UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE MÁSTER

Técnicas Expresivas de Calibración para Clasificadores Multiclase

Máster Universitario en Investigación e Innovación en Inteligencia Computacional y Sistemas Interactivos

Autor: Sergio Alvarez Balanya

Tutor: Daniel Ramos Castro
Tutor: Juan Maroñas Molano

Junio 2020

Trabajo de Fin de Máster

TÉCNICAS EXPRESIVAS DE CALIBRACIÓN PARA CLASIFICADORES MULTICLASE

AUTOR: Sergio Alvarez Balanya

DIRECTOR: Daniel Ramos Castro
DIRECTOR: Juan Maroñas Molano

AUDIAS
Dpto. de Tecnología electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio 2020

Abstract

Abstract

In the last years Deep Neural Networks (DNNs) have gained popularity as classification systems and are now applied to multiple tasks where they obtain state-of-the-art accuracy. Some of this tasks benefit from confidence scores associated to predictions. To this aim, a probabilistic classifier outputs confidence scores which are identified with the categorical distribution over the set of possible classes. In this context, calibration is identified with the validity of confidence scores as true probabilities.

Although DNNs report state-of-the-art accuracy in a plethora of tasks, it has been pointed out recently that they present poor calibration. This has motivated a growing body of literature on the calibration of DNNs. One common approach, and the taken in this work, is that of training a separate model to map uncalibrated outputs to better calibrated predictions, namely re-calibration.

We extend the work in [1] where the authors propose to use Bayesian Neural Networks (BNNs) as a decoupled stage from the main training of a DNN. The Bayesian approach is appealing in the sense that it presents proper uncertainty modeling. However, recent work [2] argues that current approximations to the standard Bayesian approach fail to recognize fundamental assumptions of the Bayesian paradigm. We apply the novel framework of Generalized Variational Inference (GVI) to approximate the posterior of the decoupled BNNs. In particular, we substitute the traditional KL divergence used in the standard Variational Inference method for a robust divergence. We provide a comprehensive comparison between both divergences. Moreover, since there is no much literature on the use of robust divergences along GVI, we also contribute with a sensitivity study in which we analyze the influence of the divergence parameters.

Key Words

Deep Neural Networks, Bayesian Neural Networks, Calibration, Uncertainty, Variational Inference, Generalized Variational Inference, Robust Divergences.

Resumen

Resumen

En los últimos años las Redes Neuronales han ganado mucha popularidad como sistemas de clasificación aplicándose en multitud de ámbitos donde consiguen unas medidas de precisión muy buenas. Algunas tareas pueden beneficiarse de clasificadores probabilísticos que reportan valores de confianza asociados a sus predicciones. En este contexto, se entiende la calibración de un clasificador como la validez de sus valores de confianza como verdaderas probabilidades de cada clase.

A pesar de lo precisas que son las Redes Neuronales, trabajo reciente ha señalado que presentan mala calibración. Esto ha motivado un creciente interés por parte de la comunidad incrementando el número de publicaciones. Un enfoque típico, y el tomado en este trabajo, es el de utilizar algoritmos a posteriori para transformar la salida de clasificadores a predicciones con mejor calibración, este método también se conoce como re-calibración.

Este trabajo extiende el realizado en [1], donde los autores proponen usar Redes Neuronales Bayesianas como paso de post-procesado en el que calibrar una Red Neuronal Profunda. El enfoque Bayesiano resulta adecuado ya que implica un buen modelado de la incertidumbre. Sin embargo, trabajo reciente [2] argumenta que las aproximaciones actuales al enfoque Bayesiano ignoran ciertas suposiciones fundamentales del paradigma Bayesiano. En este trabajo se aplica el reciente método de Inferencia Variacional Generalizada para aproximar la Red Neuronal Bayesiana utilizada para re-calibración. En concreto, se sustituye la tradicional divergencia KL en el método de Inferencia Variacional por una divergencia robusta. Se ha realizado una comparativa entre la aplicación de las dos divergencias. Además, se incluye un análisis de sensibilidad en el que se analiza la influencia de los parámetros de la divergencia robusta.

Palabras Clave

Redes Neuronales, Redes Neuronales Bayesianas, Calibración, Incertidumbre, Inferencia Variacional, Inferencia Variacional Generalizada, Divergencias Robustas.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

There is a growing interest in the application of machine learning algorithms to multiple tasks. Deep Neural Networks (DNNs) achieve state-of-the-art performance in a wide variety of applications including Image Recognition [3], Image Captioning [4], and Speech Recognition [5] among others.

