# Variational Inference on the Final-Layer Output of Neural Networks

Yadi Wei[1], Roni Khardon[1]

[1]Indiana University
weiyadi@iu.edu, rkhardon@iu.edu

**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. Like standard, non-Beyesian models, VIFO enjoys simple training and one can use Rademacher complexity to provide risk bounds for the model. On the other hand, using the Bayesian formulation we incorporate collapsed variational inference with VIFO which significantly improves the performance in practice. Experiments show that VIFO and ensembles of VIFO provide 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, due to its simplicity, one can derive risk bounds for the model through Rademacher complexity. Second, we show that, for the linear case, with expressive priors VIFO can capture the same predictions as standard variational inference. With practical priors and deep networks the model can be seen as a simpler alternative. Third, 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. VIFO was motivated as an effective simplification of VI and DVI. 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*), as well as ensemble variants of these 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, ensembles of VIFO provide 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

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

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:

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

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*.

Note that in regression, the prediction over $y$ given by $\int q(z|x)p(y|z)dz$ is different from the existing models known as the mean-variance estimator (Kabir et al. 2018; Khosravi et al. 2011; Kendall and 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:

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

VIFO is motivated from DVI. The approach has some similarity to Dirichlet-based models (Sensoy, Kaplan, and Kandemir 2018; Charpentier, Zügner, and Günnemann 2020; Bengs, Hüllermeier, and Waegeman 2022). However, we perform inference on the output layer whereas, as discussed by Bengs, Hüllermeier, and Waegeman (2022), these models implicitly perform variational inference on the prediction layer. 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:

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

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 $\text{KL}(q(z|x)||p(z))$ as a regularizer. In practice, we put a coefficient $\eta$ on the regularizer, as is often done in variational approximations (e.g., (Higgins et al. 2017; Jankowiak, Pleiss, and Gardner 2020; Wenzel et al. 2020)). Then, from a regularized loss minimization view, our objective becomes:

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

In most cases, the loss term is intractable, so we use Monte Carlo samples to approximate it, where reparametrization is used to reduce the variance in gradients:

$$\mathbb{E}_{q(z|x)}[-\log p(y|z)] \approx \frac{1}{M}\sum_{m=1}^{M} -\log p(y|z^{(m)}), \quad (5)$$

where $z^{(m)} \sim q(z|x)$. Equations (4) and (5) define the optimization objective of VIFO.

## 3 Rademacher Complexity of VIFO

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

**Assumption 3.1.** $\log p(y|z)$ is $L_0$-Lipschitz in $z$, i.e., $|\log p(y|z) - \log p(y|z')| \leq L_0\|z - z'\|_2$.

**Assumption 3.2.** The link function $g$ is $L_1$-Lipschitz.

We show in the Appendix that these assumptions hold for classification and with a smoothed loss for regression. With the two assumptions, we derive the main result:

**Theorem 3.3.** *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 $R(l \circ (\mathcal{H} \times \mathcal{H}) \circ S) \leq (L_0 \max\{1, L_1\}K) \cdot R(\mathcal{H} \circ S)$, 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 $\mu$ and $s$ and the same for $q_{W'}(z|x)$. We use $\cdot$ to denote 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 $\mu$, 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 $f_{W_1}(x)$ and $f_{W_2}(x)$. The theorem follows from Rademacher complexity bounds for bivariate and multivariate functions (Appendix, Lemma A.1, Corollary A.2). □

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 and 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, Rakhlin, and Shamir 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.

## 4 Comparison of VIFO and VI

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 $\theta$, then the model is:

$$y|x, \theta \sim p(y|\theta^\top x). \tag{6}$$

For example, $p(y|\theta^\top x) = \mathcal{N}(y|\theta^\top x, \frac{1}{\beta})$ where $\beta$ 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 $\theta$ 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 $Y_N = (y_1, y_2, \ldots, y_N) \in \mathbb{R}^N$,

$$
\begin{aligned}
&\sum_{i=1}^N \mathbb{E}_{q(\theta)}[\log p(y_i|\theta^\top x_i)] - \mathrm{KL}(q(\theta)||p(\theta)) \\
=&\sum_{i=1}^N \mathbb{E}_{q(\theta)}[\log p(y_i|\theta^\top x_i)] - \frac{1}{2}[\mathrm{tr}(S_0^{-1}S) - \log|S_0^{-1}S|] \\
&- \frac{1}{2}(m-m_0)^\top S_0^{-1}(m-m_0) + \frac{d}{2}.
\end{aligned} \tag{7}
$$

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 4.1.** *Let $q(z|x) = \mathcal{N}(z|w^\top x, x^\top Vx)$ 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 4.2.** *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 significance of these results is twofold. On the one hand, we see from Theorem 4.2 and the conditions of Theorem 4.1 that the representation is more limited, i.e., efficiency comes at some cost. On the other hand, Theorem 4.1 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 as we do in the next section.

In practice, computing 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 ensembles of VIFO.

## 5 Collapsed VI Applied to 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. In this section, we show how this idea is applicable in VIFO. In addition to $z$, we model the prior mean $\mu_p$ and variance $\sigma_p^2$ 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 $q(z|x)q(\mu_p, \sigma_p^2)$. Then the objective becomes:

$$\log p(y|x) \geq \mathbb{E}_{q(z|x)q(\mu_p,\sigma_p^2)}\left[\log \frac{p(y, z, \mu_p, \sigma_p^2|x)}{q(z|x)q(\mu_p, \sigma_p^2)}\right]$$
$$= \mathbb{E}_{q(z|x)}[\log p(y|z)] - \mathbb{E}_{q(\mu_p,\sigma_p^2)}[\mathrm{KL}(q(z|x)||p(z|\mu_p, \sigma_p^2))]$$
$$- \mathrm{KL}(q(\mu_p, \sigma_p^2)||p(\mu_p, \sigma_p^2)). \quad (8)$$

Similar to Eq (4), we treat the first term as a loss and the other two terms as a regularizer along with a coefficient $\eta$ and aggregate over all data. Since the loss does not contain $\mu_p$ and $\sigma_p^2$, we can get the optimal $q^*(\mu_p, \sigma_p^2)$ by optimizing the regularizer and the choice of $\eta$ will not affect $q^*(\mu_p, \sigma_p^2)$. Then we can plug in the value of $q^*$ into Eq (8). 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 Eq (8), the regularizer becomes:

$$\frac{1}{2\gamma}\left[1^\top\sigma_q^2(x) + \frac{\gamma}{\gamma+\alpha}\mu_q(x)^\top\mu_q(x)\right] - \frac{1}{2}1^\top\log\sigma_q^2(x)$$
$$+\frac{K}{2}\log(\gamma+\alpha) - \frac{K}{2}. \quad (9)$$

As in (Tomczak et al. 2021), Eq (9) 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".

**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 $p(\sigma_p^2) = \mathcal{IG}(\sigma_p^2|\alpha, \beta)$, $p(z|\sigma_p^2) = \mathcal{N}(z|0, \sigma_p^2)$ which yields "vifo-eb". Derivations are given in Appendix.

## 6 Related Work

VIFO shares some aspects with the model of Kendall and Gal (2017), where both use neural networks to output the mean and covariance of the last layer. However Kendall and Gal (2017) use the cross entropy loss, $-\log \frac{1}{M}\sum_m p(y|z^{(m)})$ instead of our loss in Eq (5), 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. Dirichlet-based methods (Sensoy, Kaplan, and Kandemir 2018; Charpentier, Zügner, and Günnemann 2020; Bengs, Hüllermeier, and Waegeman 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, Swaroop, and Turner 2020; Oleksiienko, Tran, and Iosifidis 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, Hernández-Lobato, and Turner 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 and Ghahramani 2016) randomly sets weights 0 to capture uncertainty in the model. Deep ensembles (Lakshminarayanan, Pritzel, and Blundell 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.

## 7 Experiments

In this section, we compare the empirical performance of VIFO with VI and hybrid methods that use the base model, as well as dropout (Gal and Ghahramani 2016) and the Dirichlet-based model (Sensoy, Kaplan, and Kandemir 2018). VI candidates include the VI algorithm (Blundell et al. 2015) with fixed prior parameters (called vi-naive below), 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, 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). In addition to comparing the single-model performance, for each model we also run an ensemble extension. Specifically, our comparison includes the ensemble of the base models which are known as deep ensembles (Lakshminarayanan, Pritzel, and Blundell 2017), and the ensemble of SWAG models, which is the multiSWAG model of (Wilson and Izmailov 2020), both of which are considered strong baselines for uncertainty quantification (Ovadia et al. 2019). Our main goal is to show:

- VIFO is much faster than VI and only slightly slower than base models;
- Ensembles of VIFO achieve better uncertainty quantification on shifted and out-of-distribution data than all baselines including VI and all ensembles of baselines except for ensembles of VI;
- VIFO preserves the quality of in-distribution predictions.

For our main experiments, we pick four large datasets, CIFAR10, CIFAR100, SVHN, STL10, together with two types of neural networks, AlexNet (Krizhevsky, Sutskever, and Hinton 2012) and PreResNet20 (He et al. 2016). The regularization parameter $\eta$ is fixed to 0.1 for both VIFO and VI, as this choice yields better performance compared with the standard choice $\eta = 1$ (see Appendix). SWA and SWAG are trained along with the base model using the code of Izmailov et al. (2018) and Maddox et al. (2019). Other models are optimized with the Adam optimizer. For each method we run 5 independent runs and report means and standard deviations in results. We use the average of the predictions from these runs as the prediction of the corresponding ensemble, so that we have one ensemble run per method. In addition, we performed experiments with regression and classification datasets from the UCI repository (Dua and Graff 2017), using a single-layer neural network with 50 hidden units. Due to space constraints the main paper reports detailed results for AlexNet and discusses other results as needed. Other experimental details and the full set of results is included in the Appendix. Note that results for PreResNet20 yield similar conclusions to the ones from AlexNet, so they are not discussed separately in the text.

### 7.1 Run Time

Ignoring the data preprocessing time, we compare the run time of training 1 epoch of VI, VIFO and 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.

### 7.2 Uncertainty Quantification

In this section we examine whether VIFO can capture the uncertainty in predictions for shifted and out-of-distribution (OOD) data. We measure performance using ECE, Entropy, AUC for detecting OOD data, and the appendix includes additional count confidence curves. 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 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, Cooper, and Hauskrecht 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.

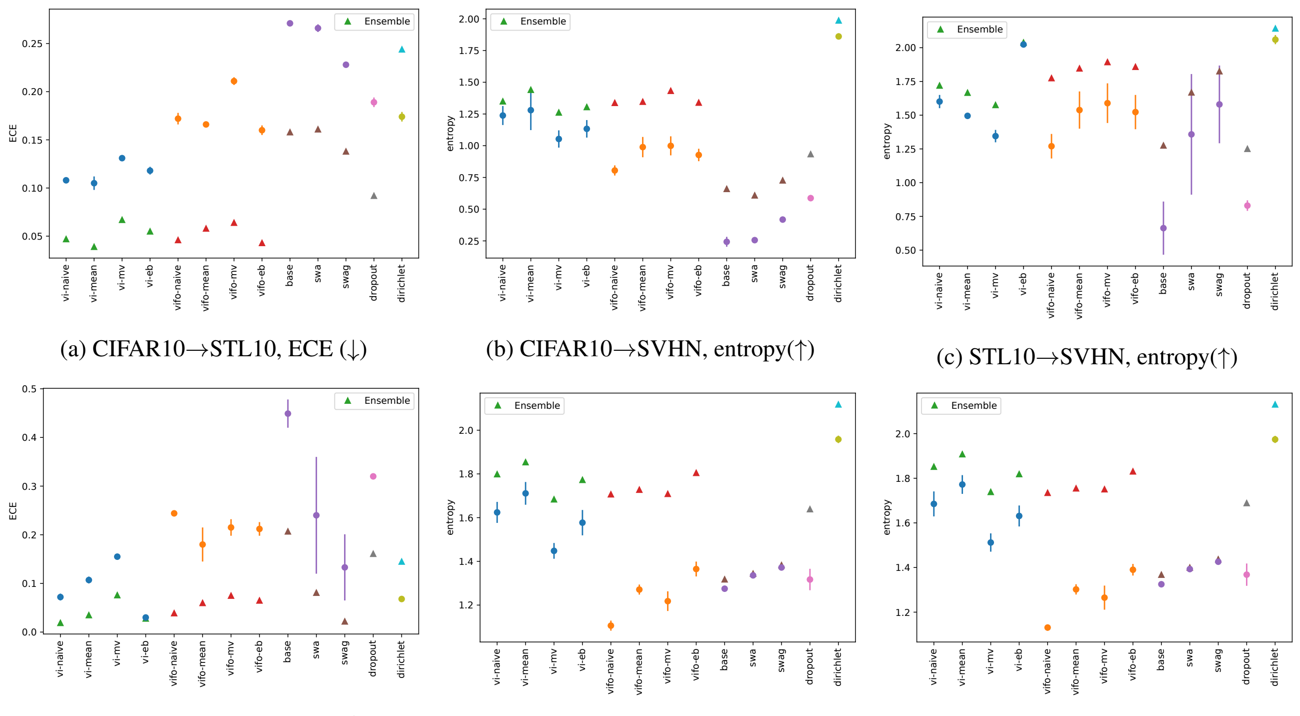
Figure 1a and 1d (and Table 7, 8 in Appendix) show the ECE of each method under data shift. Both single VI and ensembles of VI perform well; SWAG and Dropout are not stable; Ensembles of Dirichlet are worse than single Dirichlet model because Dirichlet models overestimate the uncertainty (see Figure 8 in Appendix); Single VIFO performs worse than single VI; Ensembles of VIFO outperform VI and are competitive with ensembles of VI.

**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 (maximum probability) in the predictive distribution. We summarize the averaged entropy for the entire dataset in Figure 1 and present count-confidence plots in Appendix. Here single-model VIFO outperforms the base model and Dropout. Ensembles of VIFO outperform VI and are competitive with ensembles of VI.

**AUROC:** We use maximum probability of the categorical predictive distribution as the criterion to separate in-

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

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



(a) CIFAR10→STL10, ECE (↓)

(b) CIFAR10→SVHN, entropy(↑)

(c) STL10→SVHN, entropy(↑)

(d) STL10→CIFAR10, ECE (↓)

(e) SVHN→CIFAR10, entropy(↑)

(f) SVHN→STL10, entropy(↑)

Figure 1: ECE and entropy on AlexNet. Each dot with error bar represents the mean and standard deviation of 5 independent runs and the triangle represents the ensemble of each method that aggregate these 5 runs for prediction (same for other figures). Numerical results are listed in Table 7, 9, 10 in the Appendix.

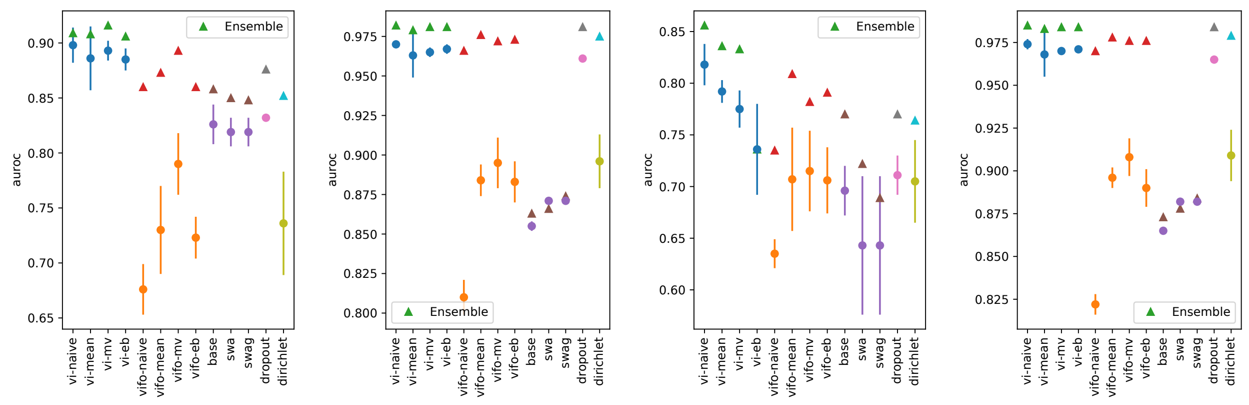
distribution and OOD data and compute the area under the ROC curve (Malinin and Gales 2018). AUROC overcomes the drawbacks of ECE and entropy because a trivial model cannot yield the best performance. Figure 2 shows the AUROC result on AlexNet. Single VIFO, especially vifo-mean, performs better than all baselines in most cases; Ensembles of VIFO are better than ensembles of baselines; they are competitive with single VI, and are close to ensemble of VI which has the best performance.

## 7.3 In-distribution Log Loss

In this section we use log loss to measure uncertainty quantification. In addition, for completeness, the Appendix includes standard performance criteria (accuracy for classification and RMSE for regression) results for all experiments.

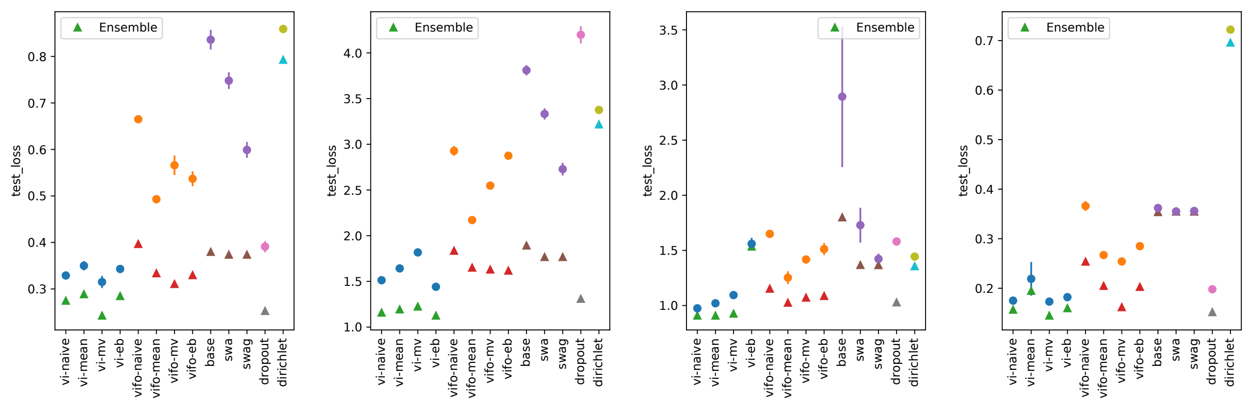
Results for small datasets are reported in the Appendix (Figures 4-7). For regression, vifo-mean and vifo-mv both perform well and they are better than VI and the hybrid methods in three out of four datasets. For classification, VIFO methods are comparable to VI and other baselines.

Figure 3 (and Tables 3, 5 in Appendix) compare all methods on large datasets in terms of log loss. First we observe that using collapsed variational inference in VI does not significantly improve the performance as shown by Tomczak et al. (2021). This is because we use $\eta = 0.1$ that yields better performance while Tomczak et al. (2021) use $\eta = 1$. Second, consider single models, we observe that vifo-mean is the best among all VIFO methods and it is better than the base model. VI performs better than single model VIFO. SWAG is slightly better in 2 cases but it is significantly worse in other cases. Dropout is not stable across different datasets. The Dirichlet-based model is significantly worse than other methods. Third, we observe that for all methods, ensembles perform better than single models. The ensemble of vifo-mean is competitive with VI. Thus, ensembles of VIFO can help alleviate the limitation of the expressiveness of VIFO. Ensembles of dropout also perform well. Last but not least, we observe that the ensemble of VI performs the



(a) CIFAR10→SVHN (b) SVHN→CIFAR10 (c) STL10→SVHN (d) SVHN→STL10

Figure 2: AUROC (↑) results on AlexNet. Exact numbers are listed in Appendix.



(a) CIFAR10 (b) CIFAR100 (c) STL10 (d) SVHN

Figure 3: Test log loss of image datasets on AlexNet. Numerical results are listed in Table 3 in the Appendix.

best for in distribution log loss (although optimizing each model in the ensemble is slower than other methods). Ensembles of VI have not been well studied and our experiments highlight their potential strong performance.

Overall, ensembles of VIFO are competitive with single VI and ensembles of dropout in terms of in-distribution log loss and are much better than Dirichlet models. On the other hand they provide better uncertainty quantification for out-of-distribution data, which is competitive with ensembles of VI. Among all VIFO variants, the methods using a hierarchical prior only for the mean (vifo-mean and ens-vifo-mean) provide the best overall performance.

## 8 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 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 the strong performance of ensembles of VI (when using weak regularization $\eta = 0.1$ instead of 1) albeit at the cost of high run time. Ensembles of VIFO are competitive with VI and other methods in terms of in-distribution loss, outperform VI for out of distribution data, and are competitive with ensembles of VI. 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.

## Acknowledgments

This work was partly supported by NSF under grants 1906694 and 2246261. Some of the experiments in this paper were run on the Big Red computing system at Indiana University, supported in part by Lilly Endowment, Inc., through its support for the Indiana University Pervasive Technology Institute.

## References

Bengs, V.; Hüllermeier, E.; and Waegeman, W. 2022. Pitfalls of Epistemic Uncertainty Quantification through Loss Minimisation. In Oh, A. H.; Agarwal, A.; Belgrave, D.; and Cho, K., eds., *Advances in Neural Information Processing Systems*.

Blundell, C.; Cornebise, J.; Kavukcuoglu, K.; and Wierstra, D. 2015. Weight Uncertainty in Neural Network. In Bach, F.; and Blei, D., eds., *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, 1613–1622. Lille, France: PMLR.

Brosse, N.; Riquelme, C.; Martin, A.; Gelly, S.; and Moulines, E. 2020. On Last-Layer Algorithms for Classification: Decoupling Representation from Uncertainty Estimation.

Charpentier, B.; Zügner, D.; and Günnemann, S. 2020. Posterior Network: Uncertainty Estimation without OOD Samples via Density-Based Pseudo-Counts. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 1356–1367. Curran Associates, Inc.

Dua, D.; and Graff, C. 2017. UCI Machine Learning Repository.

Gal, Y.; and Ghahramani, Z. 2016. 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, 1050–1059. JMLR.org.

Germain, P.; Bach, F.; Lacoste, A.; and Lacoste-Julien, S. 2016. PAC-Bayesian Theory Meets Bayesian Inference. In Lee, D.; Sugiyama, M.; Luxburg, U.; Guyon, I.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.

Golowich, N.; Rakhlin, A.; and Shamir, O. 2018. Size-Independent Sample Complexity of Neural Networks. In Bubeck, S.; Perchet, V.; and Rigollet, P., eds., *Proceedings of the 31st Conference On Learning Theory*, volume 75 of *Proceedings of Machine Learning Research*, 297–299. PMLR.

Graves, A. 2011. Practical Variational Inference for Neural Networks. In Shawe-Taylor, J.; Zemel, R.; Bartlett, P.; Pereira, F.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc.

Guo, C.; Pleiss, G.; Sun, Y.; and Weinberger, K. Q. 2017. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, 1321–1330. JMLR.org.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Identity Mappings in Deep Residual Networks. In Leibe, B.; Matas, J.; Sebe, N.; and Welling, M., eds., *Computer Vision – ECCV 2016*, 630–645. Cham: Springer International Publishing. ISBN 978-3-319-46493-0.

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

Izmailov, P.; Podoprikhin, D.; Garipov, T.; Vetrov, D. P.; and Wilson, A. G. 2018. Averaging Weights Leads to Wider Optima and Better Generalization. In Globerson, A.; and Silva, R., eds., *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*, 876–885. AUAI Press.

Izmailov, P.; Vikram, S.; Hoffman, M. D.; and Wilson, A. G. G. 2021. What Are Bayesian Neural Network Posteriors Really Like? In Meila, M.; and Zhang, T., eds., *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, 4629–4640. PMLR.

Jankowiak, M.; Pleiss, G.; and Gardner, J. R. 2020. Parametric Gaussian Process Regressors. In *ICML*.

Kabir, H. M. D.; Khosravi, A.; Hosen, M. A.; and Nahavandi, S. 2018. Neural Network-Based Uncertainty Quantification: A Survey of Methodologies and Applications. *IEEE Access*, 6: 36218–36234.

Kendall, A.; and Gal, Y. 2017. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Khosravi, A.; Nahavandi, S.; Creighton, D.; and Atiya, A. F. 2011. Comprehensive Review of Neural Network-Based Prediction Intervals and New Advances. *IEEE Transactions on Neural Networks*, 22(9): 1341–1356.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In Pereira, F.; Burges, C.; Bottou, L.; and Weinberger, K., eds., *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc.

Lakshminarayanan, B.; Pritzel, A.; and Blundell, C. 2017. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Li, Y.; Hernández-Lobato, J. M.; and Turner, R. E. 2015. Stochastic Expectation Propagation. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'15, 2323–2331. Cambridge, MA, USA: MIT Press.

Maddox, W. J.; Garipov, T.; Izmailov, P.; Vetrov, D.; and Wilson, A. G. 2019. *A Simple Baseline for Bayesian Un-*

*certainty in Deep Learning*. Red Hook, NY, USA: Curran Associates Inc.

Malinin, A.; and Gales, M. 2018. Predictive Uncertainty Estimation via Prior Networks. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

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

Naeini, M. P.; Cooper, G. F.; and Hauskrecht, M. 2015. Obtaining Well Calibrated Probabilities Using Bayesian Binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, 2901–2907. AAAI Press. ISBN 0262511290.

Oleksiienko, I.; Tran, D. T.; and Iosifidis, A. 2022. Variational Neural Networks.

Ovadia, Y.; Fertig, E.; Ren, J.; Nado, Z.; Sculley, D.; Nowozin, S.; Dillon, J. V.; Lakshminarayanan, B.; and Snoek, J. 2019. *Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty under Dataset Shift*. Red Hook, NY, USA: Curran Associates Inc.

Petersen, K. B.; and Pedersen, M. S. 2012. The Matrix Cookbook.

Sensoy, M.; Kaplan, L.; and Kandemir, M. 2018. Evidential Deep Learning to Quantify Classification Uncertainty. In Bengio, S.; Wallach, H.; Larochelle, H.; Grauman, K.; Cesa-Bianchi, N.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

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

Sheth, R.; and Khardon, R. 2017. Excess Risk Bounds for the Bayes Risk using Variational Inference in Latent Gaussian Models. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Sun, S.; Zhang, G.; Shi, J.; and Grosse, R. 2019. FUNCTIONAL VARIATIONAL BAYESIAN NEURAL NETWORKS. In *International Conference on Learning Representations*.

Teh, Y.; Hasenclever, L.; Lienart, T.; Vollmer, S.; Webb, S.; Lakshminarayanan, B.; and Blundell, C. 2015. Distributed Bayesian Learning with Stochastic Natural Gradient Expectation Propagation and the Posterior Server. *Journal of Machine Learning Research*, 18.

Tomczak, M.; Swaroop, S.; and Turner, R. 2020. Efficient Low Rank Gaussian Variational Inference for Neural Networks. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 4610–4622. Curran Associates, Inc.

Tomczak, M. B.; Swaroop, S.; Foong, A. Y. K.; and Turner, R. E. 2021. Collapsed Variational Bounds for Bayesian Neural Networks. In Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*.

Tran, B.-H.; Rossi, S.; Milios, D.; and Filippone, M. 2022. All You Need is a Good Functional Prior for Bayesian Deep Learning. *Journal of Machine Learning Research*, 23: 1–56.

Wenzel, F.; Roth, K.; Veeling, B. S.; Świątkowski, J.; Tran, L.; Mandt, S.; Snoek, J.; Salimans, T.; Jenatton, R.; and Nowozin, S. 2020. 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.

Wilson, A. G.; and Izmailov, P. 2020. Bayesian Deep Learning and a Probabilistic Perspective of Generalization. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 4697–4708. Curran Associates, Inc.

Wilson, A. G.; Izmailov, P.; Hoffman, M. D.; Gal, Y.; Li, Y.; Pradier, M. F.; Vikram, S.; Foong, A.; Lotfi, S.; and Farquhar, S. 2022. Evaluating Approximate Inference in Bayesian Deep Learning. In Kiela, D.; Ciccone, M.; and Caputo, B., eds., *Proceedings of the NeurIPS 2021 Competitions and Demonstrations Track*, volume 176 of *Proceedings of Machine Learning Research*, 113–124. PMLR.

Wu, A.; Nowozin, S.; Meeds, E.; Turner, R. E.; Hernandez-Lobato, J. M.; and Gaunt, A. L. 2019. Deterministic Variational Inference for Robust Bayesian Neural Networks. In *International Conference on Learning Representations*.

# A Proofs

## A.1 Proofs in Section 3

The next two lemmas extend known bounds for Rademacher complexity through Lipschitz bounds (Shalev-Shwartz and Ben-David 2014) to functions with multi-dimensional inputs.

**Lemma A.1.** *Consider an $L$-Lipschitz function $\phi : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, i.e. $\phi(a_1, b_1) - \phi(a_2, b_2) \le L(|a_1 - a_2| + |b_1 - b_2|)$. For $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^N$, let $\phi(\boldsymbol{a}, \boldsymbol{b})$ denote the vector $(\phi(a_1, b_1), \dots, \phi(a_N, b_N))$. Let $\phi(A \times B)$ denote $\{\phi(\boldsymbol{a}, \boldsymbol{b}) : \boldsymbol{a} \in A, \boldsymbol{b} \in B\}$, then*

$$R(\phi(A \times B)) \le L(R(A) + R(B)). \tag{10}$$

*Proof.* 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 $R(\phi(A \times B)) \le LR(\phi'(A \times B))$. Let $C_i = \{(a_1 + b_1, \dots, a_{i-1} + b_{i-1}, \phi'(a_i, b_i), a_{i+1} + b_{i+1}, \dots, a_N + b_N) : a \in A, b \in B\}$. It suffices to prove that for any set $A, B$ and all $i$ we have $R(C_i) \le R(A) + R(B)$. Without loss of generality we prove the case for $i = 1$.

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

□

Applying the previous lemma sequentially over multiple dimensions we can obtain:

**Corollary A.2.** *Consider an $L$-Lipschitz function $\phi : \mathbb{R}^d \to \mathbb{R}$, i.e., for any $x, x' \in \mathbb{R}^d$, $\phi(x) - \phi(x') \le L\|x - x'\|_1$. Let $\phi(A^d) = \{\phi(a_{1:d,i}) : \boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_d \in A \subset \mathbb{R}^N\}$, then $R(\phi(A^d)) \le Ld(R(A))$.*

We next verify that Assumption 3.1 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 $l$.

For Assumption 3.2, 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 4

*Proof of Theorem 4.1.* 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:

$$\sum_{i=1}^{N} \left\{ \mathbb{E}_{q(z|x_i)}[\log p(y_i|z)] \right\} - \mathrm{KL}(q(z|X_N)||p(z|X_N)) \tag{11}$$

$$= \sum_{i=1}^{N} \left\{ \mathbb{E}_{q(z|x_i)}[\log p(y_i|z)] \right\} - \frac{1}{2}\mathrm{tr}((X_N^\top S_0 X_N)^{-1}(X_N^\top V X_N)) + \frac{N}{2}$$
$$+ \frac{1}{2}\log|(X_N^\top S_0 X_N)^{-1}(X_N^\top V X_N)| - \frac{1}{2}(w^\top X_N - m_0^\top X_N)(X_N^\top S_0 X_N)^{-1}(w^\top X_N - m_0^\top X_N)^\top. \tag{12}$$

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

$$\begin{aligned}\mathbb{E}_{q(z|x_i)}[\log p(y_i|z)] &= \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I_d)}[\log p(y_i|w^\top x_i + x_i^\top L\epsilon)] \\ &= \mathbb{E}_{\epsilon\sim\mathcal{N}(0,I_d)}[\log p(y_i|(w + L\epsilon)^\top x_i)] \\ &= \mathbb{E}_{\theta\sim\mathcal{N}(w,LL^\top)}[\log p(y_i|\theta^\top x_i)],\end{aligned} \tag{13}$$

where the last equality uses reparametrization in a reverse order. By aligning $w = m$ and $V = LL^\top = S$, we recognize that Eq (13) is exactly the loss term in Eq (7). 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 and 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

$$(X_N^\ddagger S_0^{-1} V X_N) X_N^\ddagger u_i = X_N^\ddagger S_0^{-1} V u_i = \lambda X_N^\ddagger u_i, \tag{14}$$

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 (7) can be simplified to:

$$-\mathrm{KL}(q(z|X_N)||p(z|X_N)) = -\frac{1}{2}\mathrm{tr}(S_0^{-1}V) + \frac{1}{2}\log|S_0^{-1}V| - \frac{1}{2}(w - m_0)^\top S_0^{-1}(w - m_0) + \frac{N}{2}. \tag{15}$$

By aligning $w = m, V = S$, we can seet that eq (15) is exactly the regularizer in eq (7) ignoring the constant. $\square$

**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 $\theta_k$ is the $k$-th column of $\theta$. 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

$$\mathrm{KL}(q(\theta)||p(\theta)) = \sum_k \mathrm{KL}(q(\theta_k), p(\theta_k)) \doteq \sum_k \mathrm{KL}(q(z_k|X_N)||p(z_k|X_N)), \tag{16}$$

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

*Proof of Theorem 4.2.* 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 $x_1 \geq 0$,

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

if $x_1 < 0$, then

$$\begin{aligned}\mathbb{E}_{q(w)q(u)}\left[z^{(k)}\right] &= \mathbb{E}_{w,u}\left[w_{k,1}\psi(u_{1,1}x_1)\right] \\ &= \bar{w}_{k,1}\left(\bar{u}_{1,1} - \frac{\phi\left(-\frac{\bar{u}_{1,1}}{\sigma_u}\right)}{\Phi\left(-\frac{\bar{u}_{1,1}}{\sigma_u}\right)}\sigma_u\right)\Phi\left(-\frac{\bar{u}_{1,1}}{\sigma_u}\right)x_1,\end{aligned} \tag{18}$$

where $\phi$ and $\Phi$ 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 (17) and (18). If $\tilde{u}_{1,1} \geq 0$, it cannot successfully recover (18) because the ReLU activation will have 0 when $x_1 < 0$ so that it cannot recover (18); if $\tilde{u}_{1,1} < 0$, for the same reason it cannot recover (17). ☐

## B Derivations of Collapsed Variational Inference

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

$$\log q^*(\mu_p, \sigma_p^2|x) \propto \log p(\mu_p, \sigma_p^2) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_p, \sigma_p^2)], \tag{19}$$

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 $\mu_p$ and $\sigma_p$ 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

$$(\alpha + \frac{1}{2})1^\top \log\left[\beta 1 + \frac{t}{2(1+t)}\mu_q(x)^2 + \frac{1}{2}\sigma_q^2(x)\right] - \frac{1}{2}1^\top \log \sigma_q^2(x). \tag{20}$$

## 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 $\sigma_p^2$ 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:

$$\begin{aligned}
&\frac{1}{2}\left[K\log\frac{\mu_q(x)^\top \mu_q(x) + 1^\top \sigma_q^2(x) + 2\beta}{K + 2\alpha + 2} - 1^\top \log|\sigma_q^2(x)|\right] \\
&- \frac{K}{2} + \frac{1}{2}\frac{(K + 2\alpha + 2)(\mu_q(x)^\top \mu_q(x) + 1^\top \sigma_q^2(x))}{\mu_q(x)^\top \mu_q(x) + 1^\top \sigma_q^2(x) + 2\beta}.
\end{aligned} \tag{21}$$

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 eq (21).

## 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

$$\log q^*(\mu_p, \sigma_p^2) \propto \frac{1}{N}\sum_{(x,y)\in\mathcal{D}}\left\{\log p(\mu_p, \sigma_p^2) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_p, \sigma_p^2)]\right\}. \tag{22}$$

## 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 eq (22),

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

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:

$$\begin{aligned}&\sum_{(x,y)} \left\{ \mathbb{E}_{q(\mu_p)}[\mathrm{KL}(q(z|x)||p(z|\mu_p,\gamma))] + \mathrm{KL}(q(\mu_p)||p(\mu_p)) \right\} \\ &= \sum_{(x,y)} \left\{ \mathbb{E}_{q(\mu_p)} \left[ K\log\gamma - 1^\top \log\sigma_q^2(x) - K + \frac{1}{\gamma}1^\top\sigma_q^2(x) - \frac{1}{\gamma}(\mu_q(x)-\mu_p)^\top(\mu_q(x)-\mu_p) \right] \right\} \\ &\quad + \frac{N}{2}\left[ K\log\frac{\alpha+\gamma}{\gamma} - K + K\frac{\gamma}{\gamma+\alpha} + \frac{\alpha^2}{(\alpha+\gamma)^2}\bar{\mu}_q^\top\bar{\mu}_q \right] \\ &= \sum_{(x,y)} \left\{ \frac{1}{2\gamma}(1^\top\sigma_q^2(x) + \mu_q(x)^\top\mu_q(x)) - \frac{1}{2}1^\top\log\sigma_q^2(x) \right\} - \frac{N}{2}\left(\frac{1}{\gamma} - \frac{1}{\alpha+\gamma}\right)\bar{\mu}_q^\top\bar{\mu}_q + \frac{NK}{2}\log(\alpha+\gamma) - \frac{NK}{2}. \qquad (23)\end{aligned}$$

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

$$\begin{aligned}&\log p(\mu_{p,i}, \sigma_{p,i}^2) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_{p,i}, \sigma_{p,i}^2)] && (24)\\ =&\log p(\mu_{p,i}|\sigma_{p,i}^2) + \log p(\sigma_{p,i}^2) + \mathbb{E}_{q(z|x)}[\log p(z|\mu_{p,i}, \sigma_{p,i}^2)] && (25)\\ \propto& -\frac{t}{2}\frac{\mu_{p,i}^2}{\sigma_{p,i}^2} - \frac{1}{2}\log\sigma_{p,i}^2 - (\alpha+1)\log\sigma_{p,i}^2 - \frac{\beta}{\sigma_{p,i}^2} - \frac{1}{2}\log\sigma_{p,i}^2 - \frac{1}{2\sigma_{p,i}^2}((\mu_{p,i} - \mu_{q,i}(x))^2 + \sigma_{q,i}^2(x)) && (26)\\ =& -\frac{t}{2}\frac{\mu_{p,i}^2}{\sigma_{p,i}^2} - \log\sigma_{p,i}^2 - (\alpha+1)\log\sigma_{p,i}^2 - \frac{\beta}{\sigma_{p,i}^2} - \frac{1}{2\sigma_{p,i}^2}((\mu_{p,i} - \mu_{q,i}(x))^2 + \sigma_{q,i}^2(x)), && (27)\\ =& -\frac{t+1}{2\sigma_{p,i}^2}\left(\mu_{p,i} - \frac{1}{t+1}\mu_{q,i}(x)\right)^2 - \frac{t\mu_{q,i}^2(x)}{2(t+1)\sigma_{p,i}^2} - (\alpha+2)\log\sigma_{p,i}^2 - \frac{\beta}{\sigma_{p,i}^2} - \frac{\sigma_{q,i}^2(x)}{2\sigma_{p,i}^2} && (28)\end{aligned}$$

Then by extracting the $\mu_p$ part from eq (28), 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 $\mu_p$ 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:

$$(\alpha+\frac{1}{2})N1^\top\log\left[\beta 1 + \frac{t}{2(1+t)}\tilde{\mu}_q^2 + \frac{1}{2}\tilde{\sigma}_q^2\right] - \sum_{(x,y)}\frac{1}{2}1^\top\log\sigma_q^2(x) \tag{29}$$

$$+ KN\log\frac{\Gamma(\alpha)}{\Gamma(\alpha+\frac{1}{2})} - NK\alpha\log\beta + \frac{NK}{2}\log\frac{t+1}{t} - \frac{NK}{2}. \tag{30}$$

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

## C.3 Empirical Bayes for all data

If we optimize $\sigma_p^2$ 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

In this section we elaborate the experimental details, including the choice of hyperparameters, learning rates and the number of training epochs. Since all experiments use the same sets of hyperparameters for collapsed variational inference, we introduce these here. 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 $\alpha$ in empirical Bayes follows (Wu et al. 2019) but the choice of $\beta$ 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.

Next we list the choices of the variance for naive methods and regularization parameters. The choice of prior variance significantly affects the performance. For UCI classification and regression tasks, we set the prior variance to be 0.1 for both vi-naive and vifo-naive. 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 in (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.

Next we list the choices for $\eta$ and optimization routine. For UCI classification and regression tasks, the regularization parameter $\eta \in \{0.1, 0.3, 0.5, 0.75, 1.0, 3.0, 5.0, 7.5, 10.0\}$ is selected based on validation log loss. The regularization parameter $\eta$ is fixed 0.1 for large datasets. For all VI and VIFO methods, we use the Adam optimizer with learning rate 0.001, except that we choose learning rate 0.0005 for boston and concrete which are hard to optimize.

We next list the details for running the hybrid methods, including the base model, SWA and SWAG. The hybrid methods 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 2. 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 2. For the weight decay on small data experiments, we select it from the same range of $\eta$ based on validation log loss.

| | lr | wd | swag_lr | swag_start | epochs |
|---|---|---|---|---|---|
| UCI Classification | 0.001 | - | 0.001 | 1001 | 2000 |
| boston | 0.0005 | - | 0.0005 | 1001 | 5000 |
| concrete | 0.0005 | - | 0.0005 | 1001 | 5000 |
| yacht | 0.001 | - | 0.001 | 1001 | 5000 |
| power | 0.001 | - | 0.001 | 1001 | 5000 |
| 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 |

Table 2: The parameters for running the hybrid algorithms. "lr" means the learning rate, "wd" means weight decay, "swag_lr" means the learning rate after we start collecting models in SWA and SWAG algorithms, "swag_start" means the epochs when we start to collect models, "epochs" is the number of training epochs.

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. As discussed by (Bengs, Hüllermeier, and Waegeman 2022), the models of Sensoy, Kaplan, and Kandemir (2018); Charpentier, Zügner, and Günnemann (2020) implicitly perform variational inference:

$$\mathbf{p} \sim \text{Dir}(\boldsymbol{\alpha}_0), \quad y|\mathbf{p} \sim \text{Cat}(\mathbf{p}), \tag{31}$$

and the ELBO becomes

$$\log p(y|x) \geq \mathbb{E}_{q(\mathbf{p}|x)}[\log p(y|\mathbf{p})] - \text{KL}(q(\mathbf{p}|x)||\text{Dir}(\mathbf{p}|\boldsymbol{\alpha}_0)), \tag{32}$$

where $q(\mathbf{p}|x) = \text{Dir}(\mathbf{p}|\boldsymbol{\alpha}(x))$. In the experiments, following Sensoy, Kaplan, and Kandemir (2018); Bengs, Hüllermeier, and Waegeman (2022), we use a uniform prior with $\boldsymbol{\alpha}_0 = [1, \ldots, 1]$ and optimize dirichlet models using Adam optimizer with learning rate 0.001.

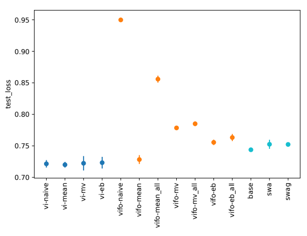
## E Additional Experimental Results

In this section we list the results that are not in main paper due to space limit. We also include the comparisons of all methods discussed, including those in Section C.

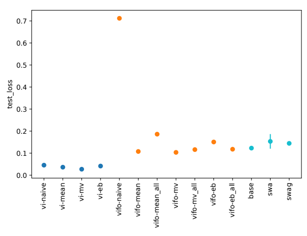
### E.1 Log Loss and Accuracy for In-distribution Data

Figure 4, 5, 6, 7 show the results of all methods performing on small datasets. Table 3, 4, 5, 6 show the log loss and accuracy of in-distribution data in image datasets on different neural network structures. In terms of both log loss and accuracy, we observe that

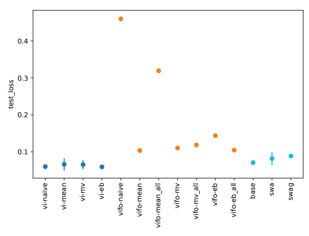
- The Dirichlet model is the worst in terms of log loss and does not perform well in terms of accuracy;
- vifo-mean is the best among all VIFO methods and it is better than baselines in most cases. However, it is outperformed by VI.
- Ensemble of vifo-mean is competitive with VI, and often competitive with ensemble of VI.



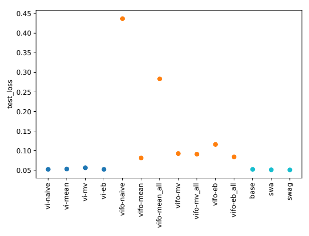
(a) abalone



(b) thyroid

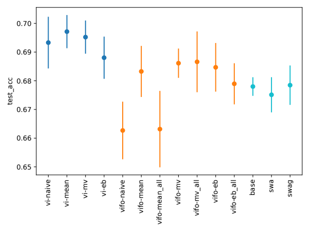


(c) ringnorm

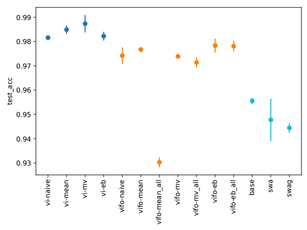


(d) twonorm

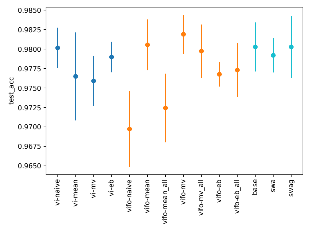
Figure 4: Test log loss on UCI classification datasets. Each dot with the error bar shows the mean and standard deviation of 5 independent runs (same for Figure 6). The standard deviations for some methods are very small.



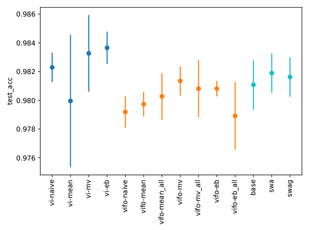
(a) abalone



(b) thyroid



(c) ringnorm

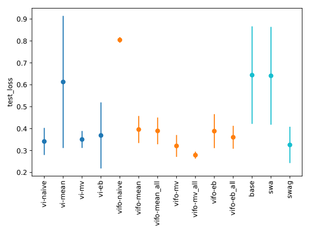


(d) twonorm

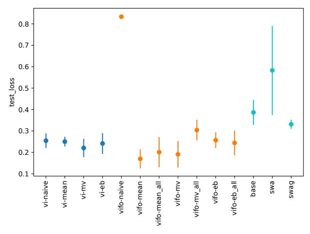
Figure 5: Test accuracy on UCI classification datasets.

Table 3: Test loss on AlexNet. Each cell contains the mean and standard deviation of the results running with 5 random seeds and the separate number is the result of the ensemble of corresponding models. The same representation is used in the remaining tables.

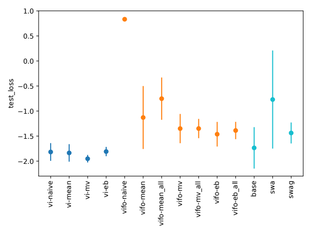
| **method** | **CIFAR10** | **CIFAR100** | **STL10** | **SVHN** |
|---|---|---|---|---|
| vi-naive | $0.329 \pm 0.006$, 0.275 | $1.513 \pm 0.024$, 1.160 | $0.975 \pm 0.010$, 0.910 | $0.175 \pm 0.002$, 0.157 |
| vi-mean | $0.350 \pm 0.010$, 0.289 | $1.642 \pm 0.023$, 1.195 | $1.021 \pm 0.013$, 0.910 | $0.219 \pm 0.034$, 0.195 |
| vi-mv | $0.315 \pm 0.013$, 0.243 | $1.817 \pm 0.022$, 1.226 | $1.095 \pm 0.018$, 0.928 | $0.173 \pm 0.003$, 0.145 |
| vi-eb | $0.343 \pm 0.004$, 0.285 | $1.441 \pm 0.016$, 1.127 | $1.560 \pm 0.054$, 1.536 | $0.182 \pm 0.004$, 0.160 |
| vifo-naive | $0.665 \pm 0.008$, 0.397 | $2.928 \pm 0.053$, 1.837 | $1.650 \pm 0.034$, 1.154 | $0.366 \pm 0.010$, 0.254 |
| vifo-mean | $0.493 \pm 0.005$, 0.334 | $2.171 \pm 0.012$, 1.652 | $1.253 \pm 0.058$, 1.027 | $0.267 \pm 0.007$, 0.205 |
| vifo-mv | $0.566 \pm 0.021$, 0.311 | $2.548 \pm 0.019$, 1.631 | $1.418 \pm 0.037$, 1.073 | $0.254 \pm 0.008$, 0.162 |
| vifo-eb | $0.537 \pm 0.016$, 0.330 | $2.875 \pm 0.032$, 1.619 | $1.512 \pm 0.055$, 1.088 | $0.285 \pm 0.007$, 0.203 |
| base | $0.836 \pm 0.021$, 0.380 | $3.810 \pm 0.056$, 1.894 | $2.894 \pm 0.639$, 1.801 | $0.362 \pm 0.005$, 0.354 |
| swa | $0.748 \pm 0.018$, 0.374 | $3.332 \pm 0.061$, 1.768 | $1.729 \pm 0.158$, 1.369 | $0.355 \pm 0.000$, 0.355 |
| swag | $0.599 \pm 0.017$, 0.374 | $2.728 \pm 0.069$, 1.768 | $1.423 \pm 0.048$, 1.368 | $0.356 \pm 0.000$, 0.355 |
| dropout | $0.391 \pm 0.012$, 0.253 | $4.198 \pm 0.094$, 1.311 | $1.581 \pm 0.037$, 1.030 | $0.198 \pm 0.003$, 0.152 |
| dirichlet | $0.859 \pm 0.006$, 0.793 | $3.376 \pm 0.017$, 3.219 | $1.444 \pm 0.006$, 1.358 | $0.722 \pm 0.007$, 0.696 |



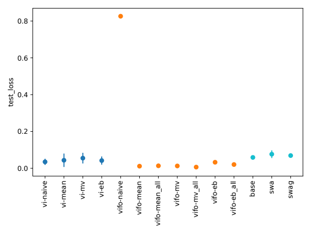
(a) boston, test loss



(b) concrete, test loss

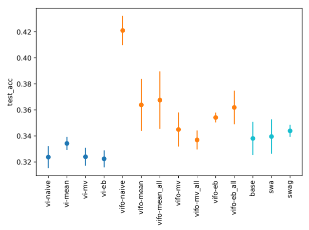


(c) yacht, test loss

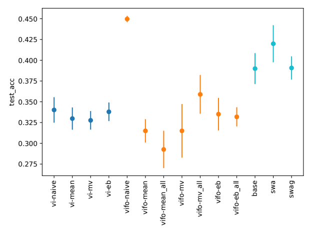


(d) power, test loss

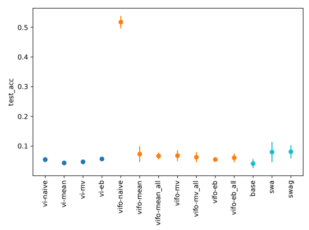
Figure 6: Test log loss on UCI regression datasets.



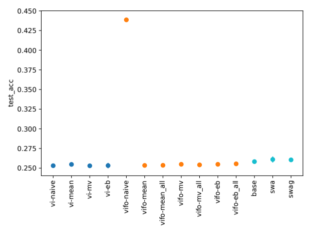
(a) boston



(b) concrete



(c) yacht



(d) power

Figure 7: Test RMSE on UCI regression datasets. In some plots vifo-naive is missing because its performance is too bad.

Table 4: Test accuracy on AlexNet

| **method** | **CIFAR10** | **CIFAR100** | **STL10** | **SVHN** |
|---|---|---|---|---|
| vi-naive | $0.893 \pm 0.004$, 0.916 | $0.620 \pm 0.003$, 0.686 | $0.661 \pm 0.003$, 0.685 | $0.953 \pm 0.002$, 0.964 |
| vi-mean | $0.884 \pm 0.004$, 0.912 | $0.608 \pm 0.002$, 0.681 | $0.649 \pm 0.009$, 0.688 | $0.945 \pm 0.008$, 0.961 |
| vi-mv | $0.901 \pm 0.003$, 0.921 | $0.606 \pm 0.002$, 0.681 | $0.644 \pm 0.006$, 0.682 | $0.955 \pm 0.001$, 0.965 |
| vi-eb | $0.886 \pm 0.003$, 0.909 | $0.629 \pm 0.004$, 0.693 | $0.414 \pm 0.025$, 0.434 | $0.950 \pm 0.001$, 0.962 |
| vifo-naive | $0.856 \pm 0.003$, 0.908 | $0.497 \pm 0.010$, 0.650 | $0.562 \pm 0.009$, 0.660 | $0.926 \pm 0.003$, 0.962 |
| vifo-mean | $0.882 \pm 0.001$, 0.917 | $0.587 \pm 0.002$, 0.664 | $0.629 \pm 0.011$, 0.674 | $0.947 \pm 0.002$, 0.964 |
| vifo-mv | $0.876 \pm 0.005$, 0.917 | $0.569 \pm 0.004$, 0.668 | $0.630 \pm 0.006$, 0.684 | $0.946 \pm 0.002$, 0.963 |
| vifo-eb | $0.878 \pm 0.004$, 0.915 | $0.577 \pm 0.001$, 0.672 | $0.611 \pm 0.009$, 0.664 | $0.943 \pm 0.001$, 0.962 |
| base | $0.881 \pm 0.002$, 0.904 | $0.577 \pm 0.002$, 0.651 | $0.540 \pm 0.015$, 0.605 | $0.895 \pm 0.002$, 0.898 |
| swa | $0.886 \pm 0.001$, 0.908 | $0.586 \pm 0.004$, 0.652 | $0.485 \pm 0.064$, 0.602 | $0.898 \pm 0.000$, 0.899 |
| swag | $0.885 \pm 0.001$, 0.908 | $0.589 \pm 0.005$, 0.652 | $0.539 \pm 0.028$, 0.602 | $0.899 \pm 0.000$, 0.900 |
| dropout | $0.892 \pm 0.003$, 0.920 | $0.522 \pm 0.006$, 0.660 | $0.604 \pm 0.007$, 0.681 | $0.952 \pm 0.000$, 0.965 |
| dirichlet | $0.876 \pm 0.002$, 0.914 | $0.557 \pm 0.007$, 0.655 | $0.626 \pm 0.004$, 0.665 | $0.946 \pm 0.001$, 0.962 |

Table 5: Test loss on PreResNet20

| **method** | **CIFAR10** | **CIFAR100** | **STL10** | **SVHN** |
|---|---|---|---|---|
| vi-naive | $0.410 \pm 0.028$, 0.333 | $1.642 \pm 0.030$, 1.236 | $0.920 \pm 0.032$, 0.786 | $0.314 \pm 0.024$, 0.270 |
| vi-mean | $0.415 \pm 0.032$, 0.313 | $1.753 \pm 0.086$, 1.245 | $1.000 \pm 0.029$, 0.813 | $0.342 \pm 0.040$, 0.283 |
| vi-mv | $0.437 \pm 0.029$, 0.314 | $1.804 \pm 0.089$, 1.233 | $1.002 \pm 0.036$, 0.791 | $0.359 \pm 0.033$, 0.280 |
| vi-eb | $0.429 \pm 0.035$, 0.335 | $1.731 \pm 0.097$, 1.289 | $1.018 \pm 0.028$, 0.919 | $0.379 \pm 0.057$, 0.307 |
| vifo-naive | $0.783 \pm 0.067$, 0.471 | $2.782 \pm 0.040$, 2.002 | $1.886 \pm 0.067$, 1.127 | $0.652 \pm 0.042$, 0.427 |
| vifo-mean | $0.582 \pm 0.033$, 0.404 | $2.085 \pm 0.068$, 1.692 | $1.405 \pm 0.015$, 1.016 | $0.426 \pm 0.030$, 0.313 |
| vifo-mv | $0.637 \pm 0.026$, 0.353 | $3.287 \pm 0.027$, 2.731 | $1.866 \pm 0.033$, 1.045 | $0.481 \pm 0.024$, 0.307 |
| vifo-eb | $0.612 \pm 0.029$, 0.386 | $2.902 \pm 0.114$, 2.110 | $1.749 \pm 0.047$, 1.045 | $0.467 \pm 0.044$, 0.328 |
| base | $0.751 \pm 0.024$, 0.331 | $2.513 \pm 0.105$, 1.445 | $2.544 \pm 0.237$, 1.209 | $0.639 \pm 0.109$, 0.338 |
| swa | $0.673 \pm 0.015$, 0.320 | $2.181 \pm 0.097$, 1.355 | $1.873 \pm 0.317$, 1.100 | $0.537 \pm 0.117$, 0.295 |
| swag | $0.514 \pm 0.010$, 0.320 | $1.848 \pm 0.074$, 1.354 | $1.470 \pm 0.270$, 1.100 | $0.421 \pm 0.091$, 0.294 |
| dropout | $0.416 \pm 0.021$, 0.330 | $2.286 \pm 0.096$, 1.431 | $1.732 \pm 0.051$, 0.829 | $0.708 \pm 0.099$, 0.337 |
| dirichlet | $1.107 \pm 0.206$, 0.968 | $3.848 \pm 0.045$, 3.590 | $1.576 \pm 0.125$, 1.392 | $0.886 \pm 0.058$, 0.817 |

Table 6: Test accuracy on PreResNet20

| **method** | **CIFAR10** | **CIFAR100** | **STL10** | **SVHN** |
|---|---|---|---|---|
| vi-naive | $0.862 \pm 0.009, 0.898$ | $0.578 \pm 0.004, 0.673$ | $0.690 \pm 0.009, 0.733$ | $0.914 \pm 0.009, 0.941$ |
| vi-mean | $0.867 \pm 0.009, 0.901$ | $0.572 \pm 0.016, 0.672$ | $0.674 \pm 0.010, 0.719$ | $0.901 \pm 0.015, 0.935$ |
| vi-mv | $0.864 \pm 0.010, 0.904$ | $0.574 \pm 0.017, 0.681$ | $0.688 \pm 0.009, 0.731$ | $0.902 \pm 0.012, 0.940$ |
| vi-eb | $0.859 \pm 0.011, 0.903$ | $0.560 \pm 0.016, 0.659$ | $0.638 \pm 0.011, 0.686$ | $0.890 \pm 0.019, 0.938$ |
| vifo-naive | $0.810 \pm 0.020, 0.893$ | $0.423 \pm 0.005, 0.583$ | $0.543 \pm 0.014, 0.649$ | $0.838 \pm 0.016, 0.915$ |
| vifo-mean | $0.848 \pm 0.010, 0.906$ | $0.535 \pm 0.011, 0.647$ | $0.597 \pm 0.004, 0.676$ | $0.896 \pm 0.010, 0.939$ |
| vifo-mv | $0.852 \pm 0.007, 0.903$ | $0.349 \pm 0.011, 0.435$ | $0.591 \pm 0.008, 0.668$ | $0.882 \pm 0.009, 0.929$ |
| vifo-eb | $0.851 \pm 0.010, 0.906$ | $0.469 \pm 0.019, 0.572$ | $0.590 \pm 0.014, 0.674$ | $0.888 \pm 0.014, 0.933$ |
| base | $0.854 \pm 0.006, 0.903$ | $0.517 \pm 0.016, 0.638$ | $0.595 \pm 0.012, 0.674$ | $0.849 \pm 0.030, 0.928$ |
| swa | $0.861 \pm 0.004, 0.904$ | $0.542 \pm 0.015, 0.650$ | $0.604 \pm 0.008, 0.679$ | $0.874 \pm 0.033, 0.937$ |
| swag | $0.863 \pm 0.005, 0.904$ | $0.552 \pm 0.015, 0.650$ | $0.610 \pm 0.008, 0.679$ | $0.884 \pm 0.029, 0.937$ |
| dropout | $0.863 \pm 0.008, 0.906$ | $0.516 \pm 0.013, 0.635$ | $0.644 \pm 0.009, 0.720$ | $0.845 \pm 0.024, 0.928$ |
| dirichlet | $0.758 \pm 0.123, 0.897$ | $0.376 \pm 0.016, 0.610$ | $0.548 \pm 0.077, 0.671$ | $0.884 \pm 0.021, 0.940$ |

## E.2 ECE for Data Shift

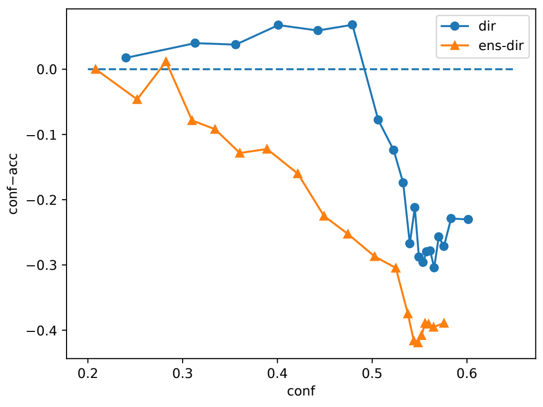
Table 7 and 8 list the expected calibration error (ECE) of all methods on AlexNet and PreResNet20. Notice that for Dirichlet models, the ECE of a single model is smaller than the ECE of the ensemble of the models. This is because Dirichlet models overestimate the uncertainty and ensembles of Dirichlet models further increase the uncertainty estimation and make the confidence further smaller than the accuracy. See Figure 8.

Table 7: ECE for data shift on AlexNet

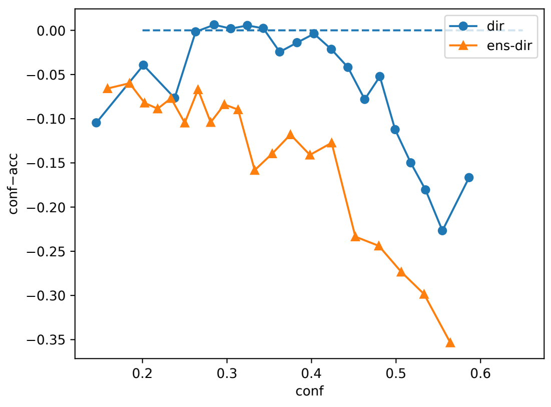
| **method** | **CIFAR10→STL10** | **STL10→CIFAR10** |
|---|---|---|
| vi-naive | $0.108 \pm 0.002$, 0.047 | $0.072 \pm 0.007$, 0.019 |
| vi-mean | $0.105 \pm 0.007$, 0.039 | $0.107 \pm 0.007$, 0.035 |
| vi-mv | $0.131 \pm 0.003$, 0.067 | $0.155 \pm 0.004$, 0.076 |
| vi-eb | $0.118 \pm 0.004$, 0.055 | $0.030 \pm 0.006$, 0.028 |
| vifo-naive | $0.172 \pm 0.006$, 0.046 | $0.244 \pm 0.004$, 0.039 |
| vifo-mean | $0.166 \pm 0.002$, 0.058 | $0.180 \pm 0.035$, 0.060 |
| vifo-mv | $0.211 \pm 0.004$, 0.064 | $0.215 \pm 0.017$, 0.075 |
| vifo-eb | $0.160 \pm 0.005$, 0.043 | $0.212 \pm 0.014$, 0.065 |
| base | $0.271 \pm 0.003$, 0.158 | $0.449 \pm 0.029$, 0.207 |
| swa | $0.266 \pm 0.004$, 0.161 | $0.240 \pm 0.120$, 0.081 |
| swag | $0.228 \pm 0.002$, 0.138 | $0.133 \pm 0.068$, 0.022 |
| dropout | $0.189 \pm 0.005$, 0.092 | $0.320 \pm 0.005$, 0.161 |
| dirichlet | $0.174 \pm 0.005$, 0.244 | $0.068 \pm 0.005$, 0.145 |

Table 8: ECE for data shift on PreResNet20

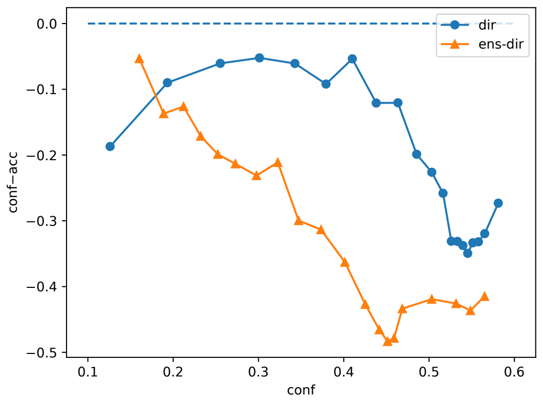
| **method** | **CIFAR10→STL10** | **STL10→CIFAR10** |
|---|---|---|
| vi-naive | $0.126 \pm 0.004$, 0.051 | $0.129 \pm 0.003$, 0.038 |
| vi-mean | $0.147 \pm 0.002$, 0.066 | $0.151 \pm 0.004$, 0.049 |
| vi-mv | $0.159 \pm 0.003$, 0.069 | $0.171 \pm 0.011$, 0.062 |
| vi-eb | $0.137 \pm 0.005$, 0.051 | $0.094 \pm 0.005$, 0.037 |
| vifo-naive | $0.169 \pm 0.007$, 0.031 | $0.293 \pm 0.005$, 0.017 |
| vifo-mean | $0.154 \pm 0.006$, 0.031 | $0.232 \pm 0.005$, 0.019 |
| vifo-mv | $0.220 \pm 0.004$, 0.053 | $0.362 \pm 0.006$, 0.094 |
| vifo-eb | $0.149 \pm 0.006$, 0.025 | $0.267 \pm 0.006$, 0.035 |
| base | $0.265 \pm 0.004$, 0.098 | $0.413 \pm 0.009$, 0.126 |
| swa | $0.258 \pm 0.004$, 0.107 | $0.359 \pm 0.038$, 0.120 |
| swag | $0.161 \pm 0.020$, 0.047 | $0.172 \pm 0.027$, 0.031 |
| dropout | $0.112 \pm 0.008$, 0.039 | $0.227 \pm 0.009$, 0.069 |
| dirichlet | $0.049 \pm 0.009$, 0.206 | $0.177 \pm 0.024$, 0.314 |



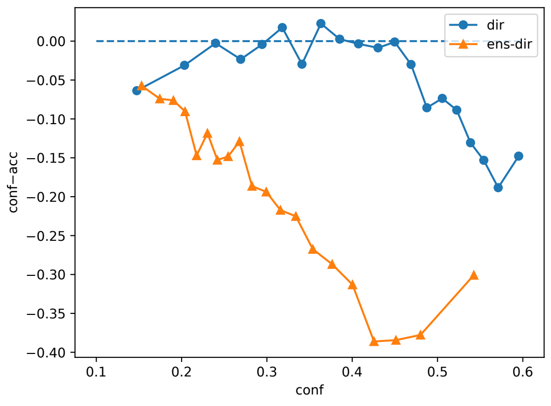
(a) CIFAR10→STL10, AlexNet



(b) STL10→CIFAR10, AlexNet



(c) CIFAR10→STL10, PreResNet20



(d) STL10→CIFAR10, PreResNet20

Figure 8: Confidence v.s. accuracy for Dirichlet model. X-axis is the confidence and y-axis is confidence minus accuracy. Confidence is expected to be equal to accuracy which corresponds to 0 in y-axis. Here we observe in most cases the value is below 0 indicating underconfidence.

## E.3 Entropy for OOD Data

Table 9, 10, 11, 12 list the entropy of OOD data.

Table 9: Entropy for OOD data on AlexNet between CIFAR10 and SVHN.

| **method** | **CIFAR10→SVHN** | **SVHN→CIFAR10** |
|---|---|---|
| vi-naive | $1.238 \pm 0.075$, 1.350 | $1.624 \pm 0.048$, 1.799 |
| vi-mean | $1.280 \pm 0.157$, 1.441 | $1.711 \pm 0.052$, 1.854 |
| vi-mv | $1.053 \pm 0.068$, 1.262 | $1.448 \pm 0.036$, 1.684 |
| vi-eb | $1.133 \pm 0.069$, 1.305 | $1.577 \pm 0.058$, 1.773 |
| vifo-naive | $0.805 \pm 0.040$, 1.338 | $1.106 \pm 0.023$, 1.707 |
| vifo-mean | $0.989 \pm 0.080$, 1.347 | $1.271 \pm 0.023$, 1.728 |
| vifo-mv | $0.999 \pm 0.075$, 1.433 | $1.218 \pm 0.045$, 1.709 |
| vifo-eb | $0.927 \pm 0.049$, 1.340 | $1.365 \pm 0.034$, 1.805 |
| base | $0.243 \pm 0.039$, 0.660 | $1.275 \pm 0.010$, 1.318 |
| swa | $0.256 \pm 0.018$, 0.609 | $1.336 \pm 0.003$, 1.344 |
| swag | $0.418 \pm 0.021$, 0.727 | $1.372 \pm 0.002$, 1.383 |
| dropout | $0.587 \pm 0.024$, 0.934 | $1.317 \pm 0.049$, 1.639 |
| dirichlet | $1.861 \pm 0.024$, 1.987 | $1.958 \pm 0.017$, 2.118 |

Table 10: Entropy for OOD data on AlexNet between STL10 and SVHN.

| **method** | **STL10→SVHN** | **SVHN→STL10** |
|---|---|---|
| vi-naive | $1.601 \pm 0.049$, 1.720 | $1.685 \pm 0.056$, 1.852 |
| vi-mean | $1.495 \pm 0.018$, 1.667 | $1.772 \pm 0.042$, 1.908 |
| vi-mv | $1.345 \pm 0.046$, 1.576 | $1.512 \pm 0.041$, 1.739 |
| vi-eb | $2.024 \pm 0.019$, 2.039 | $1.631 \pm 0.047$, 1.819 |
| vifo-naive | $1.270 \pm 0.091$, 1.775 | $1.131 \pm 0.014$, 1.735 |
| vifo-mean | $1.538 \pm 0.138$, 1.847 | $1.302 \pm 0.023$, 1.755 |
| vifo-mv | $1.589 \pm 0.147$, 1.894 | $1.265 \pm 0.054$, 1.751 |
| vifo-eb | $1.523 \pm 0.127$, 1.859 | $1.390 \pm 0.026$, 1.831 |
| base | $0.663 \pm 0.197$, 1.276 | $1.325 \pm 0.009$, 1.368 |
| swa | $1.358 \pm 0.447$, 1.669 | $1.393 \pm 0.003$, 1.400 |
| swag | $1.580 \pm 0.288$, 1.826 | $1.426 \pm 0.005$, 1.437 |
| dropout | $0.830 \pm 0.039$, 1.251 | $1.368 \pm 0.050$, 1.689 |
| dirichlet | $2.060 \pm 0.034$, 2.143 | $1.974 \pm 0.016$, 2.131 |

Table 11: Entropy for OOD data on PreResNet20 between CIFAR10 and SVHN.

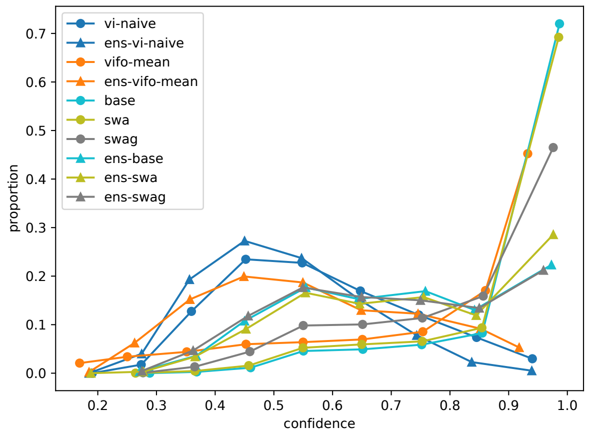
| **method** | **CIFAR10→SVHN** | **SVHN→CIFAR10** |
|---|---|---|
| vi-naive | $1.192 \pm 0.025$, 1.630 | $1.213 \pm 0.010$, 1.530 |
| vi-mean | $1.097 \pm 0.015$, 1.570 | $1.169 \pm 0.017$, 1.526 |
| vi-mv | $1.048 \pm 0.019$, 1.557 | $1.080 \pm 0.011$, 1.515 |
| vi-eb | $1.174 \pm 0.049$, 1.676 | $1.219 \pm 0.029$, 1.577 |
| vifo-naive | $0.827 \pm 0.016$, 1.601 | $0.884 \pm 0.078$, 1.617 |
| vifo-mean | $0.790 \pm 0.061$, 1.513 | $0.890 \pm 0.044$, 1.523 |
| vifo-mv | $0.837 \pm 0.054$, 1.543 | $0.978 \pm 0.072$, 1.606 |
| vifo-eb | $1.129 \pm 0.021$, 1.742 | $1.286 \pm 0.037$, 1.809 |
| base | $0.424 \pm 0.019$, 1.367 | $0.629 \pm 0.016$, 1.422 |
| swa | $0.427 \pm 0.021$, 1.340 | $0.634 \pm 0.016$, 1.374 |
| swag | $1.090 \pm 0.087$, 1.688 | $1.218 \pm 0.038$, 1.691 |
| dropout | $1.297 \pm 0.052$, 1.724 | $1.264 \pm 0.013$, 1.646 |
| dirichlet | $2.164 \pm 0.020$, 2.245 | $2.189 \pm 0.012$, 2.248 |

Table 12: Entropy for OOD data on PreResNet20 between STL10 and SVHN.

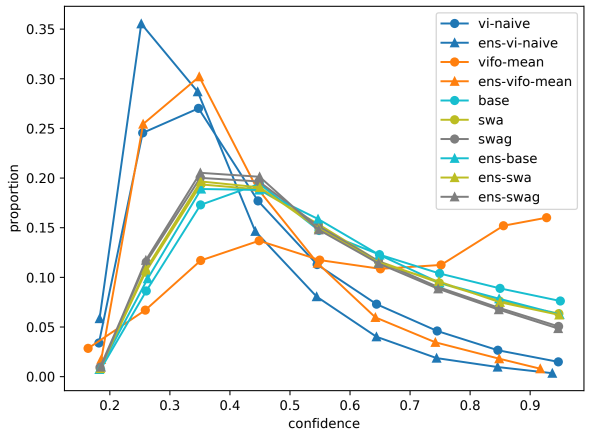
| **method** | **STL10→SVHN** | **SVHN→STL10** |
|---|---|---|
| vi-naive | $1.394 \pm 0.020$, 1.718 | $1.222 \pm 0.013$, 1.532 |
| vi-mean | $1.323 \pm 0.013$, 1.688 | $1.174 \pm 0.016$, 1.530 |
| vi-mv | $1.267 \pm 0.030$, 1.653 | $1.078 \pm 0.014$, 1.515 |
| vi-eb | $1.517 \pm 0.060$, 1.796 | $1.219 \pm 0.024$, 1.567 |
| vifo-naive | $1.041 \pm 0.013$, 1.759 | $0.893 \pm 0.072$, 1.622 |
| vifo-mean | $1.303 \pm 0.021$, 1.803 | $0.905 \pm 0.036$, 1.536 |
| vifo-mv | $0.797 \pm 0.025$, 1.555 | $0.982 \pm 0.058$, 1.609 |
| vifo-eb | $1.128 \pm 0.034$, 1.761 | $1.301 \pm 0.037$, 1.819 |
| base | $0.448 \pm 0.047$, 1.355 | $0.631 \pm 0.009$, 1.422 |
| swa | $0.617 \pm 0.126$, 1.420 | $0.637 \pm 0.011$, 1.377 |
| swag | $1.270 \pm 0.084$, 1.782 | $1.208 \pm 0.045$, 1.685 |
| dropout | $1.124 \pm 0.096$, 1.680 | $1.263 \pm 0.015$, 1.648 |
| dirichlet | $2.042 \pm 0.029$, 2.189 | $2.191 \pm 0.013$, 2.248 |

## E.4 Count-confidence Curves

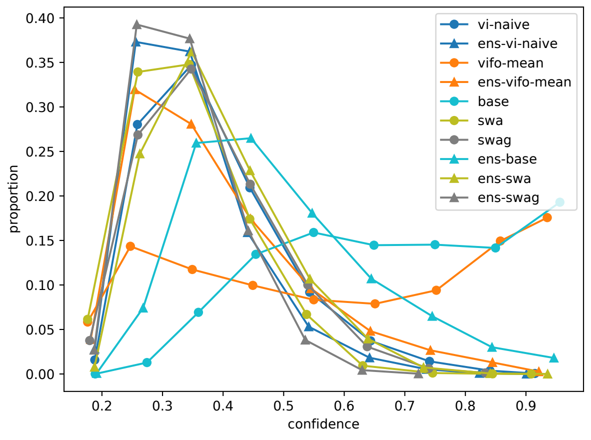
Figure 9 and 10 are the count-confidence curves. “A→B” in the subcaptions means that we train the model on dataset A and predict on a completely out-of-distribution dataset B. For each example in B, we obtain the class probabilities and choose the maximum probability to be the confidence score and the corresponding class label to be the prediction. Then we separate the interval $[0, 1]$ into 10 buckets of the same length and put all examples into these buckets according to their confidence scores. For each bucket, we compute the average confidence score and the proportion of examples that fall into this bucket and then plot the count-confidence curves. Since the dataset B is a complete out-of-distribution dataset for A, we want the confidence score to be as small as possible. So an algorithm with good uncertainty quantification will put more data in the low-confidence region and less data in the high confidence region. In this sense, we observe that the ensembles perform better than the single models, single-model VIFO performs better than base model and worse than VI and the ensemble of VIFO performs comparably to the ensemble of VI. Notice that the ensemble of VIFO costs much less time than the ensemble of VI, which confirms our conclusion in the main paper that the ensemble of VIFO provides the best tradeoff between run time and the uncertainty quantification.



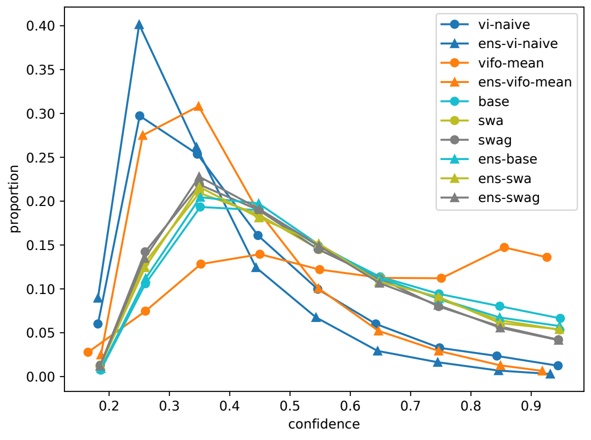
(a) CIFAR10→SVHN



(b) SVHN→CIFAR10

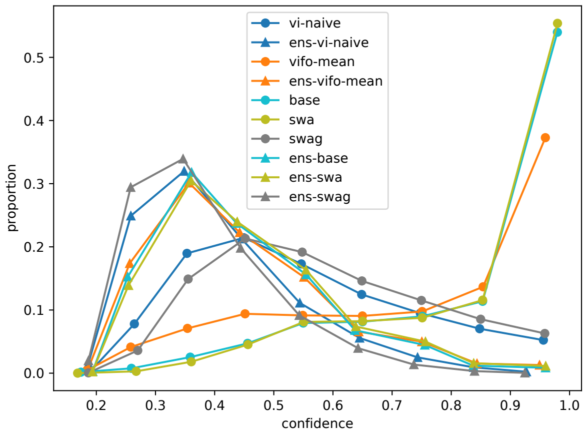


(c) STL10→STL10

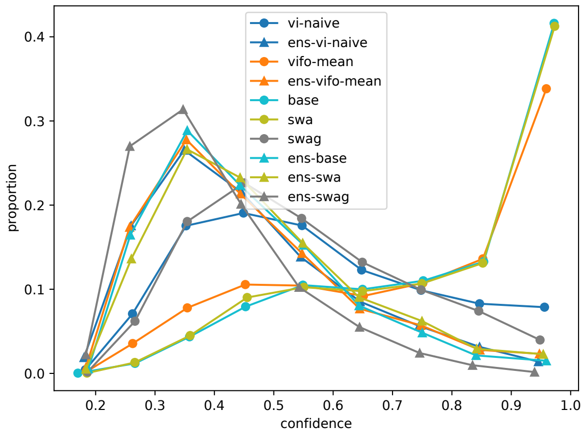


(d) STL10→SVHN

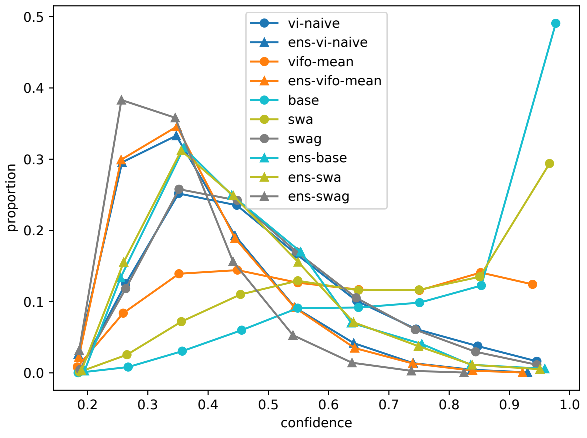
Figure 9: Count-confidence curves on AlexNet. X-axis is the confidence score, y-axis is the proportion of data that have the same range of confidence scores.



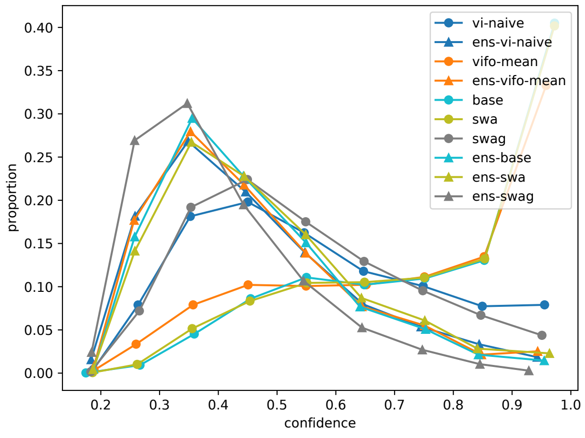
(a) CIFAR10→SVHN



(b) SVHN→CIFAR10



(c) STL10→STL10



(d) STL10→SVHN

Figure 10: Count-confidence curves on PreResNet20.

## E.5 AUROC on OOD Detection

Table 13, 14, 15, 16 show the AUROC for OOD detection.

Table 13: AUROC for OOD detection on AlexNet using maximum probability

| **method** | **CIFAR10→SVHN** | **SVHN→CIFAR10** |
|---|---|---|
| vi-naive | $0.898 \pm 0.016$, $0.909$ | $0.970 \pm 0.002$, $0.982$ |
| vi-mean | $0.886 \pm 0.029$, $0.908$ | $0.963 \pm 0.014$, $0.979$ |
| vi-mv | $0.893 \pm 0.009$, $0.916$ | $0.965 \pm 0.003$, $0.981$ |
| vi-eb | $0.885 \pm 0.010$, $0.906$ | $0.967 \pm 0.003$, $0.981$ |
| vifo-naive | $0.676 \pm 0.023$, $0.860$ | $0.810 \pm 0.011$, $0.966$ |
| vifo-mean | $0.730 \pm 0.040$, $0.873$ | $0.884 \pm 0.010$, $0.976$ |
| vifo-mv | $0.790 \pm 0.028$, $0.893$ | $0.895 \pm 0.016$, $0.972$ |
| vifo-eb | $0.723 \pm 0.019$, $0.860$ | $0.883 \pm 0.013$, $0.973$ |
| base | $0.826 \pm 0.018$, $0.858$ | $0.855 \pm 0.003$, $0.863$ |
| swa | $0.819 \pm 0.013$, $0.850$ | $0.871 \pm 0.001$, $0.866$ |
| swag | $0.819 \pm 0.013$, $0.848$ | $0.871 \pm 0.001$, $0.874$ |
| dropout | $0.832 \pm 0.003$ , $0.876$ | $0.961 \pm 0.002$ , $0.981$ |
| dirichlet | $0.736 \pm 0.047$ , $0.852$ | $0.896 \pm 0.017$ , $0.975$ |

Table 14: AUROC for OOD detection on AlexNet using maximum probability

| **method** | **STL10→SVHN** | **SVHN→STL10** |
|---|---|---|
| vi-naive | $0.818 \pm 0.020$, $0.856$ | $0.974 \pm 0.003$, $0.985$ |
| vi-mean | $0.792 \pm 0.011$, $0.836$ | $0.968 \pm 0.013$, $0.983$ |
| vi-mv | $0.775 \pm 0.018$, $0.833$ | $0.970 \pm 0.002$, $0.984$ |
| vi-eb | $0.736 \pm 0.044$, $0.736$ | $0.971 \pm 0.002$, $0.984$ |
| vifo-naive | $0.635 \pm 0.014$, $0.735$ | $0.822 \pm 0.006$, $0.970$ |
| vifo-mean | $0.707 \pm 0.050$, $0.809$ | $0.896 \pm 0.006$, $0.978$ |
| vifo-mv | $0.715 \pm 0.039$, $0.782$ | $0.908 \pm 0.011$, $0.976$ |
| vifo-eb | $0.706 \pm 0.032$, $0.791$ | $0.890 \pm 0.011$, $0.976$ |
| base | $0.696 \pm 0.024$, $0.770$ | $0.865 \pm 0.002$, $0.873$ |
| swa | $0.643 \pm 0.067$, $0.722$ | $0.882 \pm 0.001$, $0.878$ |
| swag | $0.643 \pm 0.067$, $0.689$ | $0.882 \pm 0.001$, $0.884$ |
| dropout | $0.711 \pm 0.019$, $0.770$ | $0.965 \pm 0.002$ , $0.984$ |
| dirichlet | $0.705 \pm 0.040$ , $0.764$ | $0.909 \pm 0.015$ , $0.979$ |

Table 15: AUROC for OOD detection on PreResNet20 using maximum probability

| **method** | **CIFAR10→SVHN** | **SVHN→CIFAR10** |
|---|---|---|
| vi-naive | $0.863 \pm 0.008$ , $0.939$ | $0.870 \pm 0.008$ , $0.913$ |
| vi-mean | $0.868 \pm 0.011$ , $0.943$ | $0.867 \pm 0.016$ , $0.910$ |
| vi-mv | $0.868 \pm 0.008$ , $0.942$ | $0.866 \pm 0.012$ , $0.916$ |
| vi-eb | $0.858 \pm 0.013$ , $0.946$ | $0.847 \pm 0.026$ , $0.902$ |
| vifo-naive | $0.630 \pm 0.018$, $0.900$ | $0.665 \pm 0.016$, $0.884$ |
| vifo-mean | $0.727 \pm 0.008$, $0.913$ | $0.625 \pm 0.021$ , $0.907$ |
| vifo-mv | $0.692 \pm 0.012$ , $0.920$ | $0.738 \pm 0.007$ , $0.908$ |
| vifo-eb | $0.787 \pm 0.014$ , $0.932$ | $0.841 \pm 0.009$ , $0.930$ |
| base | $0.792 \pm 0.007$ , $0.948$ | $0.785 \pm 0.030$ , $0.899$ |
| swa | $0.881 \pm 0.008$ , $0.948$ | $0.836 \pm 0.035$ , $0.910$ |
| swag | $0.881 \pm 0.008$ , $0.963$ | $0.836 \pm 0.035$ , $0.922$ |
| dropout | $0.878 \pm 0.010$ , $0.950$ | $0.852 \pm 0.017$ , $0.909$ |
| dirichlet | $0.844 \pm 0.079$ , $0.963$ | $0.924 \pm 0.022$ , $0.972$ |

Table 16: AUROC for OOD detection on PreResNet20 using maximum probability

| **method** | **STL10→SVHN** | **SVHN→STL10** |
|---|---|---|
| vi-naive | $0.788 \pm 0.014$, $0.870$ | $0.870 \pm 0.010$, $0.913$ |
| vi-mean | $0.784 \pm 0.006$, $0.869$ | $0.867 \pm 0.016$, $0.910$ |
| vi-mv | $0.788 \pm 0.015$, $0.875$ | $0.865 \pm 0.011$, $0.915$ |
| vi-eb | $0.728 \pm 0.030$ , $0.824$ | $0.846 \pm 0.027$, $0.900$ |
| vifo-naive | $0.621 \pm 0.005$, $0.806$ | $0.669 \pm 0.013$, $0.885$ |
| vifo-mean | $0.661 \pm 0.009$, $0.822$ | $0.634 \pm 0.016$, $0.908$ |
| vifo-mv | $0.596 \pm 0.009$, $0.796$ | $0.741 \pm 0.005$, $0.909$ |
| vifo-eb | $0.655 \pm 0.019$, $0.821$ | $0.845 \pm 0.009$, $0.931$ |
| base | $0.628 \pm 0.006$, $0.837$ | $0.787 \pm 0.032$, $0.899$ |
| swa | $0.706 \pm 0.029$, $0.838$ | $0.832 \pm 0.036$, $0.910$ |
| swag | $0.706 \pm 0.029$, $0.865$ | $0.832 \pm 0.036$, $0.921$ |
| dropout | $0.745 \pm 0.037$, $0.880$ | $0.852 \pm 0.016$, $0.910$ |
| dirichlet | $0.658 \pm 0.048$ , $0.815$ | $0.923 \pm 0.025$ , $0.972$ |