without auxiliary training in this case.

AUROC We use maximum probability of the categorical predictive distribution as the criterion to separate in-distribution and OOD data and compute the area under the ROC curve (Malinin & Gales, 2018). AUROC overcomes the drawbacks of ECE and entropy because a trivial model cannot yield the best performance.



(a) CIFAR10→STL10, AlexNet (b) STL10→CIFAR10, AlexNet

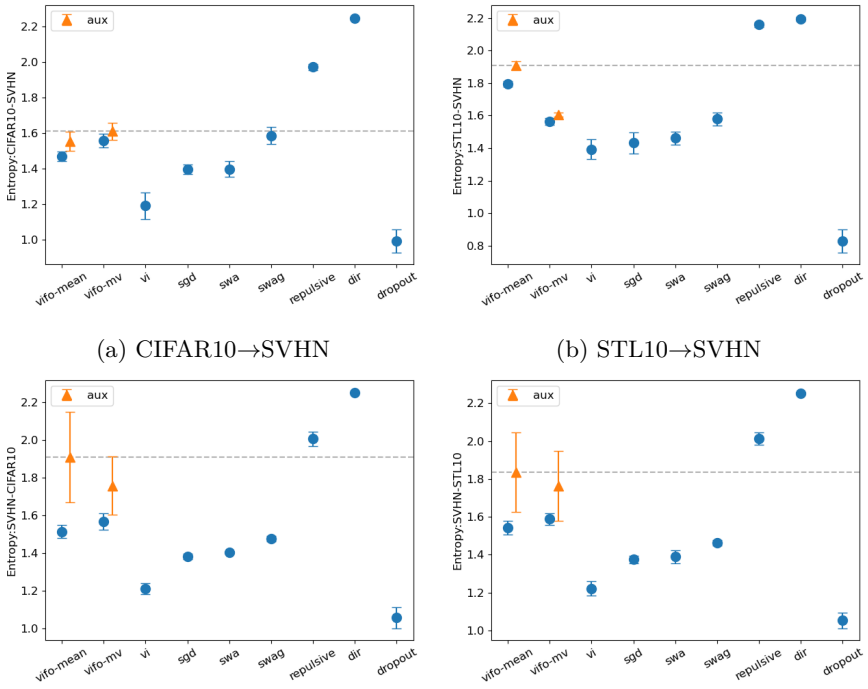
(c) CIFAR10→STL10, PreResNet20 (d) STL10→CIFAR10, PreResNet20

Figure 5: ECE (↓) on AlexNet and PreResNet20 under data shift. Dashed line indicates the best performance of VIFO. Numerical results are listed in the Appendix.

Detailed comparison plots are in given in Fig. E.4 and Fig. E.5 in the Appendix. We first note that, as above, auxiliary training improves the performance on PreResNet20 but not significantly on AlexNet. We found that there is no single method that consistently outperforms all other methods. Instead, for better visualization, we show the comparison of vifo-mean and vifo-mv with other methods in Fig. 7. We count the number of experiments that VIFO is better than one other method and get the proportion that VIFO is better. We observe that overall, vifo-mv is better than all other methods except the Dirichlet method and that it ranks better than vifo-mean. As discussed above, the success of the Dirichlet method on OOD data is achieved by sacrificing calibration and in distribution performance. On the other hand, VIFO outperforms all other baselines for OOD and has strong in distribution performance and hence give better overall predictions.

7 Conclusion

In Bayesian neural networks, the distribution of the last layer directly affects the predictive distribution. Motivated by this fact, we proposed variational inference on the final-layer output, VIFO, that uses a neural network to directly learn the mean and variance of the last layer. We showed that VIFO can match the expressive power of VI in linear cases with a strong prior but that in general it provides a less expressive model. On the other hand the simplicity of the model enables fast training of ensembles of VIFO and facilitates convergence analysis through Rademacher bounds. In addition, VIFO can be derived as a non-standard variational lower bound, which provides an approximation for the last layer. This connection allowed us to derive better regularizations for VIFO by using collapsed variational inference over a hierarchical prior. Empirical evaluation highlighted that ensembles of VIFO are competitive with or outperform other methods in terms of in-distribution loss and out-of-distribution data detection. Hence VIFO gives a new attractive approach for approximate inference in Bayesian models. The efficiency of VIFO also means faster test time predictions which can be important when deploying Bayesian models for real-time applications. In future work it would be interesting to explore the complexity performance tradeoff provided by VIFO, and the connections to variational inference in functional space that induces more complex priors.



(a) CIFAR10→SVHN (b) STL10→SVHN

(c) SVHN→CIFAR10 (d) SVHN→STL10

Figure 6: Entropy (↑) on PreResNet20.

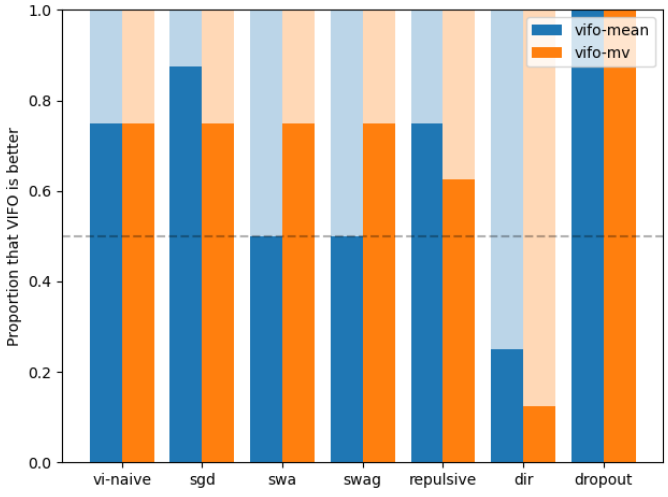


Figure 7: Comparison of VIFO with all other methods in terms of AUROC on OOD data. Y-axis is the proportion of experiments that VIFO is better than other methods. Exact AUROC values are provided in the appendix.

References

Viktor Bengs, Eyke Hüllermeier, and Willem Waegeman. Pitfalls of epistemic uncertainty quantification through loss minimisation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=epjxT_ARZW5`.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural network. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1613–1622, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/blundell15.html.

Nicolas Brosse, Carlos Riquelme, Alice Martin, Sylvain Gelly, and Éric Moulines. On last-layer algorithms for classification: Decoupling representation from uncertainty estimation, 2020. URL https://arxiv.org/abs/2001.08049.

Bertrand Charpentier, Daniel Zügner, and Stephan Günnemann. Posterior network: Uncertainty estimation without ood samples via density-based pseudo-counts. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1356–1367. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/0eac690d7059a8de4b48e90f14510391-Paper.pdf.

Francesco D'Angelo and Vincent Fortuin. Repulsive deep ensembles are bayesian. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=LAKplpLMbP8.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, pp. 1050–1059. JMLR.org, 2016.

Pascal Germain, Francis Bach, Alexandre Lacoste, and Simon Lacoste-Julien. Pac-bayesian theory meets bayesian inference. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/84d2004bf28a2095230e8e14993d398d-Paper.pdf.

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. In Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet (eds.), *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, pp. 297–299. PMLR, 06–09 Jul 2018. URL https://proceedings.mlr.press/v75/golowich18a.html.

Alex Graves. Practical variational inference for neural networks. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011. URL https://proceedings.neurips.cc/paper/2011/file/7eb3c8be3d411e8ebfab08eba5f49632-Paper.pdf.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 1321–1330. JMLR.org, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (eds.), *Computer Vision – ECCV 2016*, pp. 630–645, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46493-0.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-VAE: Learning basic visual concepts with a constrained variational framework. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=Sy2fzU9gl.

Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry P. Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. In Amir Globerson and Ricardo Silva (eds.), *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*, pp. 876–885. AUAI Press, 2018. URL http://auai.org/uai2018/proceedings/papers/313.pdf.

Pavel Izmailov, Sharad Vikram, Matthew D Hoffman, and Andrew Gordon Gordon Wilson. What are bayesian neural network posteriors really like? In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 4629–4640. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/izmailov21a.html.

Martin Jankowiak, Geoff Pleiss, and Jacob R Gardner. Parametric gaussian process regressors. In *ICML*, 2020.

H. M. Dipu Kabir, Abbas Khosravi, Mohammad Anwar Hosen, and Saeid Nahavandi. Neural network-based uncertainty quantification: A survey of methodologies and applications. *IEEE Access*, 6:36218–36234, 2018. doi: 10.1109/ACCESS.2018.2836917.

Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/2650d6089a6d640c5e85b2b88265dc2b-Paper.pdf.

Abbas Khosravi, Saeid Nahavandi, Doug Creighton, and Amir F. Atiya. Comprehensive review of neural network-based prediction intervals and new advances. *IEEE Transactions on Neural Networks*, 22(9): 1341–1356, 2011. doi: 10.1109/TNN.2011.2162110.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper/2017/file/9ef2ed4b7fd2c810847ffa5fa85bce38-Paper.pdf.

Yingzhen Li, Jose Miguel Hernández-Lobato, and Richard E. Turner. Stochastic expectation propagation. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, pp. 2323–2331, Cambridge, MA, USA, 2015. MIT Press.

Wesley J. Maddox, Timur Garipov, Pavel Izmailov, Dmitry Vetrov, and Andrew Gordon Wilson. *A Simple Baseline for Bayesian Uncertainty in Deep Learning*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/3ea2db50e62ceefceaf70a9d9a56a6f4-Paper.pdf.

Tom Minka. Inferring a Gaussian distribution, 2001. http://www.stat.cmu.edu/∼minka/papers/gaussian.html.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pp. 2901–2907. AAAI Press, 2015. ISBN 0262511290.

Illia Oleksiienko, Dat Thanh Tran, and Alexandros Iosifidis. Variational neural networks, 2022. URL https://arxiv.org/abs/2207.01524.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua V. Dillon, Balaji Lakshminarayanan, and Jasper Snoek. *Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty under Dataset Shift*. Curran Associates Inc., Red Hook, NY, USA, 2019.

Kaare Brandt Petersen and Michael Syskind Pedersen. The matrix cookbook. 2012. URL http://www.math.uwaterloo.ca/~hwolkowi//matrixcookbook.pdf.

Murat Sensoy, Lance Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/a981f2b708044d6fb4a71a1463242520-Paper.pdf.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning - From Theory to Algorithms*. Cambridge University Press, 2014.

Mrinank Sharma, Sebastian Farquhar, Eric Nalisnick, and Tom Rainforth. Do bayesian neural networks need to be fully stochastic? In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent (eds.), *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 7694–7722. PMLR, 25–27 Apr 2023. URL https://proceedings.mlr.press/v206/sharma23a.html.

Rishit Sheth and Roni Khardon. Excess risk bounds for the bayes risk using variational inference in latent gaussian models. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/7edccc661418aeb5761dbcdc06ad490c-Paper.pdf.

Shengyang Sun, Guodong Zhang, Jiaxin Shi, and Roger Grosse. FUNCTIONAL VARIATIONAL BAYESIAN NEURAL NETWORKS. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rkxacs0qY7.

Yee Teh, Leonard Hasenclever, Thibaut Lienart, Sebastian Vollmer, Stefan Webb, Balaji Lakshminarayanan, and Charles Blundell. Distributed bayesian learning with stochastic natural gradient expectation propagation and the posterior server. *Journal of Machine Learning Research*, 18, 12 2015.

Marcin Tomczak, Siddharth Swaroop, and Richard Turner. Efficient low rank gaussian variational inference for neural networks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 4610–4622. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/310cc7ca5a76a446f85c1a0d641ba96d-Paper.pdf.

Marcin B. Tomczak, Siddharth Swaroop, Andrew Y. K. Foong, and Richard E Turner. Collapsed variational bounds for bayesian neural networks. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=ykN3tbJ0qmX.

Ba-Hien Tran, Simone Rossi, Dimitrios Milios, and Maurizio Filippone. All You Need is a Good Functional Prior for Bayesian Deep Learning. *Journal of Machine Learning Research*, 23:1–56, 2022.

Yadi Wei and Roni Khardon. On the performance of direct loss minimization for bayesian neural networks, 2022.

Yadi Wei, Rishit Sheth, and Roni Khardon. Direct loss minimization for sparse gaussian processes. In Arindam Banerjee and Kenji Fukumizu (eds.), *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 2566–2574. PMLR, 13–15 Apr 2021. URL https://proceedings.mlr.press/v130/wei21b.html.

Florian Wenzel, Kevin Roth, Bastiaan S. Veeling, Jakub Swiątkowski, 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 *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org, 2020.

Andrew G Wilson and Pavel Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 4697–4708. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/322f62469c5e3c7dc3e58f5a4d1ea399-Paper.pdf.

Andrew Gordon Wilson, Pavel Izmailov, Matthew D Hoffman, Yarin Gal, Yingzhen Li, Melanie F Pradier, Sharad Vikram, Andrew Foong, Sanae Lotfi, and Sebastian Farquhar. Evaluating approximate inference in bayesian deep learning. In Douwe Kiela, Marco Ciccone, and Barbara Caputo (eds.), *Proceedings of the NeurIPS 2021 Competitions and Demonstrations Track*, volume 176 of *Proceedings of Machine Learning Research*, pp. 113–124. PMLR, 06–14 Dec 2022. URL https://proceedings.mlr.press/v176/wilson22a.html.

Anqi Wu, Sebastian Nowozin, Edward Meeds, Richard E. Turner, Jose Miguel Hernandez-Lobato, and Alexander L. Gaunt. Deterministic variational inference for robust bayesian neural networks. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=B1l08oAct7.

A Proofs

A.1 Proofs in Section 4

Proof of Lemma 4.3. We prove the lemma for $L = 1$. If this is not the case, we can define $\phi' = \frac{1}{L}\phi$, and use the fact that $R(\phi(A\times B)) \leq LR(\phi'(A\times B))$. Let $C_i = \{(a_1+b_1, \ldots, a_{i-1}+b_{i-1}, \phi'(a_i, b_i), a_{i+1}+b_{i+1}, \ldots, a_N+b_N) : a \in A, b \in B\}$. It suffices to prove that for any set A, B and all i we have $R(C_i) \leq R(A) + R(B)$. Without loss of generality we prove the case for $i = 1$.

$$
\begin{aligned}
NR(C_1) &= \mathbb{E}_\sigma \left[\sup_{c\in C_1} \sigma_1\phi(a_1, b_1) + \sum_{i=2}^{N} \sigma_i(a_i + b_i) \right] \\
&= \frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_N} \left[\sup_{a\in A, b\in B} \left(\phi(a_1, b_1) + \sum_{i=2}^{N} \sigma_i(a_i + b_i) \right) \right. \\
&\qquad \left. + \sup_{a'\in A, b'\in B} \left(-\phi(a'_1, b'_1) + \sum_{i=2}^{N} \sigma_i(a'_i + b'_i) \right) \right] \\
&= \frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_N} \left[\sup_{a,a'\in A, b,b'\in B} \left(\phi(a_1, b_1) - \phi(a'_1, b'_1) + \sum_{i=2}^{N} \sigma_i(a_i + b_i) + \sum_{i=2}^{N} \sigma_i(a'_i + b'_i) \right) \right] \\
&\leq \frac{1}{2}\mathbb{E}_{\sigma_2\ldots\sigma_N} \left[\sup_{a,a'\in A, b,b'\in B} \left(|a_1 - a'_1| + |b_1 - b'_1| + \sum_{i=2}^{N} \sigma_i(a_i + b_i) + \sum_{i=2}^{N} \sigma_i(a'_i + b'_i) \right) \right] \\
&= \frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_N} \left[\sup_{a,a'\in A} \left(a_1 - a'_1 + \sum_{i=2}^{N} \sigma_i a_i + \sum_{i=2}^{N} \sigma_i a'_i \right) \right] \\
&\qquad + \frac{1}{2}\mathbb{E}_{\sigma_2,\ldots,\sigma_N} \left[\sup_{b,b'\in B} \left(b_1 - b'_1 + \sum_{i=2}^{N} \sigma_i b_i + \sum_{i=2}^{N} \sigma_i b'_i \right) \right] \\
&= NR(A) + NR(B).
\end{aligned}
$$

□

Verifying Assumption 4.1: We next verify that Assumption 4.1 holds for classification and (with a modified loss) for regression.

For K-classification, z is K-dimensional and the negative log-likelihood is

$$-\log p(y=k|z) = -\log \frac{\exp(z_k)}{\sum_{i=1}^K \exp(z_i)} = -z_k + \log \sum_{i=1}^K \exp(z_i)$$

which is 1-Lipschitz in z.

For regression, $z=(m,l)$ is 2-dimensional, and the negative log-likelihood is:

$$-\log p(y|z) = \frac{1}{2}(y-m)^2 \exp(-l) + \frac{1}{2}l.$$

Neither the quadratic function nor exponential function is Lipschitz. But we can replace the unbounded quadratic function $(y-m)^2$ with a bounded version $\min\{(y-m)^2, B_m^2\}$, and replace the exponential function $\exp(-l)$ with $\min\{\exp(-l), B_l\}$, where $B>0$, to guarantee the Lipschitzness. Now the negative log-likelihood is:

$$-\log p(y|z) = \frac{1}{2}\min\{(y-m)^2, B_m^2\}\min\{\exp(-l), B_l\} + \frac{1}{2}l,$$

is $(B_m B_l)$-Lipschitz in m, $\left(\frac{1}{2} + \frac{1}{2}B_m^2 B_l\right)$-Lipschitz in l.

Verifying Assumption 4.2: For Assumption 4.2, we can use $g(l) = \log(1+\exp(l))$ which is 1-Lipschitz. If $g(l) = \exp(l)$ is the exponential function, we can use a bounded variant that satisfies the requirement $g(l) = \max\{\exp(x), B_g\}$.

A.2 Proofs in Section 3

Proof of Theorem 3.1. Assume $N > d$. Note that with the the correlated prior and posterior the covariance function is rank deficient so we have to interpret inverses and determinants appropriately. Here we use pseudo inverse and pseudo determinant. The VIFO objective is:

$$\sum_{i=1}^N \left\{\mathbb{E}_{q(z|x_i)}[\log p(y_i|z)]\right\} - \mathrm{KL}(q(z|X_N)||p(z|X_N)) \tag{10}$$

$$= \sum_{i=1}^N \left\{\mathbb{E}_{q(z|x_i)}[\log p(y_i|z)]\right\} - \frac{1}{2}\mathrm{tr}((X_N^\top S_0 X_N)^{-1}(X_N^\top V X_N)) + \frac{N}{2}$$
$$+\frac{1}{2}\log|(X_N^\top S_0 X_N)^{-1}(X_N^\top V X_N)| - \frac{1}{2}(w^\top X_N - m_0^\top X_N)(X_N^\top S_0 X_N)^{-1}(w^\top X_N - m_0^\top X_N)^\top. \tag{11}$$

First consider the loss term. Let L be the Cholesky decomposition of V, i.e. $V = LL^\top$. By reparametrization, for $\epsilon \sim \mathcal{N}(0, I_d)$, $w^\top x_i + x_i^\top L\epsilon \sim \mathcal{N}(w^\top x_i, x_i^\top LL^\top x_i)$ and thus

$$\begin{aligned}\mathbb{E}_{q(z|x_i)}[\log p(y_i|z)] &= \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I_d)}[\log p(y_i|w^\top x_i + x_i^\top L\epsilon)] \\ &= \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I_d)}[\log p(y_i|(w+L\epsilon)^\top x_i)] \\ &= \mathbb{E}_{\theta\sim\mathcal{N}(w,LL^\top)}[\log p(y_i|\theta^\top x_i)],\end{aligned} \tag{12}$$

where the last equality uses reparametrization in a reverse order. By aligning $w=m$ and $V = LL^\top = S$, we recognize that Eq equation 12 is exactly the loss term in Eq equation 8. Thus the low-dimensional posterior on z yields the same loss term as the high-dimensional posterior over W.

For the regularization, we use the pseudo inverse derivation from Eq (224) of Petersen & Pedersen (2012), where for $A=CD$ we have $A^+ = D^\top(DD^\top)^{-1}(C^\top C)^{-1}C^\top$ to get

$$(X_N^\top S_0 X_N)^{-1} = X_N^\top (X_N X_N^\top)^{-1} S_0^{-1} (X_N X_N^\top)^{-1} X_N$$

and the same for V. Thus,

$$\begin{aligned}(X_N^\top S_0 X_N)^{-1}(X_N^\top V X_N) &= X_N^\top (X_N X_N^\top)^{-1} S_0^{-1} (X_N X_N^\top)^{-1} X_N X_N^\top V X_N \\ &= X_N^\top (X_N X_N^\top)^{-1} S_0^{-1} V X_N, \\ \mathrm{tr}|X_N^\top (X_N X_N^\top)^{-1} S_0^{-1} V X_N| &= \mathrm{tr}|X_N X_N^\top (X_N X_N^\top)^{-1} S_0^{-1} V| \\ &= \mathrm{tr}(S_0^{-1} V),\end{aligned}$$

and

$$\begin{aligned}&(w^\top X_N - m_0^\top X_N)(X_N^\top S_0 X_N)^{-1}(w^\top X_N - m_0^\top X_N)^\top \\ =&(w-m_0)^\top X_N (X_N^\top (X_N X_N^\top)^{-1} S_0^{-1} (X_N X_N^\top)^{-1} X_N) X_N^\top (w - m_0) \\ =&(w-m_0)^\top (X_N X_N^\top)(X_N X_N^\top)^{-1} S_0^{-1} (X_N X_N^\top)^{-1} (X_N X_N^\top)(w-m_0) \\ =&(w-m_0)^\top S_0^{-1}(w-m_0).\end{aligned}$$

For the The log-determinant term we use the pseudo-determinant (Minka, 2001), which is the product of non-zero eigenvalues. Let $(\lambda_i, u_i)_{i=1}^d$ be the set of eigenvalues and eigenvectors of $S_0^{-1}V$, i.e., $S_0^{-1}Vu_i = \lambda u_i$, and let $X_N^\ddagger = X_N^\top (X_N X_N^\top)^{-1}$ denote the pseudo inverse of X_N, then

$$(X_N^\ddagger S_0^{-1} V X_N) X_N^\ddagger u_i = X_N^\ddagger S_0^{-1} V u_i = \lambda X_N^\ddagger u_i, \tag{13}$$

thus $(\lambda_i, X_N^\ddagger u_i)_{i=1}^d$ is the eigenvalues and eigenvectors of $X_N^\top (X_N X_N^\top)^{-1} S_0^{-1} V X_N$. Since the rank of this matrix is at most d, other eigenvalues are 0 and the pseudo determinant is $\prod_{i=1}^d \lambda_i$, which is exactly the determinant of $S_0^{-1}V$. Then the regularization term in Eq equation 8 can be simplified to:

$$-\mathrm{KL}(q(z|X_N)||p(z|X_N)) = -\frac{1}{2}\mathrm{tr}(S_0^{-1}V) + \frac{1}{2}\log|S_0^{-1}V| - \frac{1}{2}(w-m_0)^\top S_0^{-1}(w-m_0) + \frac{N}{2}. \tag{14}$$

By aligning $w = m, V = S$, we can seet that equation 14 is exactly the regularizer in equation 8 ignoring the constant. □

Note for the case $K > 1$: Let $\theta \in \mathbb{R}^{d\times K}$. For VI, we make a mean field assumption with $q(\theta_k) = \mathcal{N}(\theta_k|m_k, S_k)$ and $q(\theta) = \prod_{k=1}^K q(\theta_k)$, where θ_k is the k-th column of θ. For VIFO, using mean field let $q(z_k|x) = \mathcal{N}(z_k|w_k^\top x, x^\top V_k x)$ and $q(z|x) = \prod_{k=1}^K q(z_k|x)$. By aligning $w_k = m_k$ and $V = S_k$, we can find $\mathbb{E}_{q(z|x_i)}[\log p(y_i|z_1, \dots z_K)] = \mathbb{E}_{q(\theta)}[\log p(y_i|(\theta_1^\top x_i, \dots, \theta_K^\top x_i))]$, and

$$\mathrm{KL}(q(\theta)||p(\theta)) = \sum_k \mathrm{KL}(q(\theta_k), p(\theta_k)) \doteq \sum_k \mathrm{KL}(q(z_k|X_N)||p(z_k|X_N)), \tag{15}$$

where the second $\doteq$ means equivalence ignoring a constant difference.

Proof of Theorem 3.2. Consider a neural network with one single hidden layer, denote the weights of the first layer as u, and the weights of the second layer as w. Thus, the k-th output can be computed as:

$$z^{(k)} = \sum_{i=1}^I w_{k,i}\psi\left(\sum_{d=1}^D u_{i,d}x_d\right),$$

where I is the size of the hidden layer, D is the input size and $\psi(a) = \max(0, a)$ is the ReLU activation function. We further simplify the setting by considering the special case where only x_1 is non-zero and $I = 1$. Then the k-th output becomes:

$$z^{(k)} = w_{k,1}\psi(u_{1,1}x_1).$$

Consider a distribution $q(w_{k,i}) = \mathcal{N}(\bar{w}_{k,i}, \sigma_w^2), q(u_{i,d}) = \mathcal{N}(\bar{u}_{i,d}, \sigma_u^2)$. Then if $x_1 \geq 0$,

$$\begin{aligned}\mathbb{E}_{q(w)q(u)}\left[z^{(k)}\right] &= \mathbb{E}_{w,u}\left[w_{k,1}\psi(u_{1,1}x_1)\right] \\ &= \bar{w}_{k,1}\left(\bar{u}_{1,1} + \frac{\phi\left(-\frac{\bar{u}_{1,1}}{\sigma_u}\right)}{1-\Phi\left(-\frac{\bar{u}_{1,1}}{\sigma_u}\right)}\sigma_u\right)\left(1-\Phi\left(-\frac{\bar{u}_{1,1}}{\sigma_u}\right)\right)x_1; \end{aligned} \tag{16}$$

if $x_1 < 0$, then

$$\begin{aligned}\mathbb{E}_{q(w)q(u)}\left[z^{(k)}\right] &= \mathbb{E}_{w,u}\left[w_{k,1}\psi(u_{1,1}x_1)\right] \\ &= \bar{w}_{k,1}\left(\bar{u}_{1,1} - \frac{\phi\left(-\frac{\bar{u}_{1,1}}{\sigma_u}\right)}{\Phi\left(-\frac{\bar{u}_{1,1}}{\sigma_u}\right)}\sigma_u\right)\Phi\left(-\frac{\bar{u}_{1,1}}{\sigma_u}\right)x_1, \end{aligned} \tag{17}$$

where ϕ and Φ are the pdf and cdf of standard normal distribution and we directly use the expectation of the truncated normal distribution. Now consider $\tilde{w}$ and $\tilde{u}$ that aim to recover (16) and (17). If $\tilde{u}_{1,1} \geq 0$, it cannot successfully recover (17) because the ReLU activation will have 0 when $x_1 < 0$ so that it cannot recover (17); if $\tilde{u}_{1,1} < 0$, for the same reason it cannot recover (16). □

B Derivations of Collapsed Variational Inference

As is shown by Tomczak et al. (2021), for priors and approximate posteriors from the exponential family, we can derive the closed-form solution for the optimal $q^*(\mu_p, \sigma_p^2)$,

$$\log q^*(\mu_p, \sigma_p^2|x) \propto \log p(\mu_p, \sigma_p^2) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_p, \sigma_p^2)], \tag{18}$$

for optimizing $q(\mu_p, \sigma_p^2)$ for every single data.

B.1 Learn mean, fix variance

Let $p(z|\mu_p) = \mathcal{N}(z|\mu_p, \gamma I)$, $p(\mu_p) = \mathcal{N}(\mu_p|0, \alpha I)$. Recall that $q(z|x) = \mathcal{N}(\mu_q(x), \mathrm{diag}(\sigma_q^2(x)))$. Then

$$\begin{aligned}\log q^*(\mu_p|x) &\propto \log p(\mu_p) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_p)] \\ &\propto -\frac{1}{2\alpha}\mu_p^\top\mu_p - \frac{1}{2\gamma}[(\mu_p - \mu_q(x))^\top(\mu_p - \mu_q(x)) + 1^\top\sigma_q^2(x)] \\ &\propto -\frac{\alpha+\gamma}{2\alpha\gamma}\left(\mu_p - \frac{\alpha}{\alpha+\gamma}\mu_q(x)\right)^\top\left(\mu_p - \frac{\alpha}{\alpha+\gamma}\mu_q(x)\right).\end{aligned}$$

Then $q^*(\mu_p) = \mathcal{N}(\frac{\alpha}{\alpha+\gamma}\mu_q(x), \frac{\alpha\gamma}{\alpha+\gamma}I)$. Pluging q^* into the regularizer, the new regularizer becomes

$$\frac{1}{2\gamma}\left[1^\top\sigma_q^2(x) + \frac{\gamma}{\gamma+\alpha}\mu_q(x)^\top\mu_q(x)\right] - \frac{1}{2}1^\top\log\sigma_q^2(x) + \frac{K}{2}\log(\gamma+\alpha) - \frac{K}{2}.$$

B.2 Learn both mean and variance

Let $p(z|\mu_p, \sigma_p^2) = \mathcal{N}(z|\mu_p, \sigma_p^2)$, $p(\mu_p|\sigma_p^2) = \mathcal{N}(\mu_p|0, \frac{1}{t}\sigma_p^2)$, $p(\sigma_p^2) = \mathcal{IG}(\sigma_p^2|\alpha, \beta)$, where $\mathcal{IG}$ indicates the inverse Gamma distribution. Let $q(\mu_p)$ be a diagonal Gaussian and $q(\sigma_p^2)$ be inverse Gamma. Use $\mu_{p,i}$ and $\sigma_{p,i}$ to denote the i-th entry of μ_p and σ_p respectively, then

$$\begin{aligned}&\log q^*(\mu_{p,i}, \sigma_{p,i}^2) \\ &\propto \log p(\mu_{p,i}, \sigma_{p,i}^2) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_p, \sigma_p^2)] \\ &\propto -(\alpha+\frac{3}{2})\log\sigma_{p,i}^2 - \frac{2\beta + t\mu_{p,i}^2}{2\sigma_{p,i}^2} - \frac{1}{2}\log\sigma_{p,i}^2 - \frac{1}{2}\frac{(\mu_{p,i} - \mu_{q,i}(x))^2}{\sigma_{p,i}^2} - \frac{1}{2}\frac{\sigma_{q,i}^2(x)}{\sigma_{p,i}^2} \\ &\propto -(\alpha+2)\log\sigma_{p,i}^2 - \frac{1}{2\sigma_{p,i}^2}\left(2\left(\beta + \frac{t}{2(t+1)}\mu_{q,i}(x)^2 + \frac{1}{2}\sigma_{q,i}^2(x)\right) + (t+1)\left(\mu_{p,i} - \frac{\mu_{q,i}(x)}{t+1}\right)^2\right)\end{aligned}$$

follows the normal-inverse-gamma distribution. Thus $q^*(\mu_p|x) = \mathcal{N}(\mu_p|\frac{1}{t+1}\mu_q(x), \frac{1}{t+1}\sigma_p^2)$ and $q^*(\sigma_p^2|x) = \mathcal{IG}(\sigma_p^2|(\alpha+\frac{1}{2})1, \beta + \frac{t}{2(t+1)}\mu_q(x)^2 + \frac{1}{2}\sigma_q^2(x))$. Then the regularizer becomes

$$(\alpha+\frac{1}{2})1^\top \log\left[\beta 1 + \frac{t}{2(1+t)}\mu_q(x)^2 + \frac{1}{2}\sigma_q^2(x)\right] - \frac{1}{2}1^\top \log\sigma_q^2(x). \tag{19}$$

B.3 Empirical Bayes

Let $p(\sigma_p^2) = \mathcal{IG}(\sigma_p^2|\alpha,\beta)$, $p(z|\sigma_p^2) = \mathcal{N}(z|0,\sigma_p^2 I)$, and let $q(\sigma_p^2)$ be a delta distribution. Then

$$\begin{aligned}
&\mathrm{KL}(q(z|x)||p(z|\sigma_p^2)) - \log p(\sigma_p^2) \\
=&\frac{1}{2}\left[K\log\sigma_p^2 - 1^\top\log\sigma_q^2(x) - K + \frac{1^\top\sigma_q^2(x)}{\sigma_p^2} + \frac{\mu_q(x)^\top\mu_q(x)}{\sigma_p^2}\right] + (\alpha+1)\log\sigma_p^2 + \frac{\beta}{\sigma_p^2}.
\end{aligned}$$

By taking the derivatives of the above equation with respect to σ_p^2 and solving, we obtain the optimal $\sigma_p^2 = \frac{\mu_q(x)^\top\mu_q(x) + 1^\top\sigma_q^2(x) + 2\beta}{K+2\alpha+2}$. If we plug this back into the KL term, we get the regularizer:

$$\begin{aligned}
&\frac{1}{2}\left[K\log\frac{\mu_q(x)^\top\mu_q(x) + 1^\top\sigma_q^2(x) + 2\beta}{K+2\alpha+2} - 1^\top\log|\sigma_q^2(x)|\right] \\
&- \frac{K}{2} + \frac{1}{2}\frac{(K+2\alpha+2)(\mu_q(x)^\top\mu_q(x) + 1^\top\sigma_q^2(x))}{\mu_q(x)^\top\mu_q(x) + 1^\top\sigma_q^2(x) + 2\beta}.
\end{aligned} \tag{20}$$

However, if we include the negative log-prior term $(\alpha+1)\log\frac{\mu_q(x)^\top\mu_q(x)+1^\top\sigma_q^2(x)+2\beta}{K+2\alpha+2} + \beta\frac{K+2\alpha+2}{\mu_q(x)^\top\mu_q(x)+1^\top\sigma_q^2(x)+2\beta}$, adding them up we will have

$$\frac{1}{2}(K+2\alpha+2)\log\frac{\mu_q(x)^\top\mu_q(x) + 1^\top\sigma_q^2(x) + 2\beta}{K+2\alpha+2} - 1^\top\log\sigma_q^2(x) + \text{const},$$

which highly reduces the complexity of the regularizer. This performs less well in practice and therefore we follow Wu et al. (2019) and use equation 20.

C Optimizing the Variational Distribution for All Data

In the previous section we show the derivation of collapsed variational inference where $q^*(\mu_p,\sigma_p^2)$ is optimized for every data point x. In this section we show how to optimize $q(\mu_p,\sigma_p^2)$ for all data and obtain different regularizers to the ones mentioned in the above section. These perform less well in practice but we include them here for completeness. The closed-form solution for $q^*(\mu_p,\sigma_p^2)$ for all data is

$$\log q^*(\mu_p,\sigma_p^2) \propto \frac{1}{N}\sum_{(x,y)\in\mathcal{D}}\left\{\log p(\mu_p,\sigma_p^2) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_p,\sigma_p^2)]\right\}. \tag{21}$$

C.1 Learn mean, fix variance, optimize for all data

Let $p(z|\mu_p) = \mathcal{N}(z|\mu_p, \gamma)$, $p(\mu_p) = \mathcal{N}(\mu_p|0, \alpha)$. Given a dataset $\mathcal{D} = \{(x, y)\}$, we can get one optimal $q^*(\mu_p)$ for all data. According to equation 21,

$$\begin{aligned}
\log q^*(\mu_p) &\propto \frac{1}{N} \sum_{(x,y)\in\mathcal{D}} \left\{ \log p(\mu_p) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_p)] \right\} \\
&\propto -\frac{1}{2\alpha}\mu_p^\top \mu_p - \frac{1}{2\gamma N} \sum_{(x,y)\in\mathcal{D}} ((\mu_p - \mu_q(x))^\top (\mu_p - \mu_q(x)) + 1^\top \sigma_q^2(x)) \\
&\propto -\frac{\alpha+\gamma}{2\alpha\gamma} \left(\mu_p - \frac{1}{N} \sum_{(x,y)\in\mathcal{D}} \mu_q(x) \right)^\top \left(\mu_p - \frac{1}{N} \sum_{(x,y)\in\mathcal{D}} \mu_q(x) \right).
\end{aligned}$$

Then the optimal $q^*(\mu_p) = \mathcal{N}(\frac{\alpha}{\alpha+\gamma}\frac{1}{N}\sum_x \mu_q(x), \frac{\alpha\gamma}{\alpha+\gamma} I)$. Let $\bar{\mu}_q = \frac{1}{N}\sum_x \mu_q(x)$. The regularizer now is:

$$\begin{aligned}
&\sum_{(x,y)} \left\{ \mathbb{E}_{q(\mu_p)}[\mathrm{KL}(q(z|x)||p(z|\mu_p,\gamma))] + \mathrm{KL}(q(\mu_p)||p(\mu_p)) \right\} \\
&= \sum_{(x,y)} \left\{ \mathbb{E}_{q(\mu_p)} \left[K\log\gamma - 1^\top \log\sigma_q^2(x) - K + \frac{1}{\gamma} 1^\top \sigma_q^2(x) - \frac{1}{\gamma}(\mu_q(x) - \mu_p)^\top (\mu_q(x) - \mu_p) \right] \right\} \\
&\quad + \frac{N}{2} \left[K\log\frac{\alpha+\gamma}{\gamma} - K + K\frac{\gamma}{\gamma+\alpha} + \frac{\alpha^2}{(\alpha+\gamma)^2} \bar{\mu}_q^\top \bar{\mu}_q \right] \\
&= \sum_{(x,y)} \left\{ \frac{1}{2\gamma}(1^\top \sigma_q^2(x) + \mu_q(x)^\top \mu_q(x)) - \frac{1}{2} 1^\top \log\sigma_q^2(x) \right\} - \frac{N}{2} \left(\frac{1}{\gamma} - \frac{1}{\alpha+\gamma} \right) \bar{\mu}_q^\top \bar{\mu}_q + \frac{NK}{2}\log(\alpha+\gamma) - \frac{NK}{2}.
\end{aligned} \tag{22}$$

We refer to this method as "vifo-mean_all".

C.2 Learn both mean and variance, optimize mean for single data, and variance for all data

Let $p(z|\mu_p, \sigma_p^2) = \mathcal{N}(z|\mu_p, \sigma_p^2)$, $p(\mu_p|\sigma_p^2) = \mathcal{N}(\mu_p|0, \frac{1}{t}\sigma_p^2)$, $p(\frac{1}{\sigma_p^2}) = \mathcal{IG}(\frac{1}{\sigma_p^2}|\alpha, \beta)$. Consider that

$$\log p(\mu_{p,i}, \sigma_{p,i}^2) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_{p,i}, \sigma_{p,i}^2)] \tag{23}$$

$$= \log p(\mu_{p,i}|\sigma_{p,i}^2) + \log p(\sigma_{p,i}^2) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_{p,i}, \sigma_{p,i}^2)] \tag{24}$$

$$\propto -\frac{t}{2}\frac{\mu_{p,i}^2}{\sigma_{p,i}^2} - \frac{1}{2}\log\sigma_{p,i}^2 - (\alpha+1)\log\sigma_{p,i}^2 - \frac{\beta}{\sigma_{p,i}^2} - \frac{1}{2}\log\sigma_{p,i}^2 - \frac{1}{2\sigma_{p,i}^2}((\mu_{p,i} - \mu_{q,i}(x))^2 + \sigma_{q,i}^2(x)) \tag{25}$$

$$= -\frac{t}{2}\frac{\mu_{p,i}^2}{\sigma_{p,i}^2} - \log\sigma_{p,i}^2 - (\alpha+1)\log\sigma_{p,i}^2 - \frac{\beta}{\sigma_{p,i}^2} - \frac{1}{2\sigma_{p,i}^2}((\mu_{p,i} - \mu_{q,i}(x))^2 + \sigma_{q,i}^2(x)), \tag{26}$$

$$= -\frac{t+1}{2\sigma_{p,i}^2}\left(\mu_{p,i} - \frac{1}{t+1}\mu_{q,i}(x) \right)^2 - \frac{t\mu_{q,i}^2(x)}{2(t+1)\sigma_{p,i}^2} - (\alpha+2)\log\sigma_{p,i}^2 - \frac{\beta}{\sigma_{p,i}^2} - \frac{\sigma_{q,i}^2(x)}{2\sigma_{p,i}^2} \tag{27}$$

Then by extracting the μ_p part from equation 27, we have

$$\log q^*(\mu_{p,i}|\sigma_{p,i}^2, x) \propto -\frac{t+1}{2\sigma_{p,i}^2}\left(\mu_{p,i} - \frac{1}{t+1}\mu_{q,i}(x) \right)^2,$$

and thus $q^*(\mu_p|x,\sigma_p^2) = \mathcal{N}(\mu_p|\frac{1}{t+1}\mu_q(x), \frac{1}{t+1}\sigma_p^2)$. Then we try to marginalize out μ_p to compute $q^*(\sigma_p^2)$.

$$\begin{aligned}
\log q^*(\sigma_{p,i}^2) &\propto \frac{1}{N}\sum_{(x,y)\in\mathcal{D}} \log \int \exp\left(\log p(\mu_{p,i},\sigma_{p,i}^2) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_{p,i},\sigma_{p,i}^2)]\right) d\mu_{p,i} \\
&\propto \frac{1}{N}\sum_{x,y\in\mathcal{D}} \left\{ \frac{1}{2}\log \int \exp\left(-\frac{t+1}{2\sigma_{p,i}^2}\left(\mu_{p,i} - \frac{1}{t+1}\mu_{q,i}(x)\right)^2\right) d\mu_{p,i} \right. \\
&\left. - \frac{t\mu_{q,i}^2(x)}{2(t+1)\sigma_{p,i}^2} - (\alpha+2)\log\sigma_{p,i}^2 - \frac{\beta}{\sigma_{p,i}^2} - \frac{\sigma_{q,i}^2(x)}{2\sigma_{p,i}^2} \right\} \\
&= -(\alpha+2)\log\sigma_{p,i}^2 - \frac{\beta}{\sigma_{p,i}^2} + \frac{1}{N}\sum_{(x,y)\in\mathcal{D}}\left(\frac{1}{2}\log\frac{2\pi\sigma_{p,i}^2}{t+1} - \frac{\sigma_{q,i}^2(x)}{2\sigma_{p,i}^2} - \frac{t\mu_{q,i}^2(x)}{2(t+1)\sigma_{p,i}^2}\right) \\
&= -(\alpha+\frac{3}{2})\log\sigma_{p,i}^2 - \frac{\beta + \frac{t}{2(t+1)}\frac{1}{N}\sum\mu_{q,i}^2(x) + \frac{1}{2N}\sum\sigma_{q,i}^2(x)}{\sigma_{p,i}^2},
\end{aligned}$$

and $q^*(\sigma_p^2) = \mathcal{IG}(\sigma_p^2|(\alpha+\frac{1}{2})1, \beta + \frac{t}{2(t+1)}\frac{1}{N}\sum_x \mu_q(x)^2 + \frac{1}{2}\frac{1}{N}\sum_x \sigma_q^2(x))$. Let $\tilde{\mu}_q = \sqrt{\frac{1}{N}\sum_x \mu_q(x)^2}$ and $\tilde{\sigma}_q = \sqrt{\frac{1}{N}\sum_x \sigma_q(x)^2}$, then the regularizer becomes:

$$(\alpha+\frac{1}{2})N1^\top \log\left[\beta 1 + \frac{t}{2(1+t)}\tilde{\mu}_q^2 + \frac{1}{2}\tilde{\sigma}_q^2\right] - \sum_{(x,y)}\frac{1}{2}1^\top\log\sigma_q^2(x) \tag{28}$$

$$+ KN\log\frac{\Gamma(\alpha)}{\Gamma(\alpha+\frac{1}{2})} - NK\alpha\log\beta + \frac{NK}{2}\log\frac{t+1}{t} - \frac{NK}{2}. \tag{29}$$

We refer to this method as "vifo-mv_all".

C.3 Empirical Bayes for all data

If we optimize σ_p^2 for all data, then we have

$$\begin{aligned}
&\sum_{(x,y)\in\mathcal{D}}\left\{\mathrm{KL}(q(z|x)||p(z|\sigma_p^2)) - \log p(\sigma_p^2)\right\} \\
&= \sum_{(x,y)\in\mathcal{D}}\left\{\frac{1}{2}\left[K\log\sigma_p^2 - 1^\top\log\sigma_q^2(x) - K + \frac{1^\top\sigma_q^2(x)}{\sigma_p^2} + \frac{\mu_q(x)^\top\mu_q(x)}{\sigma_p^2}\right] + (\alpha+1)\log\sigma_p^2 + \frac{\beta}{\sigma_p^2}\right\}
\end{aligned}$$

and the optimal variance being $\frac{\tilde{\mu}_q^\top\tilde{\mu}_q + 1^\top\tilde{\sigma}_p^2 + 2\beta}{K+2\alpha+2}$ where $\tilde{\mu}_q = \sqrt{\frac{1}{N}\sum_x \mu_q(x)^2}$ and $\tilde{\sigma}_q = \sqrt{\frac{1}{N}\sum_x \sigma_q(x)^2}$. The objective is:

$$\begin{aligned}
&\frac{NK}{2}\log\frac{2\beta + \tilde{\mu}_q^2 + \tilde{\sigma}_q^2}{K+2\alpha+2} - \frac{1}{2}\sum_x 1^\top\log\sigma_q(x)^2 - \frac{NK}{2} \\
&+\frac{1}{2}\frac{K+2\alpha+2}{2\beta+\tilde{\mu}_q^2+\tilde{\sigma}_q^2}\sum_x(\mu_q(x)^\top\mu_q(x) + 1^\top\sigma_q^2(x)).
\end{aligned}$$

This method is called "vifo-eb_all".

D Experimental Details

D.1 Experiments on Artificial Dataset

To generate Fig. 1 and Fig. 2, we generate 100 training data points $y = 2\sin x + 0.1\epsilon, \epsilon \sim \mathcal{N}(0,1)$, where $x_{\text{train}} \in [-\frac{3}{4}\pi, -\frac{1}{2}\pi] \cup [\frac{1}{2}\pi, \frac{3}{4}\pi]$ and $x_{\text{test}} \in [-\pi, \pi]$. We use a multilayer perceptron neural network with 5

layers, each layer containing 50 hidden units to fit the data. For VI, we pick the prior standard deviation to be 0.5 and for vifo-mean, we select $\gamma = 0.3, \frac{\gamma}{\alpha+\gamma} = 0.05$. For both models, we select the regularization parameter $\eta = 0.1$ and for VIFO we choose $\eta_{\text{aux}} = 1.0$. Notice that the prior is defined on $z = (m, l)$, which is hard to visualize directly. We instead draw multiple z's from prior, then draw multiple y's from likelihood $p(y|z)$ and plot them in Fig. 2.

D.2 Experiments on Large Image Datasets

In this section we elaborate the experimental details, including the choice of hyperparameters, learning rates and the number of training epochs.

Number of training epochs: We train all methods in 500 epochs.

Learning rate: For all methods other than SGD, SWA and SWAG, we use the Adam optimizer with learning rate 0.001.

VI and VIFO: We first list the choices of the variance for naive variational methods. The choice of prior variance significantly affects the performance. For image datasets with complex neural networks, the total prior variance of VI grows with the number of parameters so we have to pick a small variance and we use 0.05 following the setting of Wilson et al. (2022). Since VIFO samples in the output space which is small, using 0.05 regularizes too strongly and we therefore set a larger value of 1 for the variance.

For collapsed variational inference, we pick $\gamma = 0.3$, $\alpha_{reg} = \frac{\gamma}{\alpha+\gamma} = 0.05$ for learn-mean regularizer (vi-mean, vifo-mean, vifo-mean-all) and $\alpha = 0.5, \beta = 0.01, \delta = \frac{t}{1+t} = 0.1$ for learn-mean-variance regularizer (vi-mv, vifo-mv, vifo-mv_all), which exactly follows Tomczak et al. (2021). We pick $\alpha = 4.4798$ and $\beta = 10$ for empirical Bayes (vi-eb, vifo-eb, vifo-eb_all). The choice of α in empirical Bayes follows Wu et al. (2019) but the choice of β is unclear in Wu et al. (2019) so we just perform a simple search from $\{1, 10, 100\}$ and set $\beta = 10$ that yields the best result.

For both VI and VIFO, the regularization parameter η is fixed 0.1.

Hybrid Methods: The hybrid methods (SGD, SWA and SWAG) are not very stable so we have to tune learning rates carefully for each dataset. We choose the momentum to be 0.9 for all cases and list all other information in Table D.1. Notice that it is hard to train the hybrid methods on SVHN using AlexNet, so we initialize with a pre-trained model that is trained with a larger learning rate 0.1 to find a region with lower training loss, and then continue to optimize with the parameters listed in Table D.1.

Dropout: For dropout we add a dropout layer following each activation layer in the base model and set the dropout probability $p = 0.1$.

Repulsive Ensembles: Repulsive ensembles run multiple copies of the base model with a kernel base penalty to make sure the models are diverse. We use RBF kernel with lengthscale being the median of the square of the norm.

Dirichlet: Dirichlet-based models are deterministic and they interpret the output of the last layer as the parameters of dirichlet distributions, i.e., $\boldsymbol{\alpha}(x) = g(f_W(x))$, where g maps the output to positive real numbers. Hence we run the Dirichlet models with the setting of the base model. We next explain the setting of hyperparameters. As discussed by Bengs et al. (2022), the models of Sensoy et al. (2018); Charpentier et al. (2020) implicitly perform variational inference:

$$\mathbf{p} \sim \text{Dir}(\boldsymbol{\alpha}_0), \quad y|\mathbf{p} \sim \text{Cat}(\mathbf{p}), \tag{30}$$

and the ELBO becomes

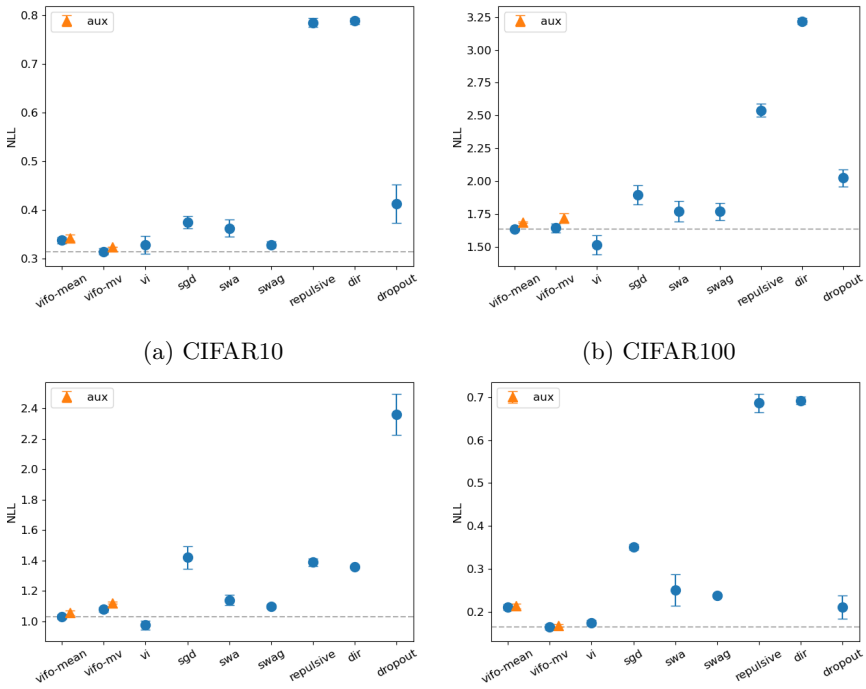
$$\log p(y|x) \geq \mathbb{E}_{q(\mathbf{p}|x)}[\log p(y|\mathbf{p})] - \text{KL}(q(\mathbf{p}|x)||\text{Dir}(\mathbf{p}|\boldsymbol{\alpha}_0)), \tag{31}$$

Table D.1: The parameters for running the hybrid algorithms. "lr"-learning rate, "wd"-weight decay, "swag_lr"-the learning rate after we start collecting models in SWA and SWAG algorithms, "swag_start"-the epochs when we start to collect models, "epochs"-the number of training epochs.

	lr	wd	swag_lr	swag_start	epochs
CIFAR10 / CIFAR100	0.05	0.0001	0.01	161	500
SVHN*	0.001	0.0001	0.005	161	500
STL10	0.05	0.001	0.01	161	500

where $q(\mathbf{p}|x) = \text{Dir}(\mathbf{p}|\boldsymbol{\alpha}(x))$. In the experiments, following Sensoy et al. (2018); Bengs et al. (2022), we use a uniform prior with $\boldsymbol{\alpha}_0 = [1, \ldots, 1]$. As in VI and VIFO, we pick the regularization parameter for KL divergence to be 0.1.

E Additional Plots



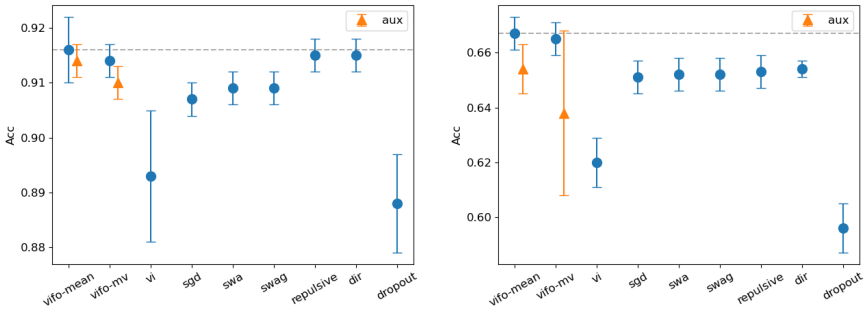
(a) CIFAR10

(b) CIFAR100

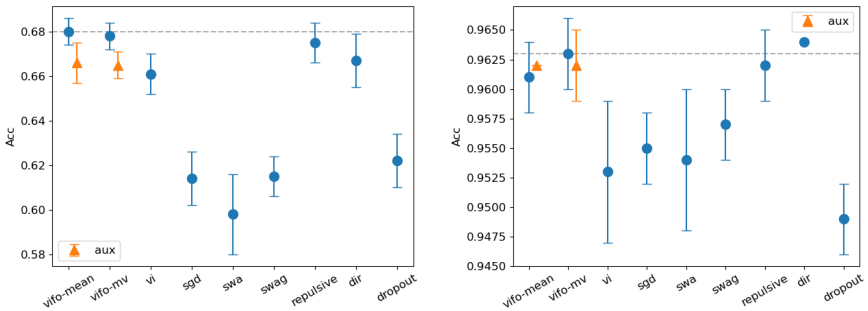
(c) STL10

(d) SVHN

Figure E.1: Test log loss of image datasets on AlexNet. Dashed lines indicate the best version of VIFO. The error bar is three times of the standard deviation for better visualization and same for other figures.

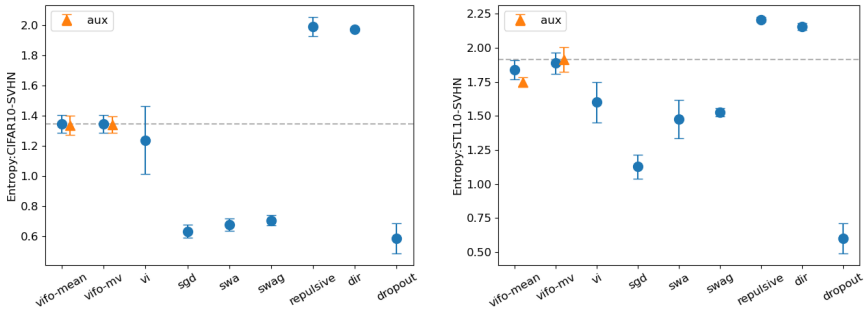


(a) CIFAR10 (b) CIFAR100

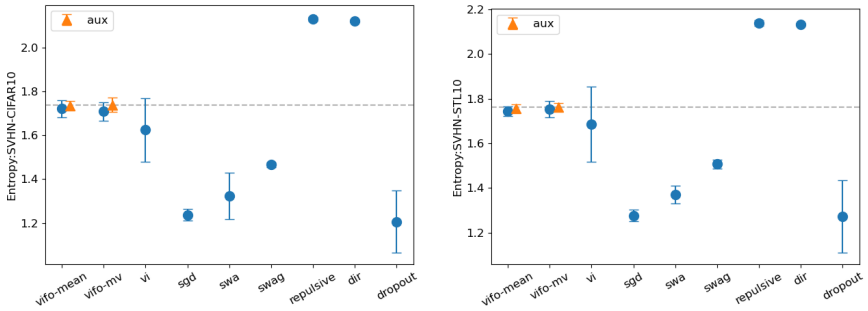


(c) STL10 (d) SVHN

Figure E.2: Test accuracy of image datasets on AlexNet. Dashed lines indicate the best version of VIFO.

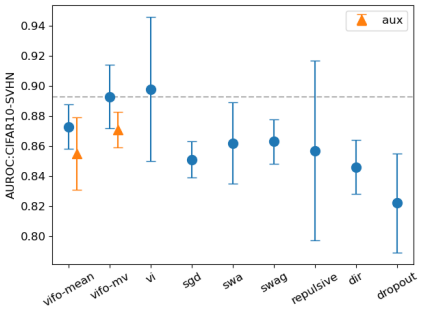


(a) CIFAR10→SVHN (b) STL10→SVHN

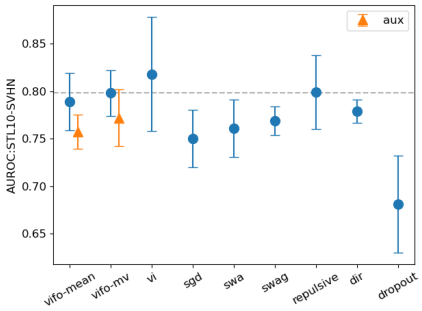


(c) SVHN→CIFAR10 (d) SVHN→STL10

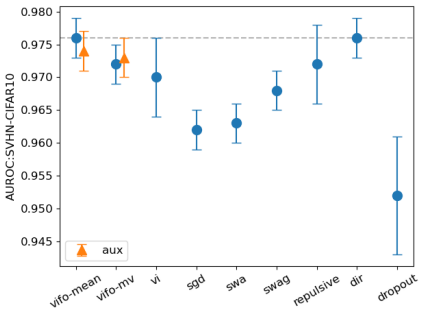
Figure E.3: Entropy (↑) on AlexNet.



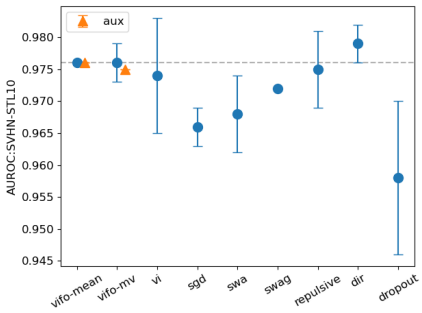
(a) CIFAR10→SVHN



(b) STL10→SVHN

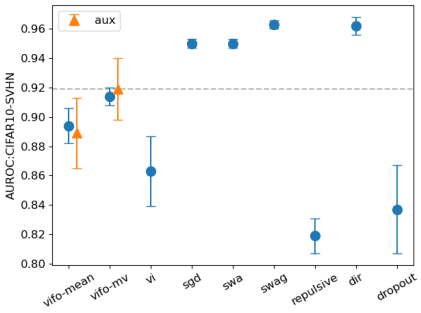


(c) SVHN→CIFAR10

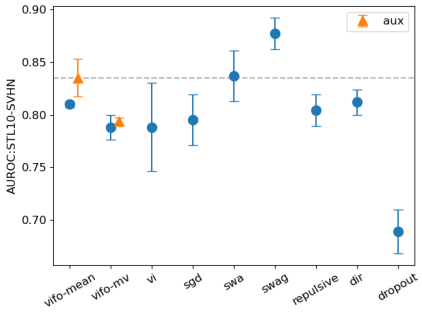


(d) SVHN→STL10

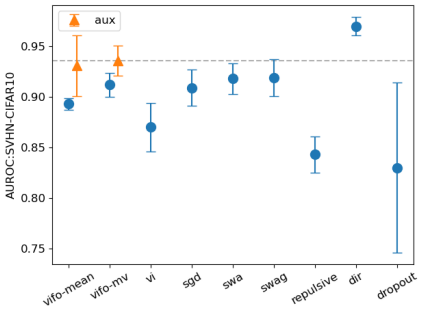
Figure E.4: AUROC (↑) on AlexNet.



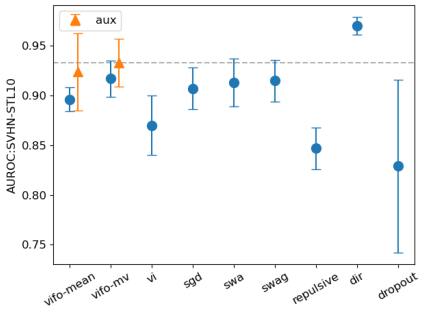
(a) CIFAR10→SVHN



(b) STL10→SVHN

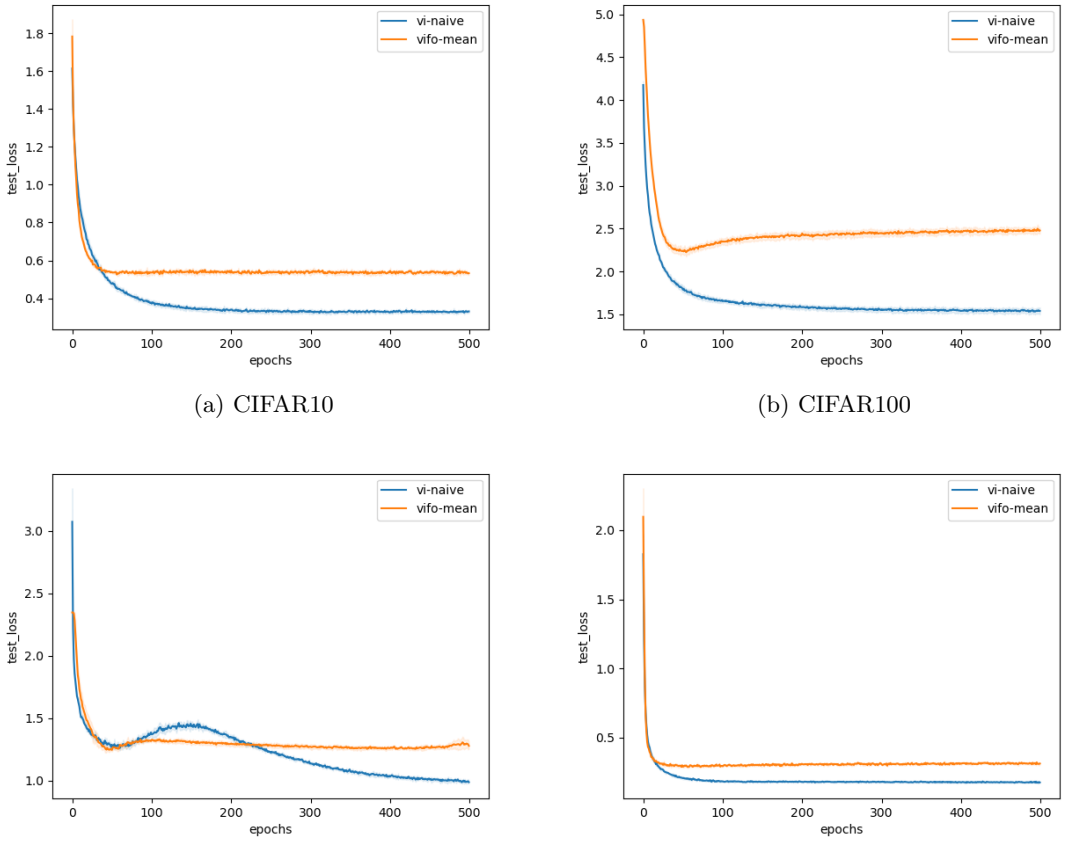


(c) SVHN→CIFAR10



(d) SVHN→STL10

Figure E.5: AUROC (↑) on PreResNet20.



(a) CIFAR10

(b) CIFAR100

(c) STL10

(d) SVHN

Figure E.6: Learning curves for all datasets on AlexNet. We conducted 5 independent runs and report the mean and standard deviation (which is very small). The results show that in all cases, VIFO-mean converges as quickly as, or faster than, VI.

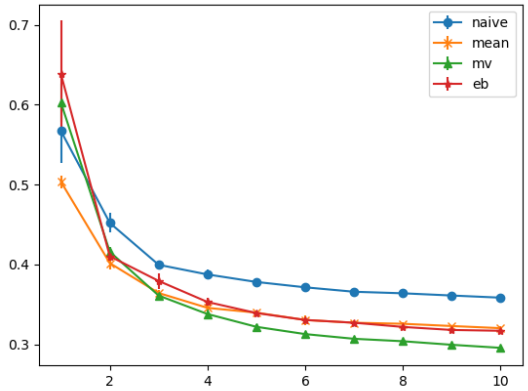


Figure E.7: Test losses vs. size of ensemble. Results are shown for 5 independent runs on CIFAR10 AlexNet. We can see when the number of ensembles is larger than 5, increasing the number of ensembles does not improve the performance significantly.

Table F.1: CIFAR10, AlexNet, NLL

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.388 ± 0.005	0.383 ± 0.002	0.382 ± 0.001	0.383 ± 0.002
vifo-mean	0.338 ± 0.002	0.343 ± 0.002	0.344 ± 0.003	0.345 ± 0.004
vifo-mv	0.315 ± 0.002	0.324 ± 0.000	0.315 ± 0.004	0.311 ± 0.002
vifo-eb	0.347 ± 0.003	0.345 ± 0.001	0.345 ± 0.003	0.347 ± 0.002
vi-naive	0.329 ± 0.006			
vi-mean	0.350 ± 0.010			
vi-mv	0.315 ± 0.013			
vi-eb	0.343 ± 0.004			
sgd	0.375 ± 0.004			
swa	0.363 ± 0.006			
swag	0.329 ± 0.002			
repulsive	0.785 ± 0.003			
dir	0.788 ± 0.002			
dropout	0.413 ± 0.013			

Table F.2: CIFAR100, AlexNet, NLL

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.774 ± 0.008	1.840 ± 0.016	1.784 ± 0.008	1.811 ± 0.012
vifo-mean	1.632 ± 0.003	1.687 ± 0.002	1.682 ± 0.017	1.683 ± 0.021
vifo-mv	1.642 ± 0.011	1.716 ± 0.012	1.971 ± 0.053	2.250 ± 0.105
vifo-eb	1.643 ± 0.008	1.651 ± 0.003	1.643 ± 0.004	1.649 ± 0.006
vi-naive	1.513 ± 0.024			
vi-mean	1.642 ± 0.023			
vi-mv	1.817 ± 0.022			
vi-eb	1.441 ± 0.016			
sgd	1.894 ± 0.024			
swa	1.768 ± 0.026			
swag	1.768 ± 0.022			
repulsive	2.540 ± 0.016			
dir	3.218 ± 0.008			
dropout	2.024 ± 0.022			

F Numerical Results

Table F.3: STL10, AlexNet, NLL

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.113 ± 0.006	1.117 ± 0.005	1.112 ± 0.005	1.135 ± 0.008
vifo-mean	1.030 ± 0.004	1.056 ± 0.005	1.066 ± 0.005	1.067 ± 0.011
vifo-mv	1.078 ± 0.005	1.121 ± 0.002	1.112 ± 0.010	1.101 ± 0.008
vifo-eb	1.141 ± 0.007	1.134 ± 0.008	1.127 ± 0.007	1.135 ± 0.002
vi-naive	0.975 ± 0.010			
vi-mean	1.021 ± 0.013			
vi-mv	1.095 ± 0.018			
vi-eb	1.560 ± 0.054			
sgd	1.419 ± 0.025			
swa	1.139 ± 0.011			
swag	1.098 ± 0.005			
repulsive	1.388 ± 0.008			
dir	1.359 ± 0.003			
dropout	2.359 ± 0.045			

Table F.4: SVHN, AlexNet, NLL

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.253 ± 0.001	0.256 ± 0.001	0.257 ± 0.001	0.251 ± 0.002
vifo-mean	0.211 ± 0.002	0.214 ± 0.002	0.209 ± 0.001	0.208 ± 0.001
vifo-mv	0.165 ± 0.002	0.169 ± 0.001	0.166 ± 0.001	0.170 ± 0.002
vifo-eb	0.211 ± 0.001	0.214 ± 0.002	0.212 ± 0.002	0.214 ± 0.001
vi-naive	0.175 ± 0.002			
vi-mean	0.219 ± 0.034			
vi-mv	0.173 ± 0.003			
vi-eb	0.182 ± 0.004			
sgd	0.351 ± 0.002			
swa	0.251 ± 0.012			
swag	0.238 ± 0.001			
repulsive	0.686 ± 0.007			
dir	0.692 ± 0.003			
dropout	0.211 ± 0.009			

Table F.5: CIFAR10, PreResNet20, NLL

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.473 ± 0.005	0.487 ± 0.011	0.486 ± 0.010	0.479 ± 0.008
vifo-mean	0.393 ± 0.010	0.433 ± 0.018	0.411 ± 0.013	0.430 ± 0.020
vifo-mv	0.361 ± 0.010	0.371 ± 0.010	0.367 ± 0.006	0.363 ± 0.016
vifo-eb	0.419 ± 0.014	0.429 ± 0.016	0.413 ± 0.007	0.425 ± 0.008
vi-naive	0.410 ± 0.028			
vi-mean	0.415 ± 0.032			
vi-mv	0.437 ± 0.029			
vi-eb	0.429 ± 0.035			
sgd	0.335 ± 0.013			
swa	0.336 ± 0.008			
swag	0.307 ± 0.010			
repulsive	0.875 ± 0.007			
dir	0.961 ± 0.022			
dropout	0.423 ± 0.026			

Table F.6: CIFAR100, PreResNet20, NLL

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.880 ± 0.009	1.935 ± 0.020	1.864 ± 0.008	1.844 ± 0.011
vifo-mean	1.632 ± 0.013	1.728 ± 0.005	1.686 ± 0.004	1.731 ± 0.018
vifo-mv	2.726 ± 0.021	2.826 ± 0.008	2.899 ± 0.019	2.867 ± 0.014
vifo-eb	2.076 ± 0.006	2.147 ± 0.004	2.340 ± 0.043	2.503 ± 0.034
vi-naive	1.642 ± 0.030			
vi-mean	1.753 ± 0.086			
vi-mv	1.804 ± 0.089			
vi-eb	1.731 ± 0.097			
sgd	1.445 ± 0.021			
swa	1.355 ± 0.023			
swag	1.354 ± 0.019			
repulsive	2.948 ± 0.020			
dir	3.580 ± 0.009			
dropout	1.644 ± 0.045			

Table F.7: STL10, PreResNet20, NLL

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.146 ± 0.012	1.159 ± 0.008	1.155 ± 0.015	1.145 ± 0.005
vifo-mean	1.067 ± 0.009	1.066 ± 0.011	1.056 ± 0.007	1.069 ± 0.009
vifo-mv	1.070 ± 0.017	1.078 ± 0.003	1.067 ± 0.019	1.073 ± 0.011
vifo-eb	1.162 ± 0.015	1.164 ± 0.020	1.191 ± 0.014	1.180 ± 0.016
vi-naive	0.920 ± 0.032			
vi-mean	1.000 ± 0.029			
vi-mv	1.002 ± 0.036			
vi-eb	1.018 ± 0.028			
sgd	1.203 ± 0.008			
swa	1.100 ± 0.006			
swag	1.100 ± 0.010			
repulsive	1.365 ± 0.009			
dir	1.418 ± 0.019			
dropout	1.665 ± 0.059			

Table F.8: SVHN, PreResNet20, NLL

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.391 ± 0.019	0.603 ± 0.057	0.486 ± 0.050	0.479 ± 0.018
vifo-mean	0.341 ± 0.004	0.357 ± 0.004	0.385 ± 0.015	0.334 ± 0.015
vifo-mv	0.269 ± 0.009	0.297 ± 0.010	0.323 ± 0.013	0.314 ± 0.009
vifo-eb	0.359 ± 0.002	0.391 ± 0.008	0.452 ± 0.018	0.402 ± 0.028
vi-naive	0.314 ± 0.024			
vi-mean	0.342 ± 0.040			
vi-mv	0.359 ± 0.033			
vi-eb	0.379 ± 0.057			
sgd	0.337 ± 0.011			
swa	0.320 ± 0.009			
swag	0.320 ± 0.009			
repulsive	0.822 ± 0.020			
dir	0.845 ± 0.031			
dropout	0.421 ± 0.068			

Table F.9: CIFAR10, AlexNet, Accuracy

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.914 ± 0.001	0.916 ± 0.001	0.915 ± 0.001	0.916 ± 0.001
vifo-mean	0.916 ± 0.002	0.914 ± 0.001	0.914 ± 0.001	0.912 ± 0.001
vifo-mv	0.914 ± 0.001	0.910 ± 0.001	0.912 ± 0.002	0.915 ± 0.002
vifo-eb	0.914 ± 0.002	0.914 ± 0.001	0.913 ± 0.001	0.914 ± 0.002
vi-naive	0.893 ± 0.004			
vi-mean	0.884 ± 0.004			
vi-mv	0.901 ± 0.003			
vi-eb	0.886 ± 0.003			
sgd	0.907 ± 0.001			
swa	0.909 ± 0.001			
swag	0.909 ± 0.001			
repulsive	0.915 ± 0.001			
dir	0.915 ± 0.001			
dropout	0.888 ± 0.003			

Table F.10: CIFAR100, AlexNet, Accuracy

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.658 ± 0.003	0.651 ± 0.004	0.657 ± 0.003	0.644 ± 0.003
vifo-mean	0.667 ± 0.002	0.654 ± 0.003	0.650 ± 0.007	0.658 ± 0.002
vifo-mv	0.665 ± 0.002	0.638 ± 0.010	0.568 ± 0.038	0.558 ± 0.028
vifo-eb	0.669 ± 0.002	0.671 ± 0.001	0.668 ± 0.003	0.664 ± 0.002
vi-naive	0.620 ± 0.003			
vi-mean	0.608 ± 0.002			
vi-mv	0.606 ± 0.002			
vi-eb	0.629 ± 0.004			
sgd	0.651 ± 0.002			
swa	0.652 ± 0.002			
swag	0.652 ± 0.002			
repulsive	0.653 ± 0.002			
dir	0.654 ± 0.001			
dropout	0.596 ± 0.003			

Table F.11: STL10, AlexNet, Accuracy

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.670 ± 0.003	0.672 ± 0.002	0.674 ± 0.002	0.665 ± 0.001
vifo-mean	0.680 ± 0.002	0.666 ± 0.003	0.669 ± 0.003	0.663 ± 0.003
vifo-mv	0.678 ± 0.002	0.665 ± 0.002	0.665 ± 0.002	0.668 ± 0.002
vifo-eb	0.662 ± 0.002	0.668 ± 0.002	0.670 ± 0.004	0.670 ± 0.002
vi-naive	0.661 ± 0.003			
vi-mean	0.649 ± 0.009			
vi-mv	0.644 ± 0.006			
vi-eb	0.414 ± 0.025			
sgd	0.614 ± 0.004			
swa	0.598 ± 0.006			
swag	0.615 ± 0.003			
repulsive	0.675 ± 0.003			
dir	0.667 ± 0.004			
dropout	0.622 ± 0.004			

Table F.12: SVHN, AlexNet, Accuracy

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.962 ± 0.001	0.962 ± 0.001	0.962 ± 0.000	0.962 ± 0.001
vifo-mean	0.961 ± 0.001	0.962 ± 0.000	0.962 ± 0.001	0.963 ± 0.001
vifo-mv	0.963 ± 0.001	0.962 ± 0.001	0.963 ± 0.000	0.962 ± 0.000
vifo-eb	0.962 ± 0.000	0.961 ± 0.001	0.961 ± 0.000	0.961 ± 0.000
vi-naive	0.953 ± 0.002			
vi-mean	0.945 ± 0.008			
vi-mv	0.955 ± 0.001			
vi-eb	0.950 ± 0.001			
sgd	0.955 ± 0.001			
swa	0.954 ± 0.002			
swag	0.957 ± 0.001			
repulsive	0.962 ± 0.001			
dir	0.964 ± 0.000			
dropout	0.949 ± 0.001			

Table F.13: CIFAR10, PreResNet20, Accuracy

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.897 ± 0.003	0.896 ± 0.002	0.893 ± 0.003	0.899 ± 0.003
vifo-mean	0.906 ± 0.002	0.900 ± 0.004	0.907 ± 0.002	0.904 ± 0.005
vifo-mv	0.903 ± 0.001	0.902 ± 0.002	0.902 ± 0.001	0.903 ± 0.002
vifo-eb	0.900 ± 0.003	0.900 ± 0.005	0.902 ± 0.002	0.901 ± 0.003
vi-naive	0.862 ± 0.009			
vi-mean	0.867 ± 0.009			
vi-mv	0.864 ± 0.010			
vi-eb	0.859 ± 0.011			
sgd	0.903 ± 0.003			
swa	0.902 ± 0.003			
swag	0.907 ± 0.002			
repulsive	0.903 ± 0.002			
dir	0.901 ± 0.002			
dropout	0.866 ± 0.007			

Table F.14: CIFAR100, PreResNet20, Accuracy

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.631 ± 0.004	0.620 ± 0.005	0.631 ± 0.002	0.628 ± 0.002
vifo-mean	0.642 ± 0.003	0.635 ± 0.003	0.643 ± 0.003	0.637 ± 0.006
vifo-mv	0.436 ± 0.005	0.413 ± 0.007	0.398 ± 0.006	0.405 ± 0.004
vifo-eb	0.579 ± 0.004	0.567 ± 0.009	0.535 ± 0.004	0.499 ± 0.017
vi-naive	0.620 ± 0.003			
vi-mean	0.608 ± 0.002			
vi-mv	0.606 ± 0.002			
vi-eb	0.629 ± 0.004			
sgd	0.638 ± 0.002			
swa	0.640 ± 0.002			
swag	0.642 ± 0.002			
repulsive	0.610 ± 0.005			
dir	0.603 ± 0.003			
dropout	0.565 ± 0.010			

Table F.15: STL10, PreResNet20, Accuracy

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.658 ± 0.004	0.658 ± 0.002	0.656 ± 0.001	0.656 ± 0.003
vifo-mean	0.660 ± 0.003	0.663 ± 0.003	0.669 ± 0.004	0.663 ± 0.004
vifo-mv	0.670 ± 0.004	0.663 ± 0.005	0.662 ± 0.005	0.662 ± 0.003
vifo-eb	0.661 ± 0.004	0.661 ± 0.004	0.657 ± 0.003	0.660 ± 0.005
vi-naive	0.690 ± 0.009			
vi-mean	0.674 ± 0.010			
vi-mv	0.688 ± 0.009			
vi-eb	0.638 ± 0.011			
sgd	0.672 ± 0.002			
swa	0.673 ± 0.003			
swag	0.679 ± 0.002			
repulsive	0.665 ± 0.005			
dir	0.668 ± 0.003			
dropout	0.621 ± 0.011			

Table F.16: SVHN, PreResNet20, Accuracy

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.930 ± 0.004	0.911 ± 0.014	0.912 ± 0.004	0.915 ± 0.008
vifo-mean	0.933 ± 0.004	0.929 ± 0.001	0.930 ± 0.001	0.936 ± 0.009
vifo-mv	0.934 ± 0.006	0.935 ± 0.005	0.928 ± 0.008	0.930 ± 0.002
vifo-eb	0.929 ± 0.004	0.917 ± 0.005	0.905 ± 0.003	0.924 ± 0.007
vi-naive	0.914 ± 0.009			
vi-mean	0.901 ± 0.015			
vi-mv	0.902 ± 0.012			
vi-eb	0.890 ± 0.019			
sgd	0.929 ± 0.007			
swa	0.936 ± 0.006			
swag	0.936 ± 0.006			
repulsive	0.931 ± 0.004			
dir	0.932 ± 0.002			
dropout	0.877 ± 0.023			

Table F.17: ECE:CIFAR10-STL10, AlexNet

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.042 ± 0.003	0.042 ± 0.002	0.047 ± 0.004	0.045 ± 0.003
vifo-mean	0.056 ± 0.002	0.053 ± 0.003	0.055 ± 0.001	0.059 ± 0.002
vifo-mv	0.067 ± 0.002	0.069 ± 0.002	0.068 ± 0.002	0.068 ± 0.003
vifo-eb	0.039 ± 0.003	0.042 ± 0.002	0.038 ± 0.002	0.038 ± 0.002
vi-naive	0.108 ± 0.002			
vi-mean	0.105 ± 0.007			
vi-mv	0.131 ± 0.003			
vi-eb	0.118 ± 0.004			
sgd	0.152 ± 0.002			
swa	0.150 ± 0.001			
swag	0.144 ± 0.001			
repulsive	0.246 ± 0.003			
dir	0.248 ± 0.002			
dropout	0.180 ± 0.002			

Table F.18: ECE:STL10-CIFAR10, AlexNet

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.053 ± 0.004	0.053 ± 0.004	0.040 ± 0.004	0.034 ± 0.004
vifo-mean	0.065 ± 0.002	0.063 ± 0.003	0.055 ± 0.003	0.046 ± 0.002
vifo-mv	0.075 ± 0.002	0.067 ± 0.003	0.078 ± 0.003	0.073 ± 0.002
vifo-eb	0.081 ± 0.004	0.078 ± 0.002	0.066 ± 0.002	0.075 ± 0.004
vi-naive	0.072 ± 0.007			
vi-mean	0.107 ± 0.007			
vi-mv	0.155 ± 0.004			
vi-eb	0.030 ± 0.006			
sgd	0.238 ± 0.007			
swa	0.137 ± 0.016			
swag	0.117 ± 0.007			
repulsive	0.197 ± 0.006			
dir	0.146 ± 0.004			
dropout	0.365 ± 0.007			

Table F.19: ECE:CIFAR10-STL10, PreResNet20

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.029 ± 0.001	0.032 ± 0.003	0.032 ± 0.004	0.037 ± 0.004
vifo-mean	0.036 ± 0.003	0.028 ± 0.005	0.032 ± 0.002	0.029 ± 0.003
vifo-mv	0.058 ± 0.003	0.047 ± 0.004	0.052 ± 0.002	0.051 ± 0.002
vifo-eb	0.024 ± 0.003	0.028 ± 0.005	0.031 ± 0.002	0.026 ± 0.002
vi-naive	0.126 ± 0.004			
vi-mean	0.147 ± 0.002			
vi-mv	0.159 ± 0.003			
vi-eb	0.137 ± 0.005			
sgd	0.092 ± 0.004			
swa	0.089 ± 0.001			
swag	0.078 ± 0.003			
repulsive	0.259 ± 0.004			
dir	0.319 ± 0.007			
dropout	0.146 ± 0.002			

Table F.20: ECE:STL10-CIFAR10, PreResNet20

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.021 ± 0.001	0.022 ± 0.002	0.025 ± 0.003	0.028 ± 0.004
vifo-mean	0.029 ± 0.003	0.024 ± 0.005	0.019 ± 0.002	0.019 ± 0.003
vifo-mv	0.088 ± 0.005	0.081 ± 0.003	0.081 ± 0.004	0.086 ± 0.002
vifo-eb	0.051 ± 0.005	0.039 ± 0.002	0.030 ± 0.003	0.038 ± 0.002
vi-naive	0.129 ± 0.003			
vi-mean	0.151 ± 0.004			
vi-mv	0.171 ± 0.011			
vi-eb	0.094 ± 0.005			
sgd	0.112 ± 0.002			
swa	0.107 ± 0.003			
swag	0.093 ± 0.004			
repulsive	0.185 ± 0.006			
dir	0.200 ± 0.005			
dropout	0.290 ± 0.007			

Table F.21: Entropy:CIFAR10-SVHN, AlexNet

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.357 ± 0.009	1.399 ± 0.023	1.286 ± 0.049	1.355 ± 0.033
vifo-mean	1.344 ± 0.020	1.336 ± 0.021	1.313 ± 0.042	1.342 ± 0.039
vifo-mv	1.344 ± 0.020	1.342 ± 0.018	1.354 ± 0.023	1.332 ± 0.017
vifo-eb	1.330 ± 0.045	1.330 ± 0.026	1.323 ± 0.017	1.296 ± 0.027
vi-naive	1.238 ± 0.075			
vi-mean	1.280 ± 0.157			
vi-mv	1.053 ± 0.068			
vi-eb	1.133 ± 0.069			
sgd	0.633 ± 0.014			
swa	0.676 ± 0.014			
swag	0.705 ± 0.011			
repulsive	1.990 ± 0.021			
dir	1.970 ± 0.002			
dropout	0.585 ± 0.033			

Table F.22: Entropy:STL10-SVHN, AlexNet

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.772 ± 0.017	1.773 ± 0.015	1.779 ± 0.024	1.763 ± 0.011
vifo-mean	1.840 ± 0.023	1.748 ± 0.012	1.818 ± 0.029	1.826 ± 0.015
vifo-mv	1.889 ± 0.026	1.915 ± 0.031	1.848 ± 0.051	1.859 ± 0.018
vifo-eb	1.764 ± 0.026	1.754 ± 0.018	1.798 ± 0.028	1.716 ± 0.012
vi-naive	1.601 ± 0.049			
vi-mean	1.495 ± 0.018			
vi-mv	1.345 ± 0.046			
vi-eb	2.024 ± 0.019			
sgd	1.127 ± 0.030			
swa	1.479 ± 0.047			
swag	1.525 ± 0.010			
repulsive	2.208 ± 0.006			
dir	2.157 ± 0.008			
dropout	0.601 ± 0.037			

Table F.23: Entropy:SVHN-CIFAR10, AlexNet

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.694 ± 0.008	1.711 ± 0.005	1.694 ± 0.003	1.693 ± 0.002
vifo-mean	1.721 ± 0.013	1.735 ± 0.007	1.729 ± 0.009	1.742 ± 0.008
vifo-mv	1.709 ± 0.014	1.738 ± 0.011	1.754 ± 0.014	1.758 ± 0.011
vifo-eb	1.740 ± 0.015	1.749 ± 0.005	1.739 ± 0.007	1.773 ± 0.033
vi-naive	1.624 ± 0.048			
vi-mean	1.711 ± 0.052			
vi-mv	1.448 ± 0.036			
vi-eb	1.577 ± 0.058			
sgd	1.237 ± 0.009			
swa	1.323 ± 0.035			
swag	1.465 ± 0.004			
repulsive	2.129 ± 0.003			
dir	2.121 ± 0.002			
dropout	1.206 ± 0.047			

Table F.24: Entropy:SVHN-STL10, AlexNet

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.720 ± 0.007	1.734 ± 0.005	1.711 ± 0.004	1.708 ± 0.008
vifo-mean	1.744 ± 0.007	1.756 ± 0.006	1.753 ± 0.011	1.756 ± 0.007
vifo-mv	1.754 ± 0.012	1.763 ± 0.006	1.804 ± 0.017	1.790 ± 0.009
vifo-eb	1.764 ± 0.016	1.776 ± 0.007	1.775 ± 0.017	1.765 ± 0.030
vi-naive	1.685 ± 0.056			
vi-mean	1.772 ± 0.042			
vi-mv	1.512 ± 0.041			
vi-eb	1.631 ± 0.047			
sgd	1.277 ± 0.009			
swa	1.371 ± 0.013			
swag	1.507 ± 0.007			
repulsive	2.138 ± 0.004			
dir	2.132 ± 0.002			
dropout	1.272 ± 0.054			

Table F.25: Entropy:CIFAR10-SVHN, PreResNet20

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.465 ± 0.007	1.516 ± 0.009	1.553 ± 0.006	1.540 ± 0.004
vifo-mean	1.469 ± 0.009	1.553 ± 0.018	1.599 ± 0.005	1.580 ± 0.010
vifo-mv	1.559 ± 0.013	1.611 ± 0.016	1.590 ± 0.017	1.543 ± 0.015
vifo-eb	1.532 ± 0.011	1.540 ± 0.013	1.582 ± 0.018	1.548 ± 0.005
vi-naive	1.192 ± 0.025			
vi-mean	1.097 ± 0.015			
vi-mv	1.048 ± 0.019			
vi-eb	1.174 ± 0.049			
sgd	1.398 ± 0.009			
swa	1.398 ± 0.015			
swag	1.586 ± 0.016			
repulsive	1.971 ± 0.006			
dir	2.246 ± 0.002			
dropout	0.995 ± 0.022			

Table F.26: Entropy:STL10-SVHN, PreResNet20

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.763 ± 0.008	1.770 ± 0.005	1.742 ± 0.009	1.728 ± 0.007
vifo-mean	1.796 ± 0.006	1.908 ± 0.008	1.876 ± 0.002	1.862 ± 0.008
vifo-mv	1.565 ± 0.007	1.607 ± 0.004	1.588 ± 0.005	1.574 ± 0.008
vifo-eb	1.603 ± 0.005	1.637 ± 0.011	1.634 ± 0.008	1.656 ± 0.017
vi-naive	1.394 ± 0.020			
vi-mean	1.323 ± 0.013			
vi-mv	1.267 ± 0.030			
vi-eb	1.517 ± 0.060			
sgd	1.432 ± 0.022			
swa	1.462 ± 0.013			
swag	1.579 ± 0.013			
repulsive	2.160 ± 0.005			
dir	2.194 ± 0.004			
dropout	0.830 ± 0.024			

Table F.27: Entropy:SVHN-CIFAR10, PreResNet20

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.531 ± 0.008	2.019 ± 0.026	1.942 ± 0.051	1.959 ± 0.039
vifo-mean	1.515 ± 0.012	1.908 ± 0.080	1.850 ± 0.065	1.780 ± 0.028
vifo-mv	1.568 ± 0.014	1.758 ± 0.051	1.848 ± 0.067	1.706 ± 0.057
vifo-eb	1.539 ± 0.004	1.775 ± 0.038	1.898 ± 0.022	1.923 ± 0.015
vi-naive	1.213 ± 0.010			
vi-mean	1.169 ± 0.017			
vi-mv	1.080 ± 0.011			
vi-eb	1.219 ± 0.029			
sgd	1.384 ± 0.005			
swa	1.403 ± 0.002			
swag	1.476 ± 0.005			
repulsive	2.006 ± 0.013			
dir	2.250 ± 0.001			
dropout	1.059 ± 0.019			

Table F.28: Entropy:SVHN-STL10, PreResNet20

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	1.529 ± 0.016	1.995 ± 0.042	1.860 ± 0.041	1.885 ± 0.054
vifo-mean	1.543 ± 0.012	1.837 ± 0.070	1.875 ± 0.066	1.762 ± 0.052
vifo-mv	1.588 ± 0.010	1.763 ± 0.061	1.875 ± 0.041	1.671 ± 0.039
vifo-eb	1.544 ± 0.004	1.724 ± 0.045	1.895 ± 0.054	1.785 ± 0.028
vi-naive	1.222 ± 0.013			
vi-mean	1.174 ± 0.016			
vi-mv	1.078 ± 0.014			
vi-eb	1.219 ± 0.024			
sgd	1.375 ± 0.007			
swa	1.389 ± 0.011			
swag	1.464 ± 0.005			
repulsive	2.014 ± 0.011			
dir	2.252 ± 0.002			
dropout	1.053 ± 0.014			

Table F.29: AUROC:CIFAR10-SVHN, AlexNet

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.860 ± 0.005	0.857 ± 0.008	0.833 ± 0.018	0.853 ± 0.018
vifo-mean	0.873 ± 0.005	0.855 ± 0.008	0.858 ± 0.007	0.856 ± 0.008
vifo-mv	0.893 ± 0.007	0.871 ± 0.004	0.873 ± 0.004	0.861 ± 0.006
vifo-eb	0.860 ± 0.007	0.859 ± 0.007	0.864 ± 0.002	0.844 ± 0.010
vi-naive	0.898 ± 0.016			
vi-mean	0.886 ± 0.029			
vi-mv	0.893 ± 0.009			
vi-eb	0.885 ± 0.010			
sgd	0.851 ± 0.004			
swa	0.862 ± 0.009			
swag	0.863 ± 0.005			
repulsive	0.857 ± 0.020			
dir	0.846 ± 0.006			
dropout	0.822 ± 0.011			

Table F.30: AUROC:STL10-SVHN, AlexNet

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.787 ± 0.005	0.776 ± 0.009	0.781 ± 0.010	0.755 ± 0.015
vifo-mean	0.789 ± 0.010	0.757 ± 0.006	0.768 ± 0.013	0.774 ± 0.006
vifo-mv	0.798 ± 0.008	0.772 ± 0.010	0.764 ± 0.016	0.779 ± 0.013
vifo-eb	0.793 ± 0.007	0.793 ± 0.010	0.769 ± 0.009	0.758 ± 0.005
vi-naive	0.818 ± 0.020			
vi-mean	0.792 ± 0.011			
vi-mv	0.775 ± 0.018			
vi-eb	0.736 ± 0.044			
sgd	0.750 ± 0.010			
swa	0.761 ± 0.010			
swag	0.769 ± 0.005			
repulsive	0.799 ± 0.013			
dir	0.779 ± 0.004			
dropout	0.681 ± 0.017			

Table F.31: AUROC:SVHN-CIFAR10, AlexNet

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.969 ± 0.001	0.968 ± 0.001	0.967 ± 0.001	0.967 ± 0.001
vifo-mean	0.976 ± 0.001	0.974 ± 0.001	0.973 ± 0.001	0.973 ± 0.002
vifo-mv	0.972 ± 0.001	0.973 ± 0.001	0.972 ± 0.000	0.973 ± 0.001
vifo-eb	0.969 ± 0.001	0.971 ± 0.001	0.970 ± 0.001	0.973 ± 0.003
vi-naive	0.970 ± 0.002			
vi-mean	0.963 ± 0.014			
vi-mv	0.965 ± 0.003			
vi-eb	0.967 ± 0.003			
sgd	0.962 ± 0.001			
swa	0.963 ± 0.001			
swag	0.968 ± 0.001			
repulsive	0.972 ± 0.002			
dir	0.976 ± 0.001			
dropout	0.952 ± 0.003			

Table F.32: AUROC:SVHN-STL10, AlexNet

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.971 ± 0.000	0.972 ± 0.001	0.969 ± 0.001	0.971 ± 0.001
vifo-mean	0.976 ± 0.000	0.976 ± 0.000	0.977 ± 0.001	0.977 ± 0.001
vifo-mv	0.976 ± 0.001	0.975 ± 0.000	0.977 ± 0.001	0.976 ± 0.000
vifo-eb	0.974 ± 0.001	0.974 ± 0.001	0.973 ± 0.001	0.977 ± 0.002
vi-naive	0.974 ± 0.003			
vi-mean	0.968 ± 0.013			
vi-mv	0.970 ± 0.002			
vi-eb	0.971 ± 0.002			
sgd	0.966 ± 0.001			
swa	0.968 ± 0.002			
swag	0.972 ± 0.000			
repulsive	0.975 ± 0.002			
dir	0.979 ± 0.001			
dropout	0.958 ± 0.004			

Table F.33: AUROC:CIFAR10-SVHN, PreResNet20

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.885 ± 0.004	0.880 ± 0.006	0.889 ± 0.005	0.898 ± 0.006
vifo-mean	0.894 ± 0.004	0.889 ± 0.008	0.907 ± 0.007	0.896 ± 0.002
vifo-mv	0.914 ± 0.002	0.919 ± 0.007	0.917 ± 0.001	0.915 ± 0.003
vifo-eb	0.908 ± 0.002	0.912 ± 0.005	0.918 ± 0.002	0.913 ± 0.004
vi-naive	0.863 ± 0.008			
vi-mean	0.868 ± 0.011			
vi-mv	0.868 ± 0.008			
vi-eb	0.858 ± 0.013			
sgd	0.950 ± 0.001			
swa	0.950 ± 0.001			
swag	0.963 ± 0.001			
repulsive	0.819 ± 0.004			
dir	0.962 ± 0.002			
dropout	0.837 ± 0.010			

Table F.34: AUROC:STL10-SVHN, PreResNet20

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.810 ± 0.005	0.810 ± 0.004	0.804 ± 0.004	0.803 ± 0.003
vifo-mean	0.810 ± 0.001	0.835 ± 0.006	0.836 ± 0.004	0.820 ± 0.003
vifo-mv	0.788 ± 0.004	0.794 ± 0.001	0.789 ± 0.003	0.784 ± 0.005
vifo-eb	0.799 ± 0.004	0.800 ± 0.005	0.803 ± 0.002	0.803 ± 0.004
vi-naive	0.788 ± 0.014			
vi-mean	0.784 ± 0.006			
vi-mv	0.788 ± 0.015			
vi-eb	0.728 ± 0.030			
sgd	0.795 ± 0.008			
swa	0.837 ± 0.008			
swag	0.877 ± 0.005			
repulsive	0.804 ± 0.005			
dir	0.812 ± 0.004			
dropout	0.689 ± 0.007			

Table F.35: AUROC:SVHN-CIFAR10, PreResNet20

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.890 ± 0.005	0.958 ± 0.003	0.934 ± 0.010	0.948 ± 0.004
vifo-mean	0.893 ± 0.002	0.931 ± 0.010	0.925 ± 0.006	0.928 ± 0.014
vifo-mv	0.912 ± 0.004	0.936 ± 0.005	0.927 ± 0.007	0.916 ± 0.006
vifo-eb	0.916 ± 0.007	0.922 ± 0.002	0.923 ± 0.008	0.941 ± 0.009
vi-naive	0.870 ± 0.008			
vi-mean	0.867 ± 0.016			
vi-mv	0.866 ± 0.012			
vi-eb	0.847 ± 0.026			
sgd	0.909 ± 0.006			
swa	0.918 ± 0.005			
swag	0.919 ± 0.006			
repulsive	0.843 ± 0.006			
dir	0.970 ± 0.003			
dropout	0.830 ± 0.028			

Table F.36: AUROC:SVHN-STL10, PreResNet20

method	$\eta_{\text{aux}} = 0$	$\eta_{\text{aux}} = 0.1$	$\eta_{\text{aux}} = 0.5$	$\eta_{\text{aux}} = 1.0$
vifo-naive	0.894 ± 0.009	0.957 ± 0.009	0.934 ± 0.008	0.936 ± 0.010
vifo-mean	0.896 ± 0.004	0.924 ± 0.013	0.914 ± 0.010	0.921 ± 0.009
vifo-mv	0.917 ± 0.006	0.933 ± 0.008	0.928 ± 0.009	0.922 ± 0.001
vifo-eb	0.911 ± 0.006	0.916 ± 0.005	0.923 ± 0.004	0.936 ± 0.006
vi-naive	0.870 ± 0.010			
vi-mean	0.867 ± 0.016			
vi-mv	0.865 ± 0.011			
vi-eb	0.846 ± 0.027			
sgd	0.907 ± 0.007			
swa	0.913 ± 0.008			
swag	0.915 ± 0.007			
repulsive	0.847 ± 0.007			
dir	0.970 ± 0.003			
dropout	0.829 ± 0.029			