Decision making is one task prone to be automated by machine learning algorithms. In this scenario it is desirable to account for probabilistic classifiers that report confidence scores. DNNs meet this condition, but it has been recently pointed out that, although modern DNNs attain high accuracy performance, they present poor calibration [6]. Calibration can be identified with the validity of confidence predictions as the true probability of the classes. Calibration is, then, a very desirable property. If predictions can be taken as actual probabilities, then the Bayesian decision framework can be applied. This allows to minimize the expected cost of a decision problem and to combine predictions with other sources of information. Moreover, meaningful confidence estimates are of great value for auxiliary systems where the final decisions are taken by a human expert like the diagnosis of a possible melanoma or grating a loan.

The concern with obtaining reliable, i.e. well calibrated, probabilistic classifiers is not new [7, 8]. But the recent success of DNNs on other areas has motivated a growing body of literature on the calibration of DNNs [1, 9, 10, 11]. Some of these methods tackle related properties like robustness to outliers and distribution shift; hence, improving calibration implicitly. Others, target specifically calibration. Among these, the most popular approach is that of *re-calibration* in which a calibration method learns a transformation that maps outputs of some uncalibrated classifier to better calibrated predictions. Out method belongs to this latter category.

The main issue spoiling the calibration performance of modern DNNs is that they generate over-confident predictions. They may present high accuracy, but when they fail the confidence values are completely out of place, assigning most of the probability to a wrong class. This means that DNNs

consistently underestimate uncertainty. One way of introducing proper uncertainty modeling is taking a Bayesian approach, but a proper Bayesian modeling of DNNs is intractable and requires the use of approximate methods like Variational Inference. In addition, Bayesian Neural Networks often perform poorly when compared to their non-Bayesian counterparts and rapidly turn unpractical as the complexity of the task scales [12]. Moreover, recent work [2] argues that the Bayesian approximations taken to complex models like Neural Networks and Gaussian Process violate fundamental assumptions of the standard Bayesian paradigm and that this is the cause of many of the problems that arise in real life scenarios.

In this work we try to reconcile the good discrimination properties of DNNs, i.e. high accuracy, with the appealing uncertainty modeling of the Bayesian approach. We do this by extending the work in [1]. There, authors propose to use Decoupled Bayesian Neural Networks to calibrate DNNs. This way the bulk of the task is taken by the DNN which obtains high accuracy in complex tasks like Image Recognition. The BNN is left, then, with the uncertainty estimation of predictions, which can be addressed as a different and simpler task. We extend this work by incorporating the Generalized Variational Inference framework presented in [2] in order to address the concerns with Bayesian approximations observed above. Specifically, we substitute the traditional Kullback–Leibler divergence used in Variational Inference for a robust divergence, the Rényi's α-divergence.

We show that the novel framework of Generalized Variational Inference can be applied to the task of training BNNs for calibration with promising results. Our method provides a flexible way of incorporating uncertainty in a post-processing step and improves previous results by reducing sensitivity to hyperparameters.

Structure

The rest of the work is organized as follows. First we introduce some useful background knowledge, in Chapter 2 we briefly review Neural Networks, give an intuitive description of calibration, and summarize related work. Then, in Chapter 3 we introduce the novel framework of Generalized Variational Inference highlighting central aspects to our work. With the bases covered, we present our method in Chapter 4 and show experiments in Chapter 5. Finally, conclusions and future work are drawn in Chapter 6.

Chapter 2

State of the Art

2.1 Supervised Learning

Supervised Learning can be seen as the problem of learning input to output relations from some empirical data $\mathcal{D}$. Observed data consists on a collection of N samples, each of which is a tuple (input, target), $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{i=1}^{N}$, assumed to be i.i.d. Notice that both, $\mathbf{x}_n$ and $\mathbf{y}_n$, may be vectors of any dimension and not limited to scalars. The true function from $X = \{\mathbf{x}_n\}_{n=1}^{N}$ to $Y = \{\mathbf{y}_n\}_{n=1}^{N}$, which may not even exist, is approximated by a pattern recognition model. The choice of model specifies a family of functions $\mathcal{F}$ over which its possible to optimize some cost function $\mathcal{L}$ w.r.t. the empirical data:

$$f^* = \arg\min_{f \in \mathcal{F}} \mathcal{L}\big(Y, f(X)\big) \tag{2.1}$$

This process is called fitting or training the model. The trained model can be use to make predictions $\hat{\mathbf{y}} = f^*(\mathbf{x})$. Ideally we would like it to perform well on new unseen data $(\mathbf{x}_*, \mathbf{y}_*) \notin \mathcal{D}$, so that model predictions closely resemble the targets $\hat{\mathbf{y}}_* \simeq \mathbf{y}_*$. The ability of a model to meet this condition is called generalization. In practice the observed data comprises only a tiny fraction of all possible inputs, making generalization the main objective in many practical scenarios. From now on and for the sake of simplicity we will drop the $*$ from the notation of the trained model and refer to this just as "the model".

Regarding the form of the target Y, we can distinguish between regression problems and classification problems. In the regression setting the target consists of continuous variables. Whereas in classification the target is one of a set of possible categories. In this work we focus on the classification task.

2.1.1 The Classification Setting

In particular we refer to probabilistic classification problems, where predictions take the form of a probability vector over the possible classes:

$$\hat{y_i} = p(\mathcal{C}_i|\mathbf{x}),$$

where $\hat{y_i}$ index the i-th dimension of the prediction vector $\hat{\mathbf{y}}$ and $\mathcal{C}_i$ is the i-th class of the set of C possible classes. Targets, or labels, are one-hot encoded vectors indexing the true category of the sample:

$$y_i = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{otherwise,} \end{cases}$$

where $\mathcal{C}_j$ is the true category represented by $\mathbf{y}$. This encoding arises naturally from the assumption of a categorical likelihood, $p(\mathbf{y}|f, \mathbf{x}) = \prod_{j=1}^{C} \hat{y_j}^{y_j}$.

The logit space

Regression models can be applied to the classification task by taking a Generalized Linear Model (GLM) approach [13]. Let's have a regression model making predictions $\hat{\mathbf{z}}$ in the whole real domain, $\hat{\mathbf{z}} \in \mathbb{R}^C$. In GLM we assume that the conditional probability of the classes $p(\mathbf{y}|f, \mathbf{x})$ follows a distribution from the exponential family. In our setting we assume it follows a categorical distribution:

$$p(\mathbf{y}|f, \mathbf{x}) = \exp\left(\boldsymbol{\eta}^T \mathbf{y}\right), \tag{2.2}$$

where $\boldsymbol{\eta}$ are the natural parameters of the distribution which are subject to $\sum_{i=1}^{C} \exp\left(\eta_i\right) = 1$ and are obtained as some function of $\hat{\mathbf{z}}$.

Intermediate predictions $\hat{\mathbf{z}}$ are related to the expected value of targets $\mathbf{y}$ via a *link function* ψ:

$$\mathbb{E}[\mathbf{y}|\mathbf{x}, f] = \hat{\mathbf{y}} = \psi^{-1}(\hat{\mathbf{z}}) \tag{2.3}$$

As is common in the literature, we use the *canonical link function*, which is the link function that identifies $\hat{\mathbf{z}}$ with the natural parameters of $p(\mathbf{y}|f, \mathbf{x})$, i.e. $\boldsymbol{\eta} \propto \hat{\mathbf{z}}$. Notice that this choice is made for convenience and that other link functions like the *probit* may be preferred depending on the application. Particularizing to the categorical distribution, the canonical link function is given by the *Logit* function:

$$\hat{\mathbf{z}} = \psi(\hat{\mathbf{y}}) = \log \hat{\mathbf{y}} + \log Z, \tag{2.4}$$

where $Z \in \mathbb{R}_{>0}$ is a normalization constant to ensure the constraint over $\boldsymbol{\eta}$. Reversing the equation above and noting that $\sum_{i=1}^{C} \hat{\mathbf{y}} = 1$ we obtain the predictions as the *softmax* of $\hat{\mathbf{z}}$:

$$\hat{\mathbf{y}} = \frac{\exp \hat{\mathbf{z}}}{Z} = \frac{\exp \hat{\mathbf{z}}}{\sum_{j=1}^{C} \exp\left(\hat{z}_j\right)}, \tag{2.5}$$

Because of the name of the link function, $\hat{\mathbf{z}}$ is called the *logit* vector. In this work, we take directly logit predictions, this design choice allows our method to be used independently of the link function. Moreover, the application of the *softmax* may dismiss potential valuable information. Two equivalent logit vectors $\mathbf{z}_n, \mathbf{z}_m$ which differ only in a scalar constant,i.e. $\mathbf{z}_n = \mathbf{z}_m + k$, $k \in \mathbb{R}$, will yield the same probabilistic prediction $\mathbf{y}_n = \mathbf{y}_m$ despite the value of k.

2.2 Neural Networks

Neural Networks are a family of machine learning models with multiple applications. Broadly speaking, the model learns a function $f : X \to Y$ between an input space $\mathcal{X} \in \mathbb{R}^d$ and an output space $\mathcal{Y} \in \mathbb{R}^{d'}$. For instance, $\mathcal{X}$ may be the set of melanoma images and $\mathcal{Y}$ the categorical distribution over the possible classes of melanoma. Neural Networks are composed by sequential stages or layers, each of which defines a map to an intermediate vector space, $f(\mathbf{x}) = (f^L \circ f^{L-1} \circ ... \circ f^1)(x)$, where L is the total number of layers. Layers are, generally, parametrized. We denote the parameters of layer l by θ_l, and the output of each layer as $\mathbf{x}^{(l)}$:

$$\begin{aligned}
\mathbf{x}^{(0)} &= \mathbf{x}, \\
\mathbf{x}^{(1)} &= f^1(\mathbf{x}, \theta_1), \\
\mathbf{x}^{(l)} &= f^l(\mathbf{x}^{(l-1)}, \theta_l), \\
\hat{\mathbf{y}} = \mathbf{x}^{(L)} &= f^L(\mathbf{x}^{(L-1)}, \theta_L),
\end{aligned}$$

Each dimension of intermediate mappings is called neuron or unit, so if the output of layer l is of dimension h, $\mathbf{x}^{(l)} \in \mathbb{R}^h$, it is said that layer l has h neurons. The whole net is parametrized by $\boldsymbol{\theta} = \{\theta_l\}_{l=1}^L$. Parameters are trained with the backpropagation algorithm [14], which propagates the gradient of a cost function backwards throughout the net. Backpropagation works by iteratively applying the chain rule. For a single observation $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}$:

$$\begin{aligned}
\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \theta_L} &= \frac{\partial f^L(\mathbf{x}^{(L-1)}, \theta_L)}{\partial \theta_L} = \frac{\partial \mathbf{x}^{(L)}}{\partial \theta_L} \\
\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \theta_{L-1}} &= \frac{\partial \mathbf{x}^{(L)}}{\partial \mathbf{x}^{(L-1)}} \frac{\partial \mathbf{x}^{(L-1)}}{\partial \theta_{L-1}} \\
\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \theta_l} &= \frac{\partial \mathbf{x}^{(L)}}{\partial \mathbf{x}^{(L-1)}} \frac{\partial \mathbf{x}^{(L-1)}}{\partial \mathbf{x}^{(L-2)}} ... \frac{\partial \mathbf{x}^{(l)}}{\partial \theta_l}
\end{aligned} \tag{2.6}$$

Notice that layers should be differentiable in order to apply Backpropagation. Neural Networks provide very flexible models given the possibility

of arbitrary big number of layers and high-dimensional intermediate spaces. Moreover, different architectures of Neural Networks adapt better to different domains regarding the kind of map that its layers define. For example, Convolutional Neural Networks (CNNs) provide state of the art performance on Computer Vision tasks and Recurrent Neural Networks (RNNs) are well suited to model sequential data.

2.2.1 The Multi-Layer Perceptron (MLP)

MLPs are probably the most basic architecture of Neural Networks. Its layers are typically known as Fully Connected (FC), each FC layer consists of a linear mapping and a differentiable non-linearity which is applied element-wise.

$$\mathbf{x}^{(l)} = g(W^l \mathbf{x}^{(l-1)} + b^l), \tag{2.7}$$

where $g()$ is the non-linearity, also known as activation function, and the parameters of the layer are $\theta_l = (W^l, b^l)$ the weight matrix W^l and the bias vector b^l.

MLPs can be seen as a direct extension of, either, linear or logistic regression models depending on the task at hand. In this work we use this architecture to learn a mapping from a set of uncalibrated logits to their corresponding calibrated ones.

2.2.2 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks allow to increase the dimensionality of the network and, at the same time, reduce the number of parameters by sharing weights. A smaller weight matrix is convoluted with the input:

$$\mathbf{x}^{(l)} = g(W^l * \mathbf{x}^{(l-1)} + b^l), \tag{2.8}$$

where $*$ denotes the convolution operator. This kind of layers results very useful in situations where the same patterns may arise in different parts of the input. For instance, a face in a photograph remains a face independently of where in the photograph it appears.

AlexNet [15] is probably the most famous example of a CNN, it marked an inflexion point when it won the ImagenetNet Large Scale Visual Recognition Challenge [3]. Its major breakthrough was the depth of the network, it has 5 convolutional layers followed by 3 fully connected. Computation on GPU was essential to make the training feasible, but not sufficient. AlexNet success is also due to the use of the Rectified Linear Unit or ReLU as activation function.

2.2.3 Other architectures

So far we have given a brief description of MLPs and CNNs as they are the most relevant architectures concerning this work. There are, however, other architectures that are worth mentioning. Recurrent Neural Networks (RNNs) introduce a temporal structure by feeding layers with their outputs at the previous time-step. This networks are trained with an adapted version of backpropagation called *backpropagation through time* [16]. To account for long-term dependencies Long Short-Term Memory (LSTMs) extend RNNs by introducing the *hidden state*, an internal variable that may, or may not, be updated depending on the input, allowing layers to propagate information between distant time-steps. Transformer Networks [17] avoid the sequential computation inherent of RNNs by exploiting *attention* [18], a mechanism that weights the elements of a sequence by their relative importance between each other.

These architectures are state of the art in several applications like Machine Translation [17], Image Captioning [4], and Speech recognition [5] among others.

2.2.4 Improvements on the learning process

As we mention before, Neural Networks are trained with backpropagation. Although this method dates from the 60s, it hasn't been until the last decade that Neural Networks have become bigger and competitive. In part, because only now computational power is enough to take the task. However, this is only half of the story, improvements on the learning process has also been central in the development of bigger models.

ReLU

One major leap was the use of the ReLU:

$$ReLU(x) = \begin{cases} x & \text{if } x > 0, \\ 0 & \text{otherwise,} \end{cases}$$

This form divide the input space into 2 regions, the active, where inputs are positive so the function propagates its values, and the inactive or dead region, where inputs are negative and the function propagates no information forward.

Previous implementations of Neural Networks usually used the hyperbolic tangent or the sigmoid function as non-linearities. These can be seen as differentiable versions of the Heaviside step function used in traditional perceptrons. The problem with this functions is that its derivative is far from zero only in a small range of the input, for big or small values it comes close to 0. Taking a look at the computation of gradients in Equation 2.6, it

becomes clear that gradients propagating to deep layers can easily vanish as the number of factors near zero increase. The gradient of the ReLU function can only take values 1 or 0, so gradients can be potentially back-propagated much deeper as long as enough units remain in the active region.

Skip Connections

Another important breakthrough is the use of skip connections. They were introduced in [19] improving greatly the training of very deep CNNs and pushing further the state of the art on many Computer Vision tasks. The idea of skip connections is simple though. They can be seen as shortcuts between non-contiguous layers. For instance, layer $l + k$ with $k > 1$ may take as input the outputs of the layers $l + k - 1$ and l:

$$\mathbf{x}^{(l+k)} = f^{l+k}(\mathbf{x}^{(l+k-1)}, \mathbf{x}^{(l)}, \theta_{l+k})$$

Now, the gradients for layer l can be computed as follows:

$$\frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \theta_l} = \frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \mathbf{x}^{(l+1)}} \frac{\partial \mathbf{x}^{(l+1)}}{\partial \mathbf{x}^{(l)}} \frac{\partial \mathbf{x}^{(l)}}{\partial \theta_l} + \frac{\partial \mathcal{L}(\boldsymbol{\theta})}{\partial \mathbf{x}^{(l+k)}} \frac{\partial \mathbf{x}^{(l+k)}}{\partial \mathbf{x}^{(l)}} \frac{\partial \mathbf{x}^{(l)}}{\partial \theta_l}$$

Since the layer $l + k$ is much closer to the output, the second term provides a less noisy gradient. The multiplicative nature of the gradients, in combination with the stochasticity of the Monte Carlo methods used for the gradient estimation, made unfeasible the training of deep layers. However, skip connections introduce paths in the backpropagation step that allow gradients to reach very deep into the network with very few steps.

Skip connections have been exploited since then by following state of the art CNN architectures. In [20], for instance, the authors introduce *dense blocks*, stages of contiguous layers in which each layer has a skip connection with every subsequent layer.

2.2.5 The training of Neural Networks

Neural Networks are trained via backpropagation to minimize a cost function. The choice of the cost function specifies the optimization problem.

Point Estimate Techniques

The two most common approaches are probably Maximum Likelihood (ML) and Maximum a Posteriori (MAP). These techniques are called point-estimate methods since they learn fixed values of the Neural Network parameters.

Maximum Likelihood optimizes the likelihood of the parameters given the empirical data. Since the logarithm is a monotonic increasing function, setting the cost function to the Negative Log-Likelihood (NLL) is equivalent to maximizing the likelihood:

$$\begin{aligned}\mathcal{L}(\mathcal{D} \mid \boldsymbol{\theta}) &= \log p(Y \mid \boldsymbol{\theta}, X) \\ &= \sum_{n=1}^{N} \log p(\mathbf{y}_n \mid \boldsymbol{\theta}, \mathbf{x}_n) \\ &= \sum_{n=1}^{N} \sum_{j=1}^{C} y_{nj} \log \hat{y}_{nj} \end{aligned} \tag{2.9}$$

where we assume a categorical likelihood and the decomposition into a sum results from the i.i.d. assumption of the empirical data. The notation y_{nj} index the j-th dimension of the n-th sample $\mathbf{y}_n$.

Maximum a Posteriori can be interpreted as adding a regularization term to the ML criterion. Instead of maximizing the likelihood MAP maximizes the posterior of the parameters:

$$\mathcal{L}(\mathcal{D} \mid \boldsymbol{\theta}) = \log p(\boldsymbol{\theta} \mid \mathcal{D}) = \sum_{n=1}^{N} \log p(\mathbf{y}_n \mid \boldsymbol{\theta}, \mathbf{x}_n) + \log p(\boldsymbol{\theta}) \tag{2.10}$$

The $\log p(\boldsymbol{\theta})$ acts as the aforementioned penalty term. For example, in the case of an isotropic gaussian prior, this term is equivalent to a L2 penalty.

Bayesian Approach

Another approach, much less employed and the taken in this work, considers making inference over the parameters posterior $p(\boldsymbol{\theta}|\mathcal{D})$ so we can make new predictions as:

$$p(\mathbf{y}_*|\mathbf{x}_*, \mathcal{D}) = \int p(\mathbf{y}_*|\mathbf{x}_*, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\mathcal{D}) d\boldsymbol{\theta} \tag{2.11}$$

However, the problem of inferring the parameters posterior is intractable, so approximate solutions are required. Moreover, this approaches require more computational cost and usually perform worse than the point-estimate equivalents [12].

2.3 Calibration

In the scope of probabilistic classifiers, calibration refers to the quality of estimates as true correctness likelihood —i.e. how closely confidence scores resemble the probability of the class. For instance, in the case of a well calibrated weather forecaster, around 80% of the times it announces rain with an 80% of confidence, it actually rains. This notion can be easily extended to the multi-class setting. Taking the set of predictions matching an specific estimate, e.g. (0.4, 0.3, 0.3), the prevalence of each class on this

set should be the same as the predicted confidence, (0.4, 0.3, 0.3), if the model presents good calibration.

The concern with quality confidence predictions is not new and has been widely investigated in applications like forecasting [7, 8] for decades. However, in the past years there has been a growing interest on the calibration of Neural Networks. The reason might be that, as have been pointed recently [6], accurate modern models exhibit poor calibration. In this section we introduce the concept of calibration from an intuitive perspective and describe how its measured in the literature. Then we review related work regarding the calibration of Neural Networks, in particular post-processing techniques.

2.3.1 Measures

Let $\mathbb{P}(X, Y)$ be the joint probability of some input-class data collection $\mathcal{D} = (X, Y)$ where Y indicates the true class of inputs X. We say a classifier f is perfectly calibrated on $\mathcal{D}$ if for every possible prediction $\hat{\mathbf{y}}$ the distribution of the target given the prediction is exactly the prediction:

$$\mathbb{P}\big(Y \,|\, \hat{\mathbf{y}}\big) = \hat{\mathbf{y}} \tag{2.12}$$

Any deviation between both terms is called Calibration Error and most calibration measures are based on this equation.

Expected Calibration Error

The Expected Calibration Error (ECE) as defined in [6] measures the absolute difference between both terms in Equation 2.12, but considering only top-label predictions, this is, the most probable class according to the estimate:

$$\mathbb{E}\left[\left|\mathbb{P}\big(Y = \arg\max(\hat{\mathbf{y}}) \,|\, \hat{\mathbf{y}}\big) - \max(\hat{\mathbf{y}})\right|\right] \tag{2.13}$$

Computing this measure turns unfeasible in practice since $\hat{\mathbf{y}}$ is a continuous variable and we account only for a finite number of samples. The ECE is approximated in the following way:

First, the probability range is divided into M equally spaced intervals or bins $I_m = \big(\frac{m-1}{M}, \frac{m}{M}\big]$, $m = 1, 2, .., M$. Let B_m be the set of samples that its prediction falls in the interval I_m, i.e. $B_m = \{(\mathbf{x}, \mathbf{y}) \in \mathcal{D} \,|\, \max f(\mathbf{x}) \in I_m\}$.

Then, we compute the average accuracy and confidence for each interval:

$$acc(B_m) = \frac{1}{|B_m|} \sum_{(\mathbf{x},\mathbf{y}) \in B_m} \mathbf{1}(\mathbf{y} = \arg\max f(\mathbf{x})),$$

$$conf(B_m) = \frac{1}{|B_m|} \sum_{(\mathbf{x},\mathbf{y}) \in B_m} \max f(\mathbf{x}),$$

where $\mathbf{1}$ is the indicator function and $|B_m|$ the number of samples in B_m. The $acc(B_m)$ and the $conf(B_m)$ approximate the left and right terms of Equation 2.12 taking only top-label predictions.

Finally, the ECE is approximated by:

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{N} \Big| acc(B_m) - conf(B_m) \Big|, \tag{2.14}$$

where N is the number of samples in $\mathcal{D}$. When we report results or discuss performance in terms of ECE we refer to the estimate given by Equation 2.14.

Recent work argues that the ECE may not be enough to provide a comprehensive estimate of the model calibration [21] and that it is biased [22]. However, to the best of our knowledge the ECE is the most reported metric in the literature on calibration of Neural Networks and has a desirable intuitive interpretation. For these reasons we use ECE in this work to evaluate calibration.

Maximum Calibration Error

In some applications it may be of interest minimizing the worst-case deviation. This is given by the Maximum Calibration Error (MCE), which is estimated as:

$$MCE = \max_{m} \Big| acc(B_m) - conf(B_m) \Big| \tag{2.15}$$

This measure is much less used in the literature than the ECE, although it may be important in high risk applications.

Proper Scoring Rules

Proper scoring rules (PSR) are measures of a probabilistic model quality. Proper scoring rules are minimized when the model recovers the ground truth $\mathbb{P}(Y|X)$ making them appealing as cost functions. A model that minimizes a PSR will also present perfect calibration by means of Equation 2.12 because, then, $\hat{\mathbf{y}} = \mathbf{y}$. The most commonly used PSRs are the logarithmic scoring rule, which turns into the NLL in our setting, and the Brier scoring rule, which resembles the mean square error.

However, proper scoring rules do not measure directly calibration, but the sum of calibration and a *discrepancy* or *refinement* component, a property related with the accuracy of the model [8, 23, 24]. Thus, minimizing a PSR does not necessarily imply improving calibration because the refinement component may be optimized instead; moreover, Neural Networks optimizing the NLL become overconfident, improving discrimination but yielding poor calibration [6]. This means that these models minimize the NLL by increasing the confidence scores of already correctly classified samples in

such a way that outputs become more peaked, closer to one-hot vectors. This might be due to the limitations of the training set, and can derive in overfitting from the perspective of the calibration performance.

2.3.2 Methods

We can distinguish methods that improve calibration into two different groups. First, those that involve the training step. Most of these methods target related properties like robustness to out-of-distribution samples or random-noise; thus, they improve calibration implicitly. This group includes data augmentation techniques [25, 11], alternative objective functions [26, 27], specific models that better capture uncertainty [28], and taking a Bayesian approach [29] among others. These methods increase the overall computational cost of the training procedure, and may even degrade performance, like in the case of approximate methods for Bayesian modeling. Furthermore, in case of input distribution shift, a common problem on real life scenarios, the whole model may require retraining with all its associated cost.

The other approach is to re-calibrate an already trained classifier. These methods are model agnostic since re-calibration is a post-processing step. They only require a set of ill-calibrated predictions $\hat{Y}$ and their corresponding ground-truth Y. A calibration model is fitted to approximate the conditional distribution of the labels given the predictions $\mathbb{P}(Y|\hat{Y})$. This property implies that the computational overhead to pay for calibrated predictions is fixed despite the complexity of the problem. This can be very useful in situations were very big models are needed or when the input distribution is prone to change during time.

Our method belongs to this later category, so we review re-calibration methods on more detail.

Calibration of Binary Classifiers

One common approach is to divide the problem into C one-vs-all binary classification problems, train a calibrator in each one, and aggregate results. Each calibrator h_i outputs a confidence $\hat{q}_i$, stacking them together into a vector and normalizing it we obtain the new prediction vector $\hat{\mathbf{q}}$. Calibrators for binary classification problems fit a monotonically increasing function to the ill-calibrated predictions. Histogram Binning [10], divides the input range, $[0, 1]$, into equal bins and assigns a score to each bin. The prediction for a given sample is just the score of the bin on which it falls. The scores are optimized according to the Brier rule but constrained to fulfil bin-wise the monotonically increasing condition. Isotonic Regression [30] fits a piecewise constant function yielding the binning scheme, with variable size and number of bins, that best fits the data. Bayesian Binning into Quantiles [31]

extends the binning approach taking a Bayesian average over all possible binning schemes. Platt Scaling [9] fits the parameters $a \in \mathbb{R}, b \in \mathbb{R}$ of the function

$$\hat{q}_i = \sigma(az_i + b), \tag{2.16}$$

to the NLL where $\hat{z}_i$ is the *logit* associated to $\hat{y}_i$, and σ is the *sigmoid* function.

Temperature Scaling

Platt Scaling can be extended to the multiclass setting by other means, rather than one-vs-all, like matrix and vector scaling, where the function is defined by:

$$\hat{\mathbf{q}} = \sigma_{SM}(W\hat{\mathbf{z}} + \mathbf{b}), \tag{2.17}$$

where $\hat{\mathbf{z}}$ is the *logit* vector associated to $\hat{\mathbf{y}}$, σ_{SM} is the *softmax* function as defined in Equation 2.5, $b \in \mathbb{R}^C$, and $W \in \mathbb{R}^{C \times C}$ for matrix scaling and $W \in \mathbb{R}^C$ for vector scaling.

However, is the simplest extension of Platt Scaling, Temperature Scaling, which performs the best in practice [6]. Predictions take the form:

$$\hat{\mathbf{q}} = \sigma_{SM}(\frac{\hat{\mathbf{z}}}{T}) \tag{2.18}$$

The *temperature* $T \in \mathbb{R}_{>0}$ is trained to optimize the NLL. A higher temperature will attenuate differences between *logits* z_i, resulting in a much more uniform output and relaxing the confidence on predictions. On the other hand a lower temperature will amplify them, turning into over-confident predictions. Notice that the maximum is preserved independently of T, thus this method does not change the accuracy of the base model.

Temperature Scaling improves calibration when evaluated under a test set sampled from the same distribution as the train set, i.e. under no distribution shift. It mitigates the flawed tendency of Neural Networks to output overconfident predictions. This observation in conjunction with that that more complex methods tend to over-fit led authors in [6] to the conclusion that calibration is a low dimensional problem and inherently simple. However, recent work [1] argues with empirical evidence that this behaviour is due to a bad uncertainty modeling and that calibration might be a much more complex task, specially in high dimensional problems, i.e. with a high number of classes C. Moreover, the simplicity of Temperature Scaling makes it insufficient to model uncertainty under changes of data distribution and rapidly loose performance under distribution shifts [32].

Decoupled Bayesian Neural Networks

We extend the work done in [1] where the authors use Bayesian Neural Networks (BNN) as a calibration method to approximate the conditional distribution $\mathbb{P}(\mathbf{y}|\hat{\mathbf{z}})$.

The inherent uncertainty modeling of the Bayesian approach makes it a desirable option when seeking good calibration. To this aim, Bayesian Neural Networks specify a prior over its parameters $p(\boldsymbol{\theta})$ and infer the posterior given a dataset $\mathcal{D}$. But the need of aproximate methods to compute the posterior and the difficulty of assigning meaningful priors to the parameters renders them useless as soon as complexity scales. Besides, their implementation and computation are significantly slower than their point-estimate equivalents.

Motivated by the success of Neural Networks and the appealing properties of the Bayesian approach, the authors propose to use BNNs as a decoupled stage. A big DNN is trained on the classification task, which may include very high dimensional inputs such as images and audio segments. Then, the model is calibrated with a much smaller BNN, hence combining the high discriminative performance of modern Neural Networks with the desired uncertainty modeling of the Bayesian approach. The application of the BNN is possible due to the simplification of the problem. Mapping predictions to their calibrated counterparts requires a much more smaller architecture than the required to fit the whole task.

The posterior is approximated using a Mean-field Variational Inference approach. The variational distribution is a fully factorized Gaussian, and the prior is the standard Gaussian. Authors test two implementations, one based on the reparametrization trick [33] alone, which they call MFVI, and the other, MFVILR, incorporating the local reparametrization trick [34] to reduce the variance of the gradients estimations.

One well known drawback of the Variational Inference (VI) approximation is that it under-estimates the posterior variance [35]. To limit this effect, authors introduce an hyperparameter K that represents the number of Monte Carlo samples to draw from the posterior when making predictions. This hyperparameter is selected on a validation set to minimize the ