Incorporating functional summary information in Bayesian neural networks using a Dirichlet process likelihood approach

Vishnu Raj **Tianyu Cui** **Markus Heinonen** **Pekka Marttinen**
Department of Computer Science
Aalto University
Finland

Abstract

Bayesian neural networks (BNNs) can account for both aleatoric and epistemic uncertainty. However, in BNNs the priors are often specified over the weights which rarely reflects true prior knowledge in large and complex neural network architectures. We present a simple approach to incorporate prior knowledge in BNNs based on external summary information about the predicted classification probabilities for a given dataset. The available summary information is incorporated as augmented data and modeled with a Dirichlet process, and we derive the corresponding *Summary Evidence Lower BOund*. The approach is founded on Bayesian principles, and all hyperparameters have a proper probabilistic interpretation. We show how the method can inform the model about task difficulty and class imbalance. Extensive experiments show that, with negligible computational overhead, our method parallels and in many cases outperforms popular alternatives in accuracy, uncertainty calibration, and robustness against corruptions with both balanced and imbalanced data.

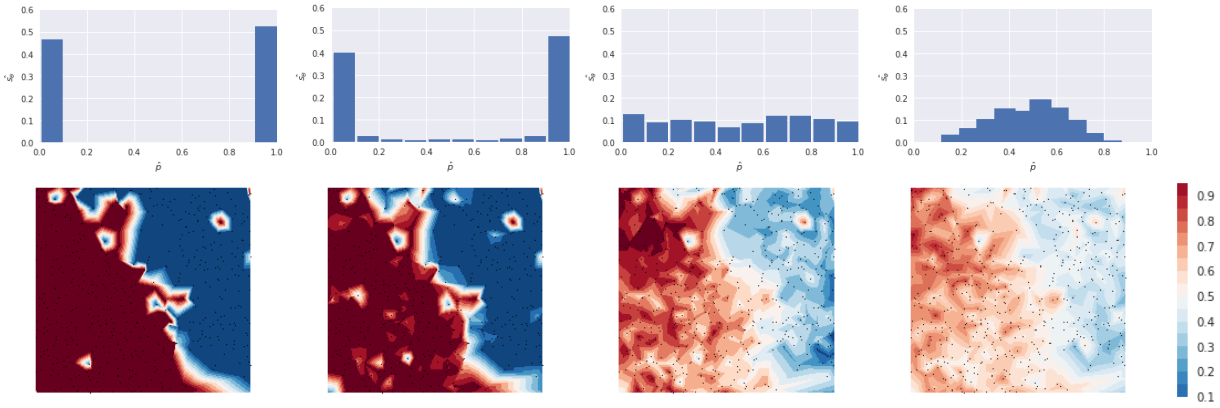
1 Introduction

Modern deep learning has opened up a plethora of possibilities that previously seemed impossible. Leveraging function approximation capabilities of neural networks, modern deep learning can tackle challenging problems (Esteva et al., 2019; George and Huerta, 2018; Silver et al., 2016), but the black-box nature of neural networks hinders researchers from developing insights into the model's predictions, and the issue is amplified in settings where uncertainty quantification is required. On the other hand, model uncertainty should be calibrated in critical areas such as healthcare and autonomous driving. Bayesian modeling enables a coherent probabilistic perspective for machine learning (Murphy, 2012) and provides valuable tools for data analysis (Gelman et al., 2014). Bayesian neural networks (BNNs) offer a formal framework with promises of improved predictions, reliable uncertainty estimates, principled model comparison, etc (Wilson, 2020; Wilson and Izmailov, 2020).

While many works in Bayesian neural networks focus on specifying priors over model parameters (Graves, 2011; Blundell et al., 2015) and functional outputs (Flam-Shepherd et al., 2017; Tran et al., 2020; Sun et al., 2019), there is a surprising gap in incorporating prior knowledge about *summary statistics* of the functional outputs. Such a prior could help in improving uncertainty quantification and calibration of Bayesian neural networks. Calibration of neural network predictions is a widely studied topic (Guo et al., 2017; Minderer et al., 2021; Wang et al., 2021) and methods such as posterior tempering (Wenzel et al., 2020) have been developed. However, these approaches typically deviate from the strictly Bayesian approach by modifying the prior or likelihood with additional parameters. Consequently, we study how to incorporate summary statistics information available about a classification task in a fully Bayesian manner. We introduce a formulation where the shape of the distribution of the predicted probabilities (such as sigmoid/softmax scores) is available as prior knowledge, and we demonstrate how such a summary can be informative, e.g., about the difficulty of classification or class imbalance (See Fig. 1). Technically, we augment the observed data with this summary, expand the likelihood with a Dirichlet process term for the summary, and derive a formal ELBO for variational training. Through empirical evaluation in multiple classification tasks, we show the proposed approach is able to improve the calibration, robustness and uncertainty of BNNs while maintaining their accuracy.

The main contributions of this work are,

Proceedings of the 26th International Conference on Artificial Intelligence and Statistics (AISTATS) 2023, Valencia, Spain. PMLR: Volume 206.



(a) Regular BNN (b) $\mathbf{s}_0 = \text{Beta}(0.1, 0.1)$ (c) $\mathbf{s}_0 = \text{Beta}(1.0, 1.0)$ (d) $\mathbf{s}_0 = \text{Beta}(5.0, 5.0)$

Figure 1: Posterior distribution of predicted sigmoid scores with different summary statistic observations $\mathbf{s}_0$ in binary classification in MNIST. Top row: Posterior distribution of predicted sigmoid samples. Bottom row: Decision surface of the model with samples projected in 2D using t-SNE and colored corresponding to the predicted sigmoid scores. We see from Fig. 1a that the regular BNN predicts scores peaked towards 0 or 1, indicating possible overconfidence. By introducing a likelihood term for the summary statistic the proposed *Summary ELBO* is able to control how the predicted sigmoid scores are distributed, and Figs. 1b - 1d show that different $\mathbf{s}_0$ can yield different predicted sigmoid score histograms. This is also evident in the decision surface; the regular BNN has a sharp decision boundary with extreme predicted values while the *Summary ELBO* yields a smoother decision surface.

1. We propose a fully Bayesian approach to incorporate summary information into Bayesian neural networks

2. We introduce how different summary information such as confidence in predictions or class imbalance can be incorporated during model training using the augmented likelihood.

3. Through comprehensive empirical studies in computer vision and natural language processing, we show that the additional knowledge can in most cases significantly improve the performance of BNNs, especially with corrupted test data or imbalanced classes.

2 Background

2.1 Bayesian neural networks and variational inference

Different from deterministic neural networks (NNs), Bayesian neural networks (BNNs) (MacKay, 1992; Neal, 2012) are commonly defined by placing a prior distribution $p(\boldsymbol{\theta})$ on the weights $\boldsymbol{\theta}$ of a NN. Moreover, instead of only finding point estimates for weights $\boldsymbol{\theta}$, a posterior distribution of the weights is computed conditionally on the data according to the Bayes' theorem. Specifically, given a dataset $\mathcal{D} = \{\mathbf{X}, \mathbf{Y}\}$ with inputs $\mathbf{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ and outputs $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_N\}$, we have the likelihood $p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta}) = p(\mathbf{Y}|f(\mathbf{X}; \boldsymbol{\theta}))$ of a BNN on the dataset where $f(\mathbf{X}; \boldsymbol{\theta})$ is the prediction of the BNN parameterized by $\boldsymbol{\theta}$. Then, training a BNN means computing the posterior distribution $p(\boldsymbol{\theta}|\mathbf{X}, \mathbf{Y}) = p(\mathbf{Y}|f(\mathbf{X}; \boldsymbol{\theta}))p(\boldsymbol{\theta})/p(\mathbf{Y}|\mathbf{X})$, and we predict a new data point $(\mathbf{x}_\star, \mathbf{y}_\star)$ by marginalizing out $\boldsymbol{\theta}$ from the likelihood according to its posterior $p(\mathbf{y}_\star|\mathbf{x}_\star, \mathbf{X}, \mathbf{Y}) = \int p(\mathbf{y}_\star|f(\mathbf{x}_\star; \boldsymbol{\theta}))p(\boldsymbol{\theta}|\mathbf{X}, \mathbf{Y})d\boldsymbol{\theta}$. Unfortunately, neither the posterior of weights nor the predictive distribution of the new data is analytically tractable for BNNs.

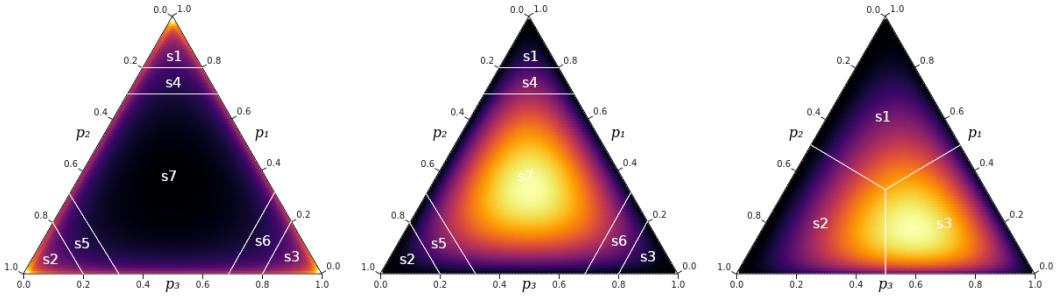
Variational inference can be used to approximate the intractable $p(\boldsymbol{\theta}|\mathbf{X}, \mathbf{Y})$ with a simpler distribution, $q_\phi(\boldsymbol{\theta})$, by minimizing $\text{KL}(q_\phi(\boldsymbol{\theta})||p(\boldsymbol{\theta}|\mathbf{X}, \mathbf{Y}))$. This is equivalent to maximizing the Evidence Lower BOund (ELBO) (Bishop, 2006)

$$\mathcal{L}(\phi) = \text{E}_{q_\phi(\boldsymbol{\theta})}[\log p(\mathbf{Y}|\mathbf{X}, \boldsymbol{\theta})] - \text{KL}[q_\phi(\boldsymbol{\theta})|p(\boldsymbol{\theta})], \quad (1)$$

where the first term is the expected log-likelihood and the second term measures the divergence between the posterior and the prior. ELBO and its gradients with respect to ϕ can be computed by backpropagation with the reparametrization trick (Kingma and Welling, 2013). Therefore, the posterior predictive distribution can be approximated by

$$\begin{aligned} p(\mathbf{y}_\star|\mathbf{x}_\star, \mathbf{X}, \mathbf{Y}) &\approx \int p(\mathbf{y}_\star|f(\mathbf{x}_\star; \boldsymbol{\theta}))q_\phi(\mathbf{w})d\boldsymbol{\theta} \\ &\approx \frac{1}{M}\sum_{l=1}^{M} p(\mathbf{y}_\star|f(\mathbf{x}_\star; \boldsymbol{\theta}^{(l)})), \end{aligned} \quad (2)$$

where $\boldsymbol{\theta}^l \sim q_\phi(\boldsymbol{\theta})$ and M is the number of Monte Carlo samples drawn from posterior distribution.



(a) Easy task, balanced data (b) Difficult task, balanced data (c) Imbalanced data

Figure 2: Different examples for selecting $\mathbf{s}_0$ to reflect prior information in a 3 class classification setting. Figure shows the heatmap of the distribution of sigmoid scores where dark means low probability and bright means high probability. Here, $\mathbf{s}_0$ is a Dirichlet distribution and, by selecting the parameters of the Dirichlet distribution appropriately, our method provides a flexible and principled approach to incorporate prior information on the difficulty of the classification task and class imbalance. The figure also shows the binning of the simplex into a finite number of regions required by the finite approximation in DP inference. See text for details.

2.2 Dirichlet processes

The Dirichlet process (DP) (Teh, 2010) is a stochastic process widely used in Bayesian nonparametrics. Different from Gaussian processes (GPs) (Seeger, 2004), which model distributions over functions with Gaussian marginals, DPs are stochastic processes over probability measures with Dirichlet marginals. In machine learning, DPs have been used as an infinite-dimensional generalization of the Dirichlet distribution in mixture models (Neal, 1992) and in topic modeling (Teh et al., 2006). A DP, $G \sim \mathcal{DP}(H, \alpha)$, is parameterized by the *base measure* H, which is a distribution H over a probability space Θ, and the *concentration parameter* α, a positive real number, such that

$$(G(A_1), \ldots, G(A_b)) \sim \mathrm{Dir}(\alpha H(A_1), \ldots, \alpha H(A_b)) \quad (3)$$

for every finite measurable partition $\{A_1, \ldots, A_b\}$ of Θ. The base measure H is the mean of the DP, i.e., for any $A \subset \Theta$, $\mathbb{E}[G(A)] = H(A)$, and it specifies the overall shape of G. The concentration parameter α serves as the inverse variance of the DP (Teh, 2010), such that a large α will force G to be close to H, see Fig. 8 in Appendix for examples of sampled distributions from a DP with different α.

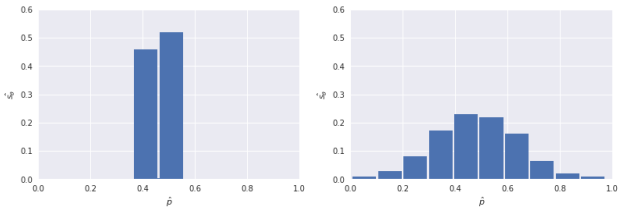
3 Incorporating summary information

We consider multiclass classification using a Bayesian neural network, where target $\mathbf{y}_i$ is encoded label, such as the one hot encoding. Let $\tilde{\mathbf{y}}_i = f(\mathbf{x}_i; \boldsymbol{\theta})$ be the prediction by the neural network for input $\mathbf{x}_i$. In binary classification, we have $\tilde{\mathbf{y}}_i \in [0, 1]$ representing the probability of one of the classes. In multiclass classification, $\tilde{\mathbf{y}}_i \in [0, 1]^K$ with $\sum_{k=1}^{K} \tilde{y}_{ik} = 1$, where $\tilde{y}_{ik}$ is the probability of class k and K is the number of classes.

Ideally, $\tilde{\mathbf{y}}_i$ would be equal to $\mathbf{y}_i$, corresponding to the perfect prediction. However, in practice in multi-class classification we get $\tilde{\mathbf{y}}_i \in [0, 1]^K$, where each entry in the predicted vector $\tilde{\mathbf{y}}_i$ is the normalized score of a particular class, corresponding to the probability that the sample belongs to the class. Here, we assume that the modeler has access to a summary statistic $\mathbf{s}_0$ representing the how the predicted probabilities $\tilde{\mathbf{y}}_i$ are distributed over the dataset. For example, in binary classification the summary statistic $\mathbf{s}_0$ is a distribution in the range $[0, 1]$ (e.g. a Beta distribution, Fig. 1) and in multiclass classification $\mathbf{s}_0$ is a distribution over the prediction simplex (e.g. a Dirichlet, Fig. 2).

The summary statistic $\mathbf{s}_0$ is then used for controlling the distribution of predicted sigmoid/softmax scores $\tilde{\mathbf{y}}_i$ according to available prior knowledge. In practice this prior knowledge should reflect considerations external to the current data set. For example, if we know that the classification task is easy and well separable, we expect $\tilde{\mathbf{y}}_i$ to take values where one entry is close to 1 while others are close to 0, which we can represent with a summary statistic $\mathbf{s}_0 = \mathrm{Dir}(\alpha_1, \alpha_2, \ldots, \alpha_k)$, with all parameters $\alpha_k < 1$ (see Fig. 2a). Conversely, if we know that the dataset is noisy and/or not easily separable, we would expect the predictions $\tilde{\mathbf{y}}_i$ to concentrate towards the center of the prediction simplex, which could be represented with a Dirichlet with $\alpha_k > 1$ (Fig. 2b). Furthermore, the relative magnitudes of the different α_k parameters can inform about the frequencies of different labels in imbalanced data (Fig. 2c).

As another point to emphasize, our approach considers how $\tilde{\mathbf{y}}_i$ are collectively (i.e. jointly) distributed in the prediction simplex for all samples. Another option would be to model $\tilde{\mathbf{y}}_i$ with a Dirichlet separately for each i, similar to Sensoy



(a) Independent Dirichlet (b) DP modeling

Figure 3: We show the posterior distribution of $\tilde{\mathbf{y}}_i$s when using $\mathbf{s}_0 = \text{Beta}(5.0, 5.0)$ with different approaches. Modeling sigmoid outputs independently can result in all predictions concentrating near the mode of the distribution as shown in Fig. 3a. However, the proposed DP modeling avoids this and tries instead to match the whole distribution of the scores set by $\mathbf{s}_0$ as shown in Fig. 3b.

et al. (2018). However, this would make all $\tilde{\mathbf{y}}_i$s concentrate near the mode of the Dirichlet (see Fig. 3). Instead, we want to control how $\tilde{\mathbf{y}}_i$s are distributed across all samples from training dataset $\mathcal{D}$, indicating that the dataset will contain both easy and difficult samples to classify.

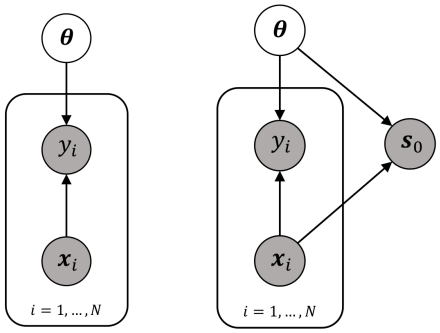
3.1 Incorporating prior knowledge through sequential Bayesian inference

We want to train a Bayesian neural network $f(\mathbf{x}; \boldsymbol{\theta})$ for classification from a dataset $\mathcal{D} = (\mathbf{X}, \mathbf{Y}) = \{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^N$ of N observations from the input space $\mathbf{x}_i \in \mathcal{X}$ and output space $\mathbf{y}_i \in \mathcal{Y}$ with K labels, where $\boldsymbol{\theta}$ denote the parameters of the neural network. We also consider a summary statistic $\mathbf{s}_{\boldsymbol{\theta}} = S(\boldsymbol{\theta}, \mathbf{X}) = S(f(\mathbf{x}_1; \boldsymbol{\theta}), \ldots, f(\mathbf{x}_N; \boldsymbol{\theta}))$, where the function S calculates the distribution of the *predicted* sigmoid scores $f(\mathbf{x}_i; \boldsymbol{\theta})$ in the training set. This can be a continuous or discrete density estimator. In addition, we denote with $\mathbf{s}_0$ the *observed* summary statistic, which corresponds to the distribution of sigmoid/softmax scores available from prior knowledge, representing information about label distribution (in case of class imbalance), or mass in different parts of the prediction simplex (in case of difficult classification tasks).

Assume now that we have a prior $p(\boldsymbol{\theta})$ for the neural network weights, and we observe an augmented data $\mathcal{D}_{\text{aug}} = (\mathcal{D}, \mathbf{s}_0)$ where $\mathbf{s}_0$ is the observed summary statistic and $\mathcal{D} = (\mathbf{X}, \mathbf{Y})$ are the observations. Fig. 4 shows a graphical model assumed by a traditional BNN and compares that with our joint model with summary information $\mathbf{s}_0$. Specifically, we assume that the joint distribution factorizes as follows:

$$p(\mathbf{X}, \mathbf{Y}, \mathbf{s}_0, \boldsymbol{\theta}) = \left[\prod_{i=1}^{N} p(\mathbf{y}_i|\mathbf{x}_i, \boldsymbol{\theta})\right] p(\mathbf{s}_0|\mathbf{X}, \boldsymbol{\theta}) p(\boldsymbol{\theta}). \quad (4)$$

Hence, the approach can be formally seen as sequential Bayesian inference, which first updates the non-informative prior $p(\boldsymbol{\theta})$ into an informative prior by multiplying with the *summary likelihood*, $p(\mathbf{s}_0|\mathbf{X}, \boldsymbol{\theta})$, and then uses the informative prior for modeling the data $(\mathbf{X}, \mathbf{Y})$ using the regular



(a) BNN (b) BNN with summary

Figure 4: Graphical model for BNNs. Fig. 4a shows the graphical model for vanilla BNN where model parameters $\boldsymbol{\theta}$ are only related to label $\mathbf{y}$. In our proposed summary likelihood model Fig. 4b, we show how to model additional summary statistics information. We model the summary information $\mathbf{s}_0$ as derived from input variable $\mathbf{x}$ and model parameters $\boldsymbol{\theta}$, and is an observed node in the model.

likelihood. We define the summary likelihood as

$$p(\mathbf{s}_0|\mathbf{X}, \boldsymbol{\theta}) = \mathcal{DP}(\mathbf{s}_0|\mathbf{s}_{\boldsymbol{\theta}}, \alpha). \quad (5)$$

In other words, the observed summary $\mathbf{s}_0$ is distributed as a Dirichlet process whose base measure is equal to $\mathbf{s}_{\boldsymbol{\theta}} = S(\boldsymbol{\theta}, \mathbf{X})$, i.e., the histogram of sigmoid/softmax outputs predicted by the NN for the training data, and a concentration hyperparameter α. Consequently, with a large α the predicted and observed summary statistics $\mathbf{s}_{\boldsymbol{\theta}}$ and $\mathbf{s}_0$ are expected be close to each other. The model definition is completed by defining the likelihood as a categorical distribution:

$$p(\mathbf{y}_i|\mathbf{x}_i, \boldsymbol{\theta}) = \text{Cat}\big(\mathbf{y}_i|f(\mathbf{x}_i, \boldsymbol{\theta})\big), \text{for all } i, \quad (6)$$

and the prior conventionally as $p(\boldsymbol{\theta}) = N_{\boldsymbol{\theta}}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

One way to think about the summary statistic, $\mathbf{s}_0$, is to interpret it as a 'pseudo-observation'; pseudo-observations are often used to interpret common priors (Gelman et al., 2014). To understand why the prior knowledge is incorporated through another likelihood term $p(s_0|X_{obs}, \theta)$, it is instructive to notice that a prior on the parameters θ, $p(\theta)$, already induces a prior on the distribution of outputs, s_θ, which we can here denote by $p_\theta(s_\theta)$. In general, when there exists some prior knowledge, captured by s_0, about s_θ, an obvious thing would be to define a prior distribution, something like $p(s_\theta|s_0)$, or equivalently a joint distribution $p(s_\theta, s_0)$. The problem is that there can't be two prior distributions: $p_\theta(s_\theta)$ and $p(s_\theta|s_0)$, for the same quantity s_θ. Instead, we calculate $p(s_\theta|s_0)$ according to the formal Bayesian procedure where we update the initial prior distribution $p_\theta(s_\theta)$ into $p(s_\theta|s_0)$ using the Bayes' rule, which happens through the multiplication of the previous prior using a likelihood term. Consequently, as we show later, this yields a well-defined

ELBO corresponding to proper Bayesian inference. Further, Gelman (2021) suggests to consider one prior as data when there are two sources of prior knowledge for the same parameter instead of two priors, because the former is more consistent with Bayesian theory.

3.2 Inference with summary ELBO

In this setting our goal is simply to infer the parameter posterior $p(\boldsymbol{\theta}|\mathcal{D})$, which we approximate variationally with $q_\phi(\boldsymbol{\theta})$. This induces an ELBO

$$\begin{aligned}
\mathcal{L}(\phi) &= \mathbb{E}_{q_\phi(\boldsymbol{\theta})} \log p(\mathcal{D}|\boldsymbol{\theta}) - \mathrm{KL}[q_\phi(\boldsymbol{\theta})||p(\boldsymbol{\theta})] \\
&= \mathbb{E}_{q_\phi(\boldsymbol{\theta})} \left[\sum_{i=1}^{N} \log p(\mathbf{y}_i|\mathbf{x}_i, \boldsymbol{\theta}) + \log p(\mathbf{s}_0|\mathbf{X}, \boldsymbol{\theta}) \right] \\
&\quad - \mathrm{KL}[q_\phi(\boldsymbol{\theta})||p(\boldsymbol{\theta})] \\
&= \mathbb{E}_{q_\phi(\boldsymbol{\theta})} \Bigg[\sum_{i=1}^{N} \log \mathrm{Cat}\big(\mathbf{y}_i|f(\mathbf{x}_i, \boldsymbol{\theta})\big) \\
&\quad + \log \mathcal{DP}\big(\mathbf{s}_0|\mathbf{s}_{\boldsymbol{\theta}}, \alpha\big)\Big] \\
&\quad - \mathrm{KL}\Big[\mathcal{N}_{\boldsymbol{\theta}}(\boldsymbol{\mu}_\phi, \boldsymbol{\Sigma}_\phi)\,||\,\mathcal{N}_{\boldsymbol{\theta}}(\mathbf{0}, \sigma^2 I)\Big] \\
&\approx \frac{1}{M} \sum_{i=1}^{N} \sum_{j=1}^{M} \log \mathrm{Cat}\big(\mathbf{y}_i|f(\mathbf{x}_i, \boldsymbol{\theta}_j)\big) \\
&\quad + \frac{1}{M} \sum_{j=1}^{M} \log \mathcal{DP}\big(\mathbf{s}_0|\mathbf{s}_{\boldsymbol{\theta}_j}, \alpha\big) \\
&\quad - \mathrm{KL}\Big[\mathcal{N}_{\boldsymbol{\theta}}(\boldsymbol{\mu}_\phi, \boldsymbol{\Sigma}_\phi)\,||\,\mathcal{N}_{\boldsymbol{\theta}}(\mathbf{0}, \sigma^2 I)\Big], \quad (7)
\end{aligned}$$

where $\{\boldsymbol{\theta}_j\}_{j=1}^{M} \sim q_\phi(\boldsymbol{\theta})$ are the samples from the inferred posterior and M is the number of Monte Carlo samples. Hence, compared to the traditional ELBO (Kingma and Welling, 2013; Neal, 1992), our objective, *Summary ELBO*, defined in (7), incorporates prior information about the modeler's belief on how the predictions $\tilde{\mathbf{y}}_i$ should be jointly distributed, as captured by the observed summary statistic $\mathbf{s}_0$.

3.3 Computation of the DP summary likelihood term

One challenge in the objective function (7) is the computation of summary likelihood involving the DP (second term). As a closed form expression is unavailable (Teh, 2010), we use a finite partition approximation, where the parameter space Θ is divided into a finite number of bins $\{A_1, \ldots, A_b\}$ corresponding to discretized histograms and the likelihood is evaluated using Eqn. (3). In the binary experiments, the softmax scores are distributed in the $[0, 1]$ interval, which we divide into multiple bins, as demonstrated in Fig.1a-1d. In the multiclass classification we divide the prediction simplex symmetrically into regions where some regions are more central and some in the corners of the simplex, allowing us to express prior knowledge about task difficulty, i.e., how much of the probability mass should be given to uncertain predictions corresponding to sigmoid scores near 0.5 and how much to confident predictions with scores close to 1, demonstrated in Figs 2a and 2b. In the imbalanced data experiment, we use a partition shown in Fig. 2c, which accounts for the total mass allocated to each predicted class, but is agnostic about how far the score is from the center of the simplex.

In practice, the predictions $\tilde{\mathbf{y}}_i$ from the model at each training step are collected from the entire minibatch and a histogram over the specified regions is constructed. To be able to backpropagate through the operation, we use SoftHistogram to construct the histogram in our experiments. The SoftHistogram function identifies the total mass in each bin using a pair of sigmoid functions and aggregating over the minibatch. Details of SoftHistogram construction is discussed in Appendix C. The major bottleneck in using this approximation is the quality of SoftHistogram results. To address this, we use a moderately large minibatch size to make the estimation less noisy.

4 Related works

Functional BNNs priors Our approach can be seen as a way to incorporate summary information about the predictive distribution, and hence it is conceptually related to functional priors. Previously, Gaussian processes have been proposed to encode rich functional structures as prior knowledge. Flam-Shepherd et al. (2017) and Tran et al. (2020) transformed a functional GP prior into a weight-space BNN prior by minimizing the Kullback–Leibler divergence and Wasserstein distance respectively. Functional BNNs (Sun et al., 2019) performed variational inference directly with GP priors. Other recent works which concern with the output behavior include Noise contrastive priors (NCPs) (Hafner et al., 2018) and Output-Constrained BNNs Yang et al. (2020). A comprehensive review of deep learning priors is given in Fortuin (2021).

Weight-space BNN priors In the weight space, a fully factorized Gaussian prior has been proposed by Graves (2011) and Blundell et al. (2015), and interpreted as equivalent to dropout when using a mixture of Dirac-deltas as the variational posterior (Gal and Ghahramani, 2016). Nalisnick et al. (2019) extended these works and interpreted NNs with any multiplicative noise as BNNs with a Gaussian scale mixture prior (Andrews and Mallows, 1974) and Automatic Relevance Determination (ARD) (MacKay, 1994). Moreover, low-rank priors, such as the k-tied normal (Swiatkowski et al., 2020) and rank-1 perturbation (Dusenberry et al., 2020), were combined with ensemble methods Lakshminarayanan et al. (2017) to capture multiple modes, and they had better convergence rates. To model the correlation between the weights, Matrix-variate Gaussian priors were proposed by Neklyudov et al. (2017) and Sun et al. (2017). Also

sparse priors have been defined, such as the log-uniform (Molchanov et al., 2017; Louizos et al., 2017), log-normal (Neklyudov et al., 2017), horseshoe (Louizos et al., 2017; Ghosh et al., 2018), and spike-and-slab priors (Deng et al., 2019). Cui et al. (2021b) proposed a two-stage procedure to encode the prior knowledge about the data signal-to-noise ratio into a Gaussian scale mixture prior. Overall, it is often challenging to incorporate more general domain knowledge other than sparsity into the weight-space priors.

Evidential Deep Learning Different from ordinary deep learning, which is trained to predict the parameters of the likelihood function with Maximum Likelihood, evidential deep learning (EDL) is trained to predict the parameters of likelihood with the Type II Maximum Likelihood (ML-II, i.e., maximizing the model evidence). Therefore, the model predictions, as well as the aleatoric and epistemic uncertainty estimations, come from the learned prior of the likelihood. In the classification setting, Sensoy et al. (2018); Malinin and Gales (2018) proposed to learn a Dirichlet prior of the categorical likelihood parameters, and in regression, Amini et al. (2020) learned a Normal Inverse-Gamma prior of the Gaussian likelihood. Although EDL provides a reasonable uncertainty estimation, a heuristic regularization on evidence has to be applied to avoid over-fitting due to the ML-II.

Non-Bayesian approaches of incorporating domain knowledge Sophisticated regularization techniques, i.e., explanation prior, have been proposed for deterministic NNs to incorporate extra domain knowledge (Ross et al., 2017). When the importance score of each feature is known *a priori*, attribution priors were proposed to regularize the feature importance of the model to agree with the prior importance score, such as DeepSHAP (Tseng et al., 2020) and Contextural Decomposition (Rieger et al., 2020). DAPr (Weinberger et al., 2020) matched the feature attribution to a learned prior feature importance from meta-features. In a genetics application, MEP Cui et al. (2021a) was proposed to incorporate the feature main effects (i.e., linear regression coefficients) on an external large dataset into NNs on a small dataset.

5 Experimental results

Here, we show the utility of the proposed method, abbreviated as **S-ELBO** for *Summary ELBO*, in classification tasks from computer vision and natural language processing domains. Specifically, we show results for image classification and sentiment analysis. The experiments cover both binary and multiclass classification tasks. In all the cases, Mean Field Variational Inference (MFVI) with $\mathcal{N}(\mathbf{0}, \sigma_0^2 \boldsymbol{I})$ prior is used to train the neural networks. We cross-validate the prior variance $\sigma_0 \in \{0.10, 0.25, 0.50, 1.00, 2.00\}$, but as the results are not sensitive to this choice (Fig. 9 in Appendix), we use the default value $\sigma_0 = 1$ throughout. For the numerical results, each model is trained 5 times independently, and the mean and standard error of each metric are reported. The code is implemented in PyTorch Paszke et al. (2019) and available at `github.com/v-i-s-h/summary-likelihood`.

For modeling the Dirichlet Process likelihood term, we cross validate the concentration parameter $\alpha \in \{10, 50, 100, 500, 1000, 2500, 5000, 10000\}$ on a separate validation set. For the summary prior histogram s_0 in the DP likelihood, we cross validate between the uniform distribution and a distribution based on automatic parameter selection as described in Appendix B. Models are trained for 3000 steps for binary and 5000 steps for multiclass classification problems. We use minibatch size of 256 and Adam optimizer with a constant learning rate 10^{-3}. We report the negative log-likelihood (NLL), accuracy, and Expected Calibration Error (ECE) (Guo et al., 2017) in clean (in-domain) and corrupted test sets. In the detection of out-of-distribution samples we report the difference in predictive entropy Δ_{OOD} between OOD and in-domain samples, and in the multi-class classification with imbalanced classes we report the F1 score. All scores are reported for held-out test sets. During cross validation, the optimal hyperparameters are chosen based on the NLL. We compare against the vanilla BNN trained with the standard ELBO (**ELBO**) (Blundell et al., 2015) , Evidential Deep Learning (**EDL**) (Sensoy et al., 2018) and Label Smoothing (**LS**), all sharing the same NN architecture in the same task. ELBO and LS are trained with MFVI, using the same prior as S-ELBO. For LS, we cross validate the smoothing factor $\epsilon \in \{0.01, 0.05, 0.10\}$. For EDL, we use the setup recommended in Sensoy et al. (2018), and train it in multiclass classification with an annealing step of 1000.

5.1 Sentiment analysis task

Sentiment analysis is an NLP task of classifying the polarity of a given text, usually posed as binary classification. However, the analysis of phrases from each of the sample texts (Socher et al., 2013) shows that the contituent phrases can have intermediate values of sentiment, not fully captured by the binary labels. Hence, sentiment analysis is a perfect example of a classification task where labels are not always too confident. We use Stanford Sentiment Treebank Socher et al. (2013) as our source data with labels and use Sentence-BERT (Reimers and Gurevych, 2019) to compute a 768 dimensional embedding for each sample text. A feed-forward BNN with a single hidden layer of dimension 128 is trained on these embeddings using the alternative methods. For training the proposed method, a uniformly distributed prior histogram $\mathbf{s}_0$ is assumed, reflecting the inherent uncertainty in the labels, and discretized into 10 regions of equal width in [0,1], similarly to the examples in Fig. 1.

In-domain prediction. Summary results for an in-domain prediction task (clean test data) are given in Table 2. While

Table 1: Results on multiclass classification task with CIFAR10 dataset. Hyperparameters are cross validated using validation NLL - for the proposed method, we used $\alpha = 1000$ and for LS, we used $\epsilon = 0.01$. For OOD experiments, we used SVHN dataset as test data. Detailed results are given in Tables 7 - 9 in Appendix G.

	In-domain testset			Corrupted testset			OOD testset
Method	NLL$^{\downarrow}$	Accuracy$^{\uparrow}$	ECE$^{\downarrow}$	NLL$^{\downarrow}$	Accuracy$^{\uparrow}$	ECE$^{\downarrow}$	$\Delta_{\text{OOD}}^{\uparrow}$
ELBO	0.76 ± 0.01	$\mathbf{0.82 \pm 0.00}$	0.10 ± 0.00	1.34 ± 0.02	$\mathbf{0.70 \pm 0.00}$	0.18 ± 0.00	0.55 ± 0.08
LS	1.93 ± 0.02	0.79 ± 0.00	0.17 ± 0.00	3.36 ± 0.09	0.67 ± 0.01	0.27 ± 0.01	0.13 ± 0.03
EDL	0.78 ± 0.01	$\mathbf{0.82 \pm 0.00}$	$\mathbf{0.08 \pm 0.00}$	1.31 ± 0.02	0.68 ± 0.00	$\mathbf{0.16 \pm 0.00}$	$\mathbf{0.80 \pm 0.08}$
Proposed	$\mathbf{0.68 \pm 0.01}$	$\mathbf{0.82 \pm 0.00}$	$\mathbf{0.08 \pm 0.00}$	$\mathbf{1.23 \pm 0.02}$	$\mathbf{0.70 \pm 0.00}$	$\mathbf{0.16 \pm 0.00}$	0.54 ± 0.03

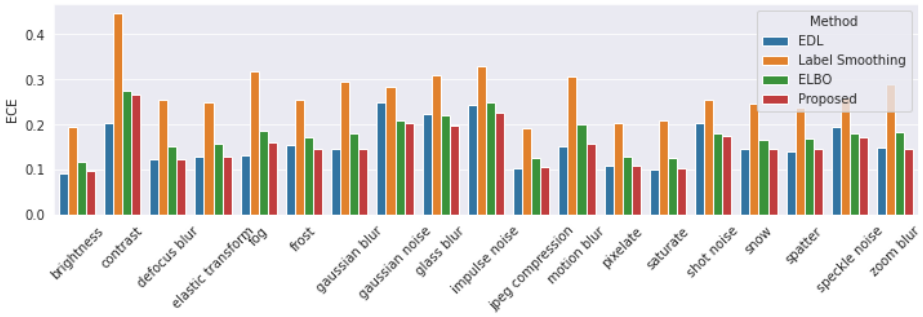


Figure 5: Comparison of ECE on different corruptions. The models are trained on clean CIFAR10 data and tested with various corruptions from the CIFAR-10-C dataset.

Table 2: Comparison of different methods on the sentiment analysis task. All models achieve $> 88\%$ accuracy. Comprehensive comparison of different variants of the proposed method along with accuracy and AUROC metrics is given in Table 4 in appendix. $^{\downarrow}$ means lower the better.

Method	NLL$^{\downarrow}$	ECE$^{\downarrow}$
ELBO	0.341 ± 0.024	0.045 ± 0.009
LS($\epsilon = 0.05$)	0.444 ± 0.029	0.071 ± 0.004
EDL	0.301 ± 0.001	0.044 ± 0.004
Proposed ($\alpha = 1000$)	$\mathbf{0.288 \pm 0.002}$	$\mathbf{0.026 \pm 0.001}$

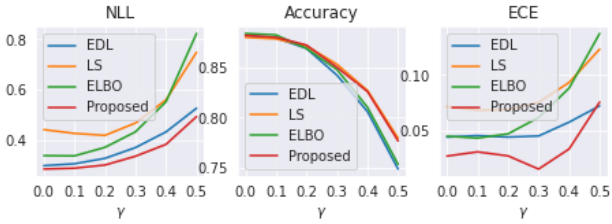


Figure 6: Comparison of different methods in corrupted test embeddings for the sentiment analysis task. γ represents the strength of noise added. A detailed comparison is available in Fig. 10 in appendix.

all methods achieve $> 88\%$ accuracy (Table 4 in Appendix), the proposed method is able to provide significantly better NLL and calibration performance. We explore alternative prior histograms s_0 in Appendix (E) and show that incorporating the information about uncertainty in constituent phrases helps the model to improve both calibration as well as prediction accuracy.

Corrupted test data. To study the robustness of the methods against data corruptions, we perturb the BERT embeddings in test data with variance preserving noise as $\tilde{\mathbf{e}} = (1 - \gamma) * \mathbf{e} + \gamma * \eta$, where $\mathbf{e}$ is the noise free embedding and $\eta \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. The results in Fig. 6 show that the proposed method is robust against corruptions. Even though both ELBO and the proposed method are trained with MFVI, we see that robustness of ELBO deteriorates significantly with added noise while the proposed method better retains its robustness, even compared to EDL.

5.2 Multiclass classification with CIFAR10

Here we consider multiclass classification with CIFAR10 data and balanced classes. For OOD experiments, we use the SVHN dataset (Netzer et al., 2011) and for corruptions, CIFAR-10-C (Hendrycks and Dietterich, 2019).

In-domain prediction. We give the results on in-domain prediction in Table 1. While we observe that none of the considered methods reaches the s-o-t-a accuracy for VGG11, we nevertheless clearly see that training the models with the proposed method can reduce the ECE significantly when compared to the regular BNN trained with ELBO, and closely matches EDL using the same architecture.

Table 3: Results with imbalanced data. Models are trained on Imbalanced CIFAR10. The parameters of each model are selected using the validation NLL. LS is not included due to a very large NLL values with corrupted test data.

	In-domain test data		Corrupted test data	
Method	NLL$^{\downarrow}$	F1 Score$^{\uparrow}$	NLL$^{\downarrow}$	F1 Score$^{\uparrow}$
ELBO	1.158 ± 0.026	$\mathbf{0.849 \pm 0.002}$	6.354 ± 0.117	0.331 ± 0.012
EDL	$\mathbf{0.703 \pm 0.009}$	0.824 ± 0.003	$\mathbf{2.840 \pm 0.034}$	0.314 ± 0.012
Proposed ($\alpha = 500$)	0.960 ± 0.022	$\mathbf{0.847 \pm 0.001}$	3.564 ± 0.070	$\mathbf{0.400 \pm 0.011}$

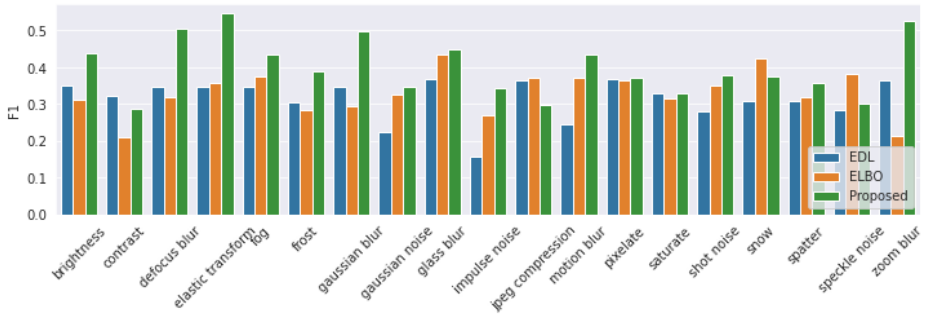


Figure 7: Comparison of F1 on different corruptions on imbalanced dataset. Classifier is trained clean imbalanced CIFAR10 data and tested with various corruptions from CIFAR-10-C dataset.

Corrupted test data. When testing on different corruptions, we can observe in Table 1 and Fig. 5 that the proposed method singificantly improves NLL and ECE compared with the vanilla ELBO, which does not incorporate the prior summary information. This demonstrates how the summary prior helps the model to regularize its predictions. In ECE the proposed method and EDL are jointly the best, whereas in NLL the proposed method is the single best method with corrupted test data.

Detection of out-of-distribution samples. Here we compare in-distribution and OOD predictive entropies. A model that captures uncertainty properly should have a smaller predictive entropy (larger confidence) for in-distribution than for OOD test samples. Here, EDL performs better with a larger Δ_{OOD}, but this comes at the cost of higher in-domain entropy (Table 9 in Appendix). Instead, incorporating prior summary information through S-ELBO appropriately balances between in-domain and OOD prediction confidence.

5.3 CIFAR10 with class imbalance

To test the ability of the method to incorporate prior knowledge about class imbalance, we create a dataset from CIFAR10 by sub-sampling image classes, such that the imbalance ratio is $1 : 1/2 : 1/4 : \ldots : 1/2^8 : 1/2^8$. We assume that the class fractions are available as prior knowledge, and use this to construct $\mathbf{s}_0$ as demonstrated in Fig. 2c.

In-domain prediction. The results of in-domain prediction are given in Table 3. Here EDL has a better NLL, but the proposed method and ELBO yield a much larger F1 score. As the NLL is dominated by the majority class, we conclude that the proposed method does a better job in the challenging task of classifying the minority class items. Note that the summary likelihood regularizes the predicted scores to reflect the class imbalance and hence the proposed model is able to provide both a low NLL (relative to ELBO) and a high F1-score.

Corrupted test data. More useful insights can be obtained when analysing the performance under corrupted test data. Table 3 and Fig. 7 show the F1 scores of different methods for corrupted test datasets. These results show that incorporating the prior knowledge of class imbalance provides a significant improvement in F1 score compared to both ELBO and EDL. This clearly points to the fact that the strategy of the proposed method to allocate probability mass for each class label through the summary likelihood results in robust predictions under noisy input.

Additional experiments. We also provide a comprehensive study of binary classification on a dataset derived from MNIST in Appendix F. We study different dataset sizes and architectures and show that the proposed method is able to provide advantage over the standard ELBO formulation in most cases. As this task is simple, all models perform rather well and there is no single best method.

6 Concluding Remarks

We presented a principled approach to incorporate prior knowledge about the distribution of predicted scores in Bayesian neural network training. Technically, we aug-

mented the data with a summary observation s_0 that captured the prior knowledge. One way to think about the summary is to interpret it as a 'pseudo-observation'; pseudo-observations are often used to interpret common priors (Gelman et al., 2014). In order to incorporate all prior knowledge into the model, we apply Bayes rule, and multiply the weight prior with a likelihood term for the summary statistic. Consequently, this yields a well-defined ELBO corresponding to proper Bayesian inference. Thorough empirical experiments in computer vision and natural language processing showed that the proposed method improved robustness and calibration of the BNNs. While we only considered MFVI training of the BNNs, the summary likelihood can be easily incorporated in other Bayesian training methods like MCMC or even to deterministic networks (See Appendix I).

Acknowledgements

This work was supported by the Academy of Finland (Flagship programme: Finnish Center for Artificial Intelligence FCAI, and grants 336033, 352986) and EU (H2020 grant 101016775 and NextGenerationEU). The authors would also like to thank Çağlar Hızlı and Manuel Haussmann for useful discussions and our reviewers for their insightful comments that helped us to improve our paper.

References

Amini, A., Schwarting, W., Soleimany, A., and Rus, D. (2020). Deep evidential regression. *Advances in Neural Information Processing Systems*, 33:14927–14937.

Andrews, D. F. and Mallows, C. L. (1974). Scale mixtures of normal distributions. *Journal of the Royal Statistical Society: Series B (Methodological)*, 36(1):99–102.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*.

Cui, T., El Mekkaoui, K., Havulinna, A., Marttinen, P., and Kaski, S. (2021a). Improving Neural Networks for Genotype-Phenotype Prediction Using Published Summary Statistics. *bioRxiv*.

Cui, T., Havulinna, A., Marttinen, P., and Kaski, S. (2021b). Informative Bayesian Neural Network Priors for Weak Signals. *Bayesian Analysis*, 1(1):1–31.

Deng, W., Zhang, X., Liang, F., and Lin, G. (2019). An adaptive empirical Bayesian method for sparse deep learning. In *Advances in Neural Information Processing Systems*, pages 5564–5574.

Dusenberry, M. W., Jerfel, G., Wen, Y., Ma, Y.-a., Snoek, J., Heller, K., Lakshminarayanan, B., and Tran, D. (2020). Efficient and scalable Bayesian neural nets with rank-1 factors. *arXiv preprint arXiv:2005.07186*.

Esteva, A., Robicquet, A., Ramsundar, B., Kuleshov, V., DePristo, M., Chou, K., Cui, C., Corrado, G., Thrun, S., and Dean, J. (2019). A guide to deep learning in healthcare. *Nature Medicine*, 25(1):24–29.

Flam-Shepherd, D., Requeima, J., and Duvenaud, D. (2017). Mapping Gaussian process priors to Bayesian neural networks. In *NIPS Bayesian deep learning workshop*.

Fortuin, V. (2021). Priors in Bayesian deep learning: A review. *arXiv preprint arXiv:2105.06868*.

Gal, Y. and Ghahramani, Z. (2016). Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pages 1050–1059.

Gelman, A. (2021). Multiple priors for the same parameter - Stan Forum. `https://discourse.mc-stan.org/t/multiple-priors-for-the-same-parameter/10943/9`. [Online; accessed 11-Oct-2022].

Gelman, A., Carlin, J., Stern, H., Dunson, D., Vehtari, A., and Rubin, D. (2014). Bayesian Data Analysis. vol. 2 CRC press. *Boca Raton, FL*.

George, D. and Huerta, E. A. (2018). Deep Learning for real-time gravitational wave detection and parameter estimation: Results with Advanced LIGO data. *Physics Letters B*, 778:64–70.

Ghosh, S., Yao, J., and Doshi-Velez, F. (2018). Structured variational learning of Bayesian neural networks with horseshoe priors. In *International Conference on Machine Learning*, pages 1739–1748.

Graves, A. (2011). Practical variational inference for neural networks. In *Advances in Neural Information Processing Systems*, pages 2348–2356.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks. In *International Conference on Machine Learning*, pages 1321–1330. PMLR.

Hafner, D., Tran, D., Lillicrap, T., Irpan, A., and Davidson, J. (2018). Noise contrastive priors for functional uncertainty. *arXiv preprint arXiv:1807.09289*.

Hendrycks, D. and Dietterich, T. (2019). Benchmarking neural network robustness to common corruptions and perturbations. *International Conference on Learning Representations*.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in Neural Information Processing Systems*, 30.

Louizos, C., Ullrich, K., and Welling, M. (2017). Bayesian compression for deep learning. In *Advances in Neural Information Processing Systems*, pages 3288–3298.

MacKay, D. J. (1992). A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472.

MacKay, D. J. (1994). Bayesian nonlinear modeling for the prediction competition. *ASHRAE transactions*, 100(2):1053–1062.

Malinin, A. and Gales, M. (2018). Predictive uncertainty estimation via prior networks. *Advances in Neural Information Processing Systems*, 31.

Minderer, M., Djolonga, J., Romijnders, R., Hubis, F., Zhai, X., Houlsby, N., Tran, D., and Lucic, M. (2021). Revisiting the calibration of modern neural networks. *Advances in Neural Information Processing Systems*, 34.

Molchanov, D., Ashukha, A., and Vetrov, D. (2017). Variational dropout sparsifies deep neural networks. In *International Conference on Machine Learning*, pages 2498–2507. JMLR. org.

Mu, N. and Gilmer, J. (2019). MNIST-C: A robustness benchmark for computer vision. *arXiv preprint arXiv:1906.02337*.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT press.

Nalisnick, E., Hernandez-Lobato, J. M., and Smyth, P. (2019). Dropout as a structured shrinkage prior. In *International Conference on Machine Learning*, pages 4712–4722.

Neal, R. M. (1992). Bayesian mixture modeling. In *Maximum Entropy and Bayesian Methods*, pages 197–211. Springer.

Neal, R. M. (2012). *Bayesian Learning for Neural Networks*, volume 118. Springer Science & Business Media.

Neklyudov, K., Molchanov, D., Ashukha, A., and Vetrov, D. P. (2017). Structured Bayesian pruning via log-normal multiplicative noise. In *Advances in Neural Information Processing Systems*, pages 6775–6784.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32.

Reimers, N. and Gurevych, I. (2019). Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Conference on Empirical Methods in Natural Language Processing*. ACL.

Rieger, L., Singh, C., Murdoch, W., and Yu, B. (2020). Interpretations are useful: penalizing explanations to align neural networks with prior knowledge. In *International Conference on Machine Learning*, pages 8116–8126. PMLR.

Ross, A. S., Hughes, M. C., and Doshi-Velez, F. (2017). Right for the right reasons: training differentiable models by constraining their explanations. In *International Joint Conference on Artificial Intelligence*, pages 2662–2670.

Seeger, M. (2004). Gaussian processes for machine learning. *International journal of neural systems*, 14(02):69–106.

Sensoy, M., Kaplan, L., and Kandemir, M. (2018). Evidential deep learning to quantify classification uncertainty. *Advances in Neural Information Processing Systems*, 31.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642. ACL.

Sun, S., Chen, C., and Carin, L. (2017). Learning structured weight uncertainty in Bayesian neural networks. In *Artificial Intelligence and Statistics*, pages 1283–1292.

Sun, S., Zhang, G., Shi, J., and Grosse, R. (2019). Functional variational Bayesian neural networks. *arXiv preprint arXiv:1903.05779*.

Swiatkowski, J., Roth, K., Veeling, B. S., Tran, L., Dillon, J. V., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. (2020). The k-tied normal distribution: A compact parameterization of Gaussian mean field posteriors in Bayesian neural networks. *arXiv preprint arXiv:2002.02655*.

Teh, Y. W. (2010). Dirichlet Process. *Encyclopedia of machine learning*, 1063:280–287.

Teh, Y. W., Jordan, M. I., Beal, M. J., and Blei, D. M. (2006). Hierarchical Dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581.

Tran, B.-H., Rossi, S., Milios, D., and Filippone, M. (2020). All you need is a good functional prior for Bayesian deep learning. *arXiv preprint arXiv:2011.12829*.

Tseng, A., Shrikumar, A., and Kundaje, A. (2020). Fourier-transform-based attribution priors improve the interpretability and stability of deep learning models for genomics. *Advances in Neural Information Processing Systems*, 33.

Wang, D.-B., Feng, L., and Zhang, M.-L. (2021). Rethinking Calibration of Deep Neural Networks: Do Not Be Afraid of Overconfidence. *Advances in Neural Information Processing Systems*, 34.

Weinberger, E., Janizek, J., and Lee, S.-I. (2020). Learning deep attribution priors based on prior knowledge. *Advances in Neural Information Processing Systems*, 33:14034–14045.

Wenzel, F., Roth, K., Veeling, B., Swiatkowski, 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 *International Conference on Machine Learning*, pages 10248–10259. PMLR.

Wilson, A. G. (2020). The case for Bayesian deep learning. *arXiv preprint arXiv:2001.10995*.

Wilson, A. G. and Izmailov, P. (2020). Bayesian deep learning and a probabilistic perspective of generalization. *arXiv preprint arXiv:2002.08791*.

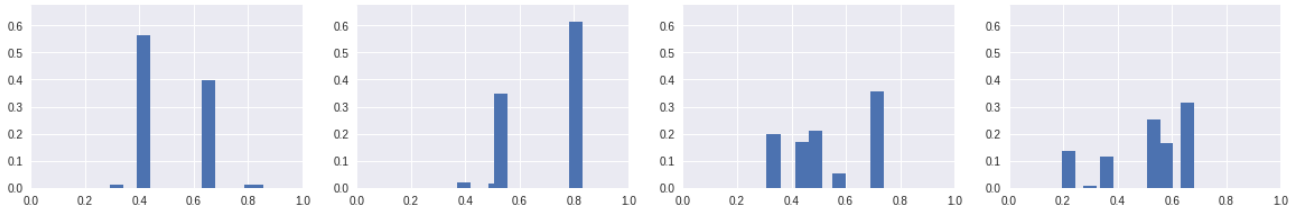
Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv preprint arXiv:1708.07747*.

Yang, W., Lorch, L., Graule, M., Lakkaraju, H., and Doshi-Velez, F. (2020). Incorporating interpretable output constraints in Bayesian neural networks. *Advances in Neural Information Processing Systems*, 33:12721–12731.

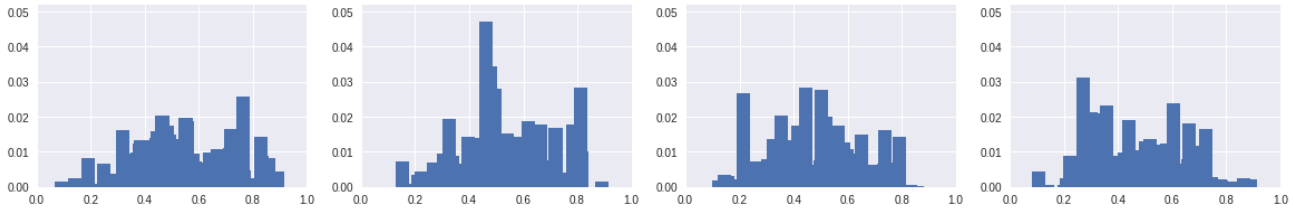
Incorporating functional summary information in Bayesian neural networks using a Dirichlet process likelihood approach

A Effect of concentration parameter α in DP

As introduced in Sec2.2 and later used in Sec 3.2, we use Dirichlet Process to model the summary statistic information $\mathbf{s}_0$. The concentration parameter, α, play a key role in defining the DP and estimating the likelihood loss for each set of summary samples predicted by neural network. In Fig. 8, we show samples drawn from a DP with base measure Beta$(5, 5)$ for different values of α.



(a) Samples from DP with $\alpha = 1$



(b) Samples from DP with $\alpha = 100$

Figure 8: Samples from $\mathcal{DP}(\text{Beta}(5, 5); \alpha)$ for different values of α. The concentration parameter α governs how closely the samples from $\mathcal{DP}$ will be to the base measure. In top row, we give the results of using $\alpha = 1$ with a base measure of Beta$(5, 5)$. The samples are widely different, has high variance between them, and individual samples are not close to the base measure used. When we increase the concentration parameter to $\alpha = 100$, the samples drawn from the DP are close to base measure.

B Derivation of parameters for $\mathbf{s}_0$

In the case of binary classification problem, we assume that the following prior knowledge is available for modeling the base distribution for Dirichlet Process.

1. Fraction of minority class samples. If n_0 is the number of majority class samples in the training set and n_1 is the number of minoty class samples, then the fraction of minority class samples is defined as

$$\gamma_1 = \frac{n_1}{n_0 + n_1} \tag{8}$$

2. Expected accuracy. This refers to the expected accuracy of a trained model and is defined as

$$\mathcal{E}_a = \int_{x=0}^{1/2} (1-x) f(x) dx + \int_{x=1/2}^{1} x f(x) dx, \tag{9}$$

where $f(x)$ is the density function of the predicted scores x.

B.1 Useful results

1. Let $a, b > 0$ and $0 < u < 1$, then

$$\int u^{a-1}(1-u)^{b-1} du = \frac{u^a}{a} {}_2F_1(a, 1-b; a+1; u) + constant, \tag{10}$$

where ${}_2F_1(\cdot)$ is the hypergeometric function.

B.2 Beta distribution as $\mathbf{s}_0$

The probability density function is defined as

$$f(x; a, b) = \frac{1}{B(a,b)} x^{a-1}(1-x)^{b-1}, \tag{11}$$

where $a, b > 0$, $x \in (0, 1)$ and $B(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$ is the Beta function.

The cummulative distribution function is defined as

$$F(x) = \frac{B(x; a, b)}{B(a, b)}, \tag{12}$$

where $B(x; a, b) = \int_0^x t^{a-1}(1-t)^{b-1} dt$ is the incomplete Beta function. We can apply (10) to express this in terms of hypergeometric function.

Mean of Beta distributed random variable is defined as

$$\mu = \frac{a}{a+b}. \tag{13}$$

B.3 Deriving parameters from prior information

Let $\gamma_0 = 1 - \gamma_1$ is the fraction of majority samples in the training dataset.

Assuming $s = \frac{1}{2}$ as the threshold for binary decision making, where s is the score predicted by the model, we can see that

$$\int_{s=0}^{1/2} f(s) ds = \gamma_0. \tag{14}$$

Assuming that score is distributed as Beta distribution, we have

$$\begin{aligned} F(1/2) &= \gamma_0 \\ \frac{B(1/2;a,b)}{B(a,b)} &= \gamma_0 \\ \frac{1}{a2^a}{}_2F_1(a,1-b;a+1;1/2) &= \gamma_0 B(a,b) \end{aligned} \tag{15}$$

Unable to proceed because of lack of closed for expression for ${}_2F_1(\cdot)$

Now, consider the expected accuracy $\mathcal{E}_a$,

$$\begin{aligned} \mathcal{E}_a &= \int_0^{1/2} (1-s)f(s)ds + \int_{s=1/2}^{1} sf(s)ds \\ &= \int_0^{1/2} f(s)ds - \int_0^{1/2} sf(s)ds + \int_{s=1/2}^{1} sf(s)ds \end{aligned} \tag{16}$$

Applying the density function, we have

$$\begin{aligned} \int sf(s)ds &= \int s \cdot \frac{1}{B(a,b)} s^{a-1}(1-s)^{b-1}ds \\ &= \frac{1}{B(a,b)} \int s^a (1-s)^{b-1}ds \\ &= \frac{1}{B(a,b)} \frac{s^{a+1}}{a+1}{}_2F_1(a+1,1-b;a+2;s) \end{aligned} \tag{17}$$

Continuing from (16),

$$\begin{aligned} \mathcal{E}_a &= F(1/2) - \frac{1}{B(a,b)} \left(\left[\frac{s^{a+1}}{a+1}{}_2F_1(a+1,1-b;a+2;x) \right]_{s=0}^{s=1/2} - \left[\frac{s^{a+1}}{a+1}{}_2F_1(a+1,1-b;a+2;x) \right]_{s=1/2}^{s=1} \right) \\ &= \frac{1}{2^a B(a,b)} \left(\frac{1}{a}{}_2F_1(a,1-b;a+1;1/2) \frac{1}{a+1} \left({}_2F_1(a+1,1-b;a+2;1/2) - {}_2F_1(a+1,1-b;a+2;1) \right) \right) \end{aligned} \tag{18}$$

Ideally, from (15) and (18), we can find the value of a and b given γ_0 and $\mathcal{E}_a$. However, closed form expression for solutions are not available. But, using optimization techniques, we can find the approximate solution for this problem. For our experiments, the objective function for optimization is chosen as the MSE between the target value $(\gamma_0, \mathcal{E}_a)$ and the value $(\hat{\gamma}_0, \hat{\mathcal{E}_a})$ observed for the pair $(\hat{a}, \hat{b})$. We used 'L-BFS-G' optimizer available with sklearn package for solving the optimization problem with $\mathcal{E}_a$ set to 0.95 - 0.98.

In multiclass classification experiments, this approach becomes a difficult problem to solve with more unknown that known variables. In those cases, we chose the base distribution parameters by considering the Dirichlet distribution with parameters proportional to number of samples and dividing this into unequal regions to build the base Dirichlet distribution for DP.

C SoftHistogram construction during training

The proposed approach requires to estimate the distribution of predicted scorea over a mini-batch of sample to train the model using summary information. We resort to a differentiable histogram for this.

Let $\mathcal{B}$ denotes the regions which constite the prediction simplex and over which we are intereseted in computing the soft histogram. We identity each region by a center c_i and width δ_i and ensuring that no two regions overlap. For a predicted score vector $\tilde{y}$, the contribution of it towards each region i is computed using

$$g_i(\tilde{y}) = \text{sigmoid}(\sigma * (\tilde{y} - b_i + \delta_i/2)) - \text{sigmoid}(\sigma * (\tilde{y} - b_i - \delta_i/2)). \tag{19}$$

Here, σ acts as the slope of the sigmoid function and effectively improves the quality of histogram estimation. In our experiments, we used $\sigma = 500$.

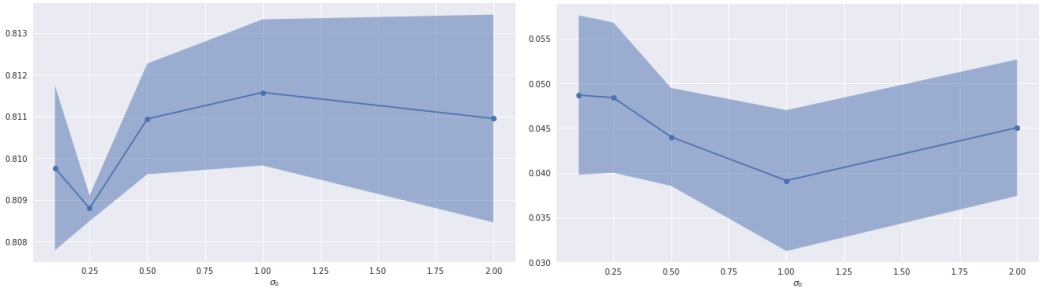
Finally, over a minibatch $\mathcal{B}$, the weight for each region i is computed as

$$w_i = \sum_{\tilde{y}=f(\mathbf{x});\mathbf{x}\in\mathcal{B}} g_i(\tilde{y}). \tag{20}$$

Other methods such as KDE-style estimation with Gaussian kernal can also be used. But in our experiments, we found that using the above method provided stable training.

D Cross validation of prior variance

In this section, we provide results for cross validating the choice of hyperparameters we chose to design the Bayesian Neural Network. The results of crossvalidating prior variance σ_0 is given in Fig. 9. Both accuracy and ECE is found to be better at $\sigma_0 = 1$ and we continue to use this value for all our experiments.



(a) Accuracy vs σ_0 (b) ECE vs σ_0

Figure 9: Results of cross validating prior variance, σ_0, of neural network parameters. The result is for CIFAR10 with VGG11 architecture.

E Additional results for Sentiment Analysis task

In this section, we provide additional results for the sentiment analysis task. In Table 4, we provide the in-domain test results and in Fig. 10 we provide the effect of corruption in embeddings.

In the labels below, for proposed method, 'auto' means base distribution parameters are computed based on the method in Sec. B, 'uniform' refers to uniform base distribution, 'eqbin' refer to equal bin width strategy and 'uneqbin' refers to unequal bin width strategy discussed in Sec. 5.1 in main text.

Table 4: Results on test dataset. Models are trained on SST. The parameters for each model are chosen based on best validation NLL.

Method	NLL$^{\downarrow}$	Accuracy$^{\uparrow}$	AUROC$^{\uparrow}$	ECE$^{\downarrow}$
ELBO	0.341 ± 0.024	0.883 ± 0.004	0.952 ± 0.002	0.045 ± 0.009
LS	0.444 ± 0.029	0.880 ± 0.002	0.950 ± 0.001	0.071 ± 0.004
EDL	0.301 ± 0.001	0.882 ± 0.001	0.954 ± 0.000	0.044 ± 0.004
Proposed (auto, eqbin, $\alpha = 10^3$)	0.297 ± 0.010	0.883 ± 0.001	0.953 ± 0.001	0.034 ± 0.004
Proposed (uniform, eqbin, $\alpha = 10^3$)	$\mathbf{0.288 \pm 0.002}$	0.881 ± 0.002	0.953 ± 0.000	0.026 ± 0.001
Proposed (auto, uneqbin, $\alpha = 10^2$)	0.302 ± 0.009	$\mathbf{0.886 \pm 0.002}$	$\mathbf{0.954 \pm 0.001}$	0.033 ± 0.005
Proposed (uniform, uneqbin, $\alpha = 500$)	0.291 ± 0.005	0.883 ± 0.003	0.951 ± 0.002	$\mathbf{0.021 \pm 0.005}$

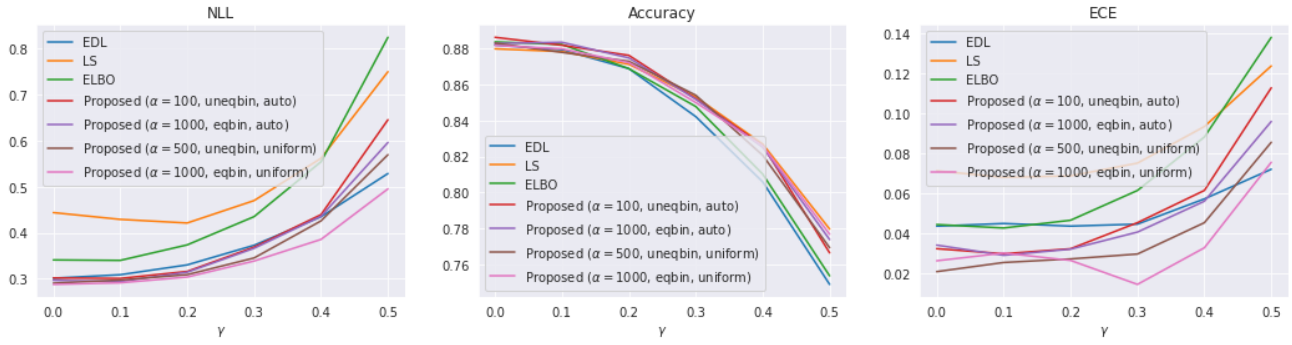


Figure 10: Effect of corruption in sentiment analysis task

F Results for BinaryMNIST classification task

A binary classification task is constructed from MNIST dataset by sampling only two labels, '3' and '5'. Popular models, LeNet and ConvNet are used as the Bayesian neural network architectures. To study the effect of dataset size, we performed experiments with different sizes - $|\mathcal{D}| = 1000$ and $|\mathcal{D}| = 8000$. We use a summary observation $\mathbf{s}_0$ for the proposed method is constructed using the method described in Appendix B. Unequal width regions are used to construct the base distribution using $\{0.01, 0.05, 0.10, 0.90, 0.95, 0.99\}$ as boundaries. This unequal width regions help the model to concentrate more of high confidence predictions to match the base distribution while giving less importance to low confidence predictions. The results are provided in Appendix F and show that the proposed method is able to provide improved performance in big architectures and large dataset regime.

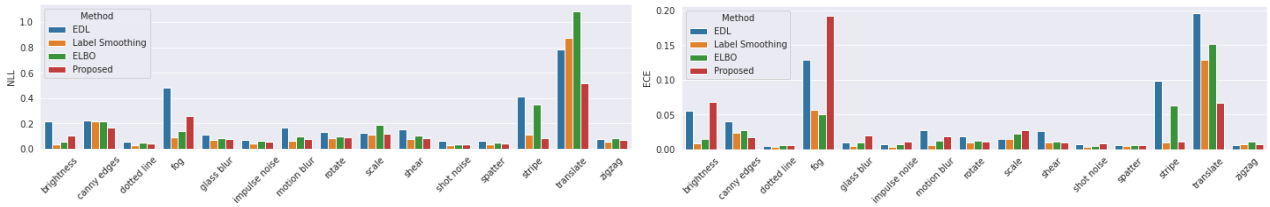
Table 5: Results on clean dataset. Models are trained on MNIST for binary classification. The parameters for each model are chosen based on best validation NLL.

	$\lvert\mathcal{D}\rvert$	Method	NLL$^{\downarrow}$	Accuracy$^{\uparrow}$	AUROC$^{\uparrow}$	ECE$^{\downarrow}$
LeNet	1000	ELBO	0.013 ± 0.001	0.995 ± 0.001	1.000 ± 0.000	$\mathbf{0.002 \pm 0.001}$
		LS	$\mathbf{0.012 \pm 0.001}$	$\mathbf{0.995 \pm 0.000}$	$\mathbf{1.000 \pm 0.000}$	0.003 ± 0.000
		EDL	0.035 ± 0.002	0.992 ± 0.001	1.000 ± 0.000	0.010 ± 0.002
		Proposed ($\alpha = 50.0$)	0.014 ± 0.001	0.994 ± 0.001	1.000 ± 0.000	0.003 ± 0.000
	8000	ELBO	0.028 ± 0.003	0.991 ± 0.000	1.000 ± 0.000	0.004 ± 0.000
		LS	$\mathbf{0.019 \pm 0.001}$	$\mathbf{0.992 \pm 0.001}$	$\mathbf{1.000 \pm 0.000}$	$\mathbf{0.003 \pm 0.000}$
		EDL	0.045 ± 0.003	0.989 ± 0.001	1.000 ± 0.000	0.006 ± 0.001
		Proposed ($\alpha = 100.0$)	0.026 ± 0.002	0.992 ± 0.001	1.000 ± 0.000	0.005 ± 0.001
ConvNet	1000	ELBO	0.039 ± 0.005	0.993 ± 0.001	1.000 ± 0.000	0.024 ± 0.004
		LS	0.040 ± 0.006	0.993 ± 0.001	$\mathbf{1.000 \pm 0.000}$	0.024 ± 0.005
		EDL	$\mathbf{0.030 \pm 0.004}$	$\mathbf{0.994 \pm 0.000}$	1.000 ± 0.000	$\mathbf{0.012 \pm 0.004}$
		Proposed ($\alpha = 10.0$)	0.063 ± 0.003	0.990 ± 0.001	1.000 ± 0.000	0.041 ± 0.004
	8000	ELBO	0.043 ± 0.005	0.994 ± 0.001	1.000 ± 0.000	0.032 ± 0.005
		LS	0.058 ± 0.018	0.996 ± 0.000	1.000 ± 0.000	0.046 ± 0.016
		EDL	0.091 ± 0.055	$\mathbf{0.996 \pm 0.000}$	1.000 ± 0.000	0.061 ± 0.039
		Proposed ($\alpha = 50.0$)	$\mathbf{0.029 \pm 0.002}$	0.995 ± 0.001	$\mathbf{1.000 \pm 0.000}$	$\mathbf{0.020 \pm 0.003}$

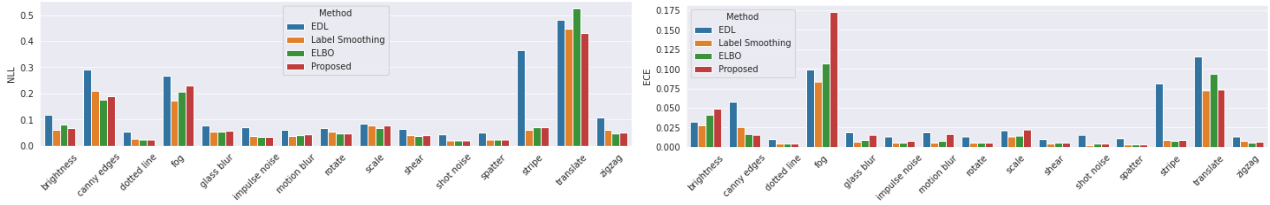
In-domain prediction. In Table 5, we study the performance in in-domain dataset and OOD datasets. Since MNIST is a well curated dataset, there is very less chance of label confusion (as opposite to the sentiment analysis task above) and hence the base distribution we used also encourages the predictions to be concentrated towards high confidence regions, similar to MFVI and LS. However, the proposed method also encourages the distribution of predicted scores to have a non zero mass in low confidence regions and this clearly seems to improve NLL and ECE in clean dataset. We can attribute this to the fact that proposed method makes the models predict less confident scores for incorrect samples and hence lowering the cost of mistake (NLL) and on the course, improving calibration (ECE).

Corrupted test data. An important aspect of the proposed method is that it can better represent the uncertainty about predictions by regularizing the posterior score distribution. To test this claim, we provide results on corrupted dataset. We use the MNIST-C (Mu and Gilmer, 2019) dataset which has 15 corruptions, and we measure the calibration error on the corrupted data. The results in Fig. 11 show that in most cases the proposed method is able to perform better than baseline methods.

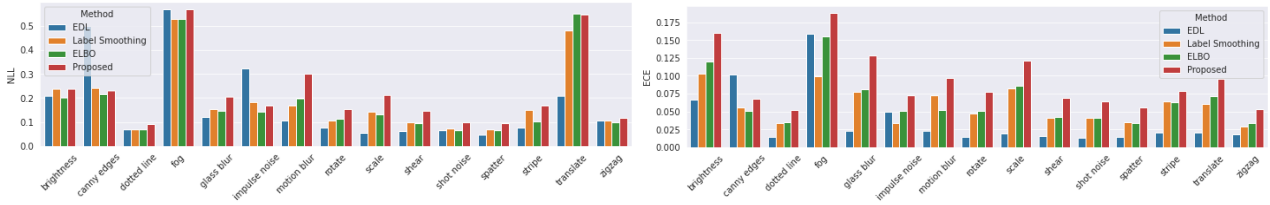
Detection of out-of-distribution samples. For OOD experiment, we used FashionMNIST (Xiao et al., 2017) as the test dataset and predictive entropy is measured as a proxy for prediction uncertainty. Δ_{OOD} is defined as the difference between OOD and in-domain predictive entropies and higher Δ_{OOD} is desirable. From Table 6, we can see that the proposed method is able to provide high OOD predictive entropy while maintaining low in-domain predictive entropy.



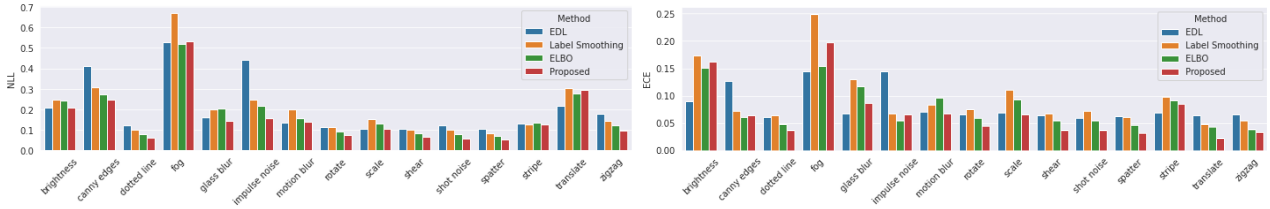
(a) LeNet, $|\mathcal{D}| = 1000$.



(b) LeNet, $|\mathcal{D}| = 8000$.



(c) ConvNet, $|\mathcal{D}| = 1000$.



(d) ConvNet, $|\mathcal{D}| = 8000$.

Figure 11: Comparison of NLL and ECE on different MNIST corruptions. Binary classifier is trained clean MNIST data and tested with various corruptions from MNISTC dataset.

Table 6: Predictive entropy results on OOD setting. Models are trained on clean MNIST dataset and tested on FashionMNIST. The parameters for each model is chosen based on best validation NLL. Δ denotes the difference between OOD predictive entropy and in-domain predictive entropy.

	$\lvert\mathcal{D}\rvert$	Method	In-domain$^{\downarrow}$	OOD$^{\uparrow}$	$\Delta^{\uparrow}$
LeNet	1000	ELBO	0.018 ± 0.001	0.317 ± 0.023	0.300 ± 0.022
		LS	0.016 ± 0.001	0.362 ± 0.011	0.346 ± 0.011
		EDL	0.065 ± 0.004	0.191 ± 0.010	0.126 ± 0.009
		Proposed ($\alpha = 50.0$)	0.033 ± 0.001	0.488 ± 0.022	$\mathbf{0.456 \pm 0.023}$
	8000	ELBO	0.013 ± 0.001	0.433 ± 0.025	0.421 ± 0.024
		LS	0.011 ± 0.001	0.405 ± 0.017	0.394 ± 0.016
		EDL	0.075 ± 0.006	0.276 ± 0.028	0.201 ± 0.024
		Proposed ($\alpha = 100.0$)	0.017 ± 0.001	0.486 ± 0.011	$\mathbf{0.469 \pm 0.011}$
ConvNet	1000	ELBO	0.098 ± 0.015	0.523 ± 0.015	0.426 ± 0.009
		LS	0.100 ± 0.020	0.544 ± 0.021	0.444 ± 0.012
		EDL	0.074 ± 0.016	0.353 ± 0.028	0.279 ± 0.031
		Proposed ($\alpha = 10.0$)	0.151 ± 0.012	0.598 ± 0.012	$\mathbf{0.446 \pm 0.012}$
	8000	ELBO	0.115 ± 0.013	0.602 ± 0.013	0.487 ± 0.010
		LS	0.156 ± 0.043	0.568 ± 0.021	0.412 ± 0.036
		EDL	0.143 ± 0.047	0.384 ± 0.056	0.241 ± 0.018
		Proposed ($\alpha = 50.0$)	0.081 ± 0.009	0.568 ± 0.011	$\mathbf{0.487 \pm 0.019}$

G Additional results for CIFAR10 multiclass classification task

Table 7: Results on clean test dataset. Models are trained on CIFAR10. The parameters for each model are chosen based on best validation NLL.

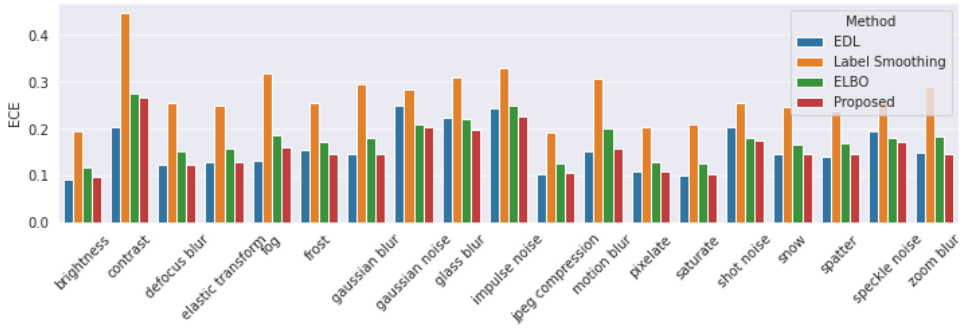
Method	NLL$^{\downarrow}$	Accuracy$^{\uparrow}$	AUROC$^{\uparrow}$	ECE$^{\downarrow}$
ELBO	0.762 ± 0.011	0.820 ± 0.001	0.978 ± 0.000	0.102 ± 0.002
LS ($\epsilon = 0.01$)	1.929 ± 0.019	0.787 ± 0.003	0.961 ± 0.001	0.169 ± 0.003
EDL	0.779 ± 0.006	0.815 ± 0.001	0.959 ± 0.001	$\mathbf{0.078 \pm 0.002}$
Proposed ($\alpha = 1000$)	$\mathbf{0.681 \pm 0.003}$	$\mathbf{0.820 \pm 0.001}$	$\mathbf{0.979 \pm 0.000}$	0.082 ± 0.002

Table 8: Results on corrupted test dataset. Models are trained on CIFAR10. The parameters for each model are chosen based on best validation NLL on testset. CIFAR10-C is used as corrupted dataset. Results are averaged over all 19 corruptions and 5 severity levels.

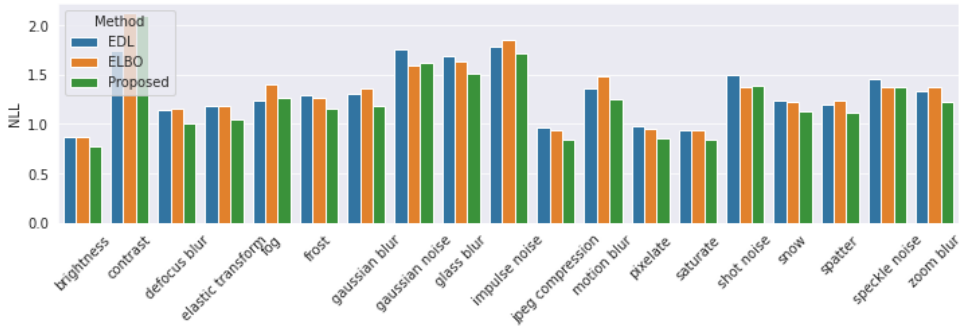
Method	NLL$^{\downarrow}$	Accuracy$^{\uparrow}$	AUROC$^{\uparrow}$	ECE$^{\downarrow}$
ELBO	1.336 ± 0.022	$\mathbf{0.700 \pm 0.004}$	$\mathbf{0.943 \pm 0.002}$	0.177 ± 0.003
LS ($\epsilon = 0.01$)	3.361 ± 0.087	0.666 ± 0.006	0.913 ± 0.003	0.270 ± 0.006
EDL	1.313 ± 0.020	0.677 ± 0.005	0.893 ± 0.003	$\mathbf{0.157 \pm 0.003}$
Proposed ($\alpha = 1000$)	$\mathbf{1.232 \pm 0.022}$	$\mathbf{0.701 \pm 0.004}$	$\mathbf{0.945 \pm 0.002}$	$\mathbf{0.155 \pm 0.003}$

Table 9: Predictive entropy results on OOD setting. Models are trained on clean CIFAR10 dataset and tested on SVHN. The parameters for each model is chosen based on best validation NLL. Δ denotes the difference between ood predictive entropy and indomain predictive entropy.

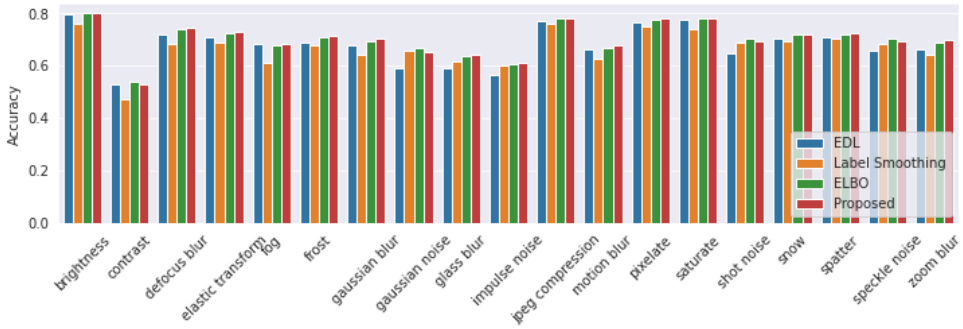
Method	In-domain$^{\downarrow}$	OOD$^{\uparrow}$	$\Delta^{\uparrow}$
ELBO	0.219 ± 0.017	0.771 ± 0.089	0.552 ± 0.077
LS ($\epsilon = 0.01$)	0.106 ± 0.003	0.240 ± 0.029	0.134 ± 0.031
EDL	0.481 ± 0.005	1.282 ± 0.084	$\mathbf{0.801 \pm 0.083}$
Proposed ($\alpha = 1000$)	0.268 ± 0.013	0.806 ± 0.035	0.537 ± 0.027



(a) Expected Calibration Error



(b) Negative Log Likelihood



(c) Accuracy

Figure 12: Comparison of performance on corrupt dataset from CIFAR10-C.

H Additional results for imbalanced classification problem

Table 10: Results on clean test dataset. Models are trained on Imbalanced CIFAR10. The parameters for each model are chosen based on best validation NLL.

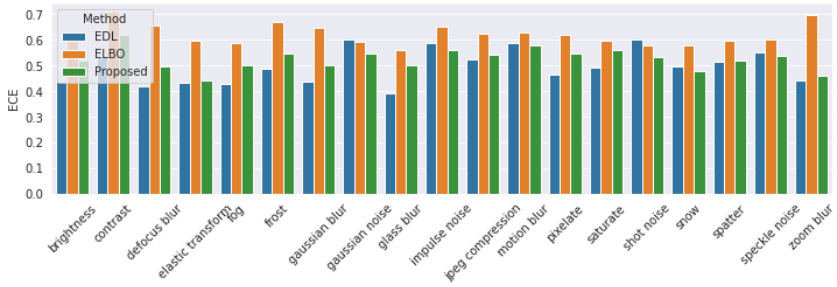
Method	NLL$^{\downarrow}$	F1 Score$^{\uparrow}$	AUROC$^{\uparrow}$	ECE$^{\downarrow}$
ELBO	1.158 ± 0.026	$\mathbf{0.849 \pm 0.002}$	0.901 ± 0.003	0.116 ± 0.002
EDL	$\mathbf{0.703 \pm 0.009}$	0.824 ± 0.003	0.862 ± 0.002	$\mathbf{0.058 \pm 0.002}$
Proposed ($\alpha = 500$)	0.960 ± 0.022	$\mathbf{0.847 \pm 0.001}$	$\mathbf{0.908 \pm 0.001}$	0.101 ± 0.003

Table 11: Results on corrupted test dataset. Models are trained on Imbalanced CIFAR10. The parameters for each model are chosen based on best validation NLL on test data.

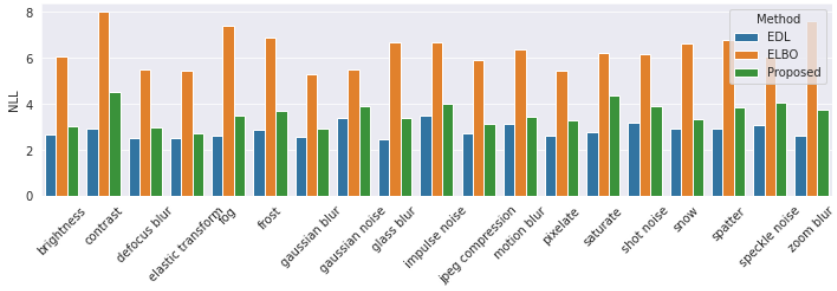
Method	NLL$^{\downarrow}$	F1 Score$^{\uparrow}$	AUROC$^{\uparrow}$	ECE$^{\downarrow}$
ELBO	6.354 ± 0.117	0.331 ± 0.012	0.421 ± 0.002	0.620 ± 0.010
EDL	$\mathbf{2.840 \pm 0.034}$	0.314 ± 0.012	0.355 ± 0.002	$\mathbf{0.495 \pm 0.007}$
Proposed ($\alpha = 500$)	3.564 ± 0.070	$\mathbf{0.400 \pm 0.011}$	$\mathbf{0.421 \pm 0.003}$	0.525 ± 0.008

Table 12: Predictive entropy results on OOD setting. Models are trained on clean imbanalced CIFAR10 dataset and tested on SVHN. The parameters for each model are chosen based on best validation NLL. Δ denotes the difference between OOD predictive entropy and indomain predictive entropy.

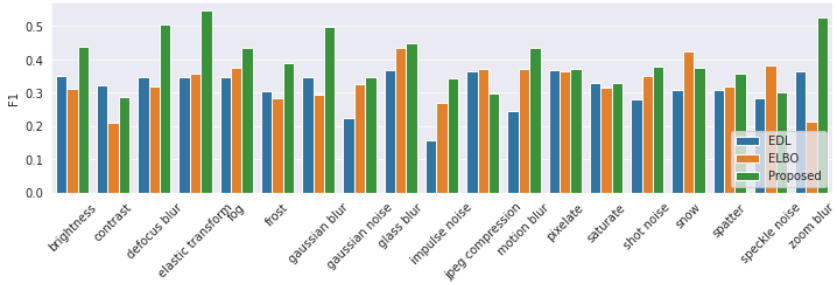
Method	In-domain$^{\downarrow}$	OOD$^{\uparrow}$	$\Delta^{\uparrow}$
ELBO	0.503 ± 0.020	1.461 ± 0.105	$\mathbf{0.959 \pm 0.106}$
EDL	0.099 ± 0.006	0.313 ± 0.028	0.213 ± 0.025
Proposed ($\alpha = 500$)	0.152 ± 0.028	0.517 ± 0.075	0.365 ± 0.051



(a) Expected Calibration Error



(b) Negative Log Likelihood



(c) F1 Score

Figure 13: Comparison of performance on corrupt dataset from Imbalanced CIFAR10-C.

I Training deterministic neural networks with Summary Likelihood

Even though in the main section, we focussed on incorporating prior knowledge to training Bayesian Neural Networks, our goal is to introduce a novel *model* to incorporate informative prior information into NNs that is widely applicable without restrictions on the *inference method* (we used standard VI in main section). To demonstrate this flexibility, we provide additional results of using *SGD + momentum* for NN and NN+SL in Tables 13 and 14 (BNNs in main section were trained 5000 steps, no DA). We observe that data augmentation can improve accuracy but importantly also that the SL consistently improves the accuracy, calibration error and OOD detection. We believe SL can be beneficial with other better inference methods of NNs, both Bayesian and non-Bayesian, such as the deep ensembles.

Table 13: Results on VGG11 trained on CIFAR10. Evaluation is performed on clean dataset. Numbers on braces indicates the number of training steps. SL - Proposed approach. DA - Data augmentation.

Method	NLL$^{\downarrow}$	Accuracy$^{\uparrow}$	AUROC$^{\uparrow}$	ECE$^{\downarrow}$
NN ($5k$)	1.006 ± 0.003	$\mathbf{0.808 \pm 0.001}$	$\mathbf{0.977 \pm 0.000}$	0.138 ± 0.001
NN + SL ($5k$)	$\mathbf{0.803 \pm 0.001}$	0.799 ± 0.001	0.976 ± 0.000	$\mathbf{0.121 \pm 0.000}$
NN ($30k$)	0.948 ± 0.003	0.810 ± 0.001	0.977 ± 0.000	0.133 ± 0.001
NN + SL ($30k$)	$\mathbf{0.772 \pm 0.005}$	$\mathbf{0.817 \pm 0.000}$	$\mathbf{0.980 \pm 0.000}$	$\mathbf{0.115 \pm 0.001}$
NN + DA ($5k$)	$\mathbf{0.498 \pm 0.004}$	$\mathbf{0.843 \pm 0.001}$	$\mathbf{0.986 \pm 0.000}$	0.054 ± 0.001
NN + SL + DA ($5k$)	0.501 ± 0.005	0.834 ± 0.002	0.985 ± 0.000	$\mathbf{0.041 \pm 0.001}$
NN + DA ($30k$)	0.566 ± 0.004	$\mathbf{0.887 \pm 0.001}$	0.991 ± 0.000	0.080 ± 0.001
NN + SL + DA ($30k$)	$\mathbf{0.449 \pm 0.002}$	0.886 ± 0.001	$\mathbf{0.992 \pm 0.000}$	$\mathbf{0.067 \pm 0.000}$

Table 14: Predictive entropy results on OOD setting. VGG11 model is trained on clean CIFAR10 dataset and tested on SVHN. Δ denotes the difference between OOD predictive entropy and indomain predictive entropy.

Method	In-domain$^{\downarrow}$	OOD$^{\uparrow}$	$\Delta^{\uparrow}$
NN ($5k$)	0.140 ± 0.001	0.381 ± 0.010	0.241 ± 0.010
NN + SL ($5k$)	0.222 ± 0.004	0.585 ± 0.045	$\mathbf{0.363 \pm 0.044}$
NN ($30k$)	0.150 ± 0.000	0.384 ± 0.011	0.234 ± 0.012
NN + SL ($30k$)	0.186 ± 0.004	0.475 ± 0.021	$\mathbf{0.290 \pm 0.020}$
NN + DA ($5k$)	0.315 ± 0.005	1.010 ± 0.038	$\mathbf{0.695 \pm 0.036}$
NN + SL + DA ($5k$)	0.377 ± 0.008	1.065 ± 0.025	0.689 ± 0.029
NN + SL ($30k$)	0.093 ± 0.001	0.365 ± 0.005	0.272 ± 0.005
NN + SL + DA ($30k$)	0.135 ± 0.001	0.531 ± 0.026	$\mathbf{0.395 \pm 0.026}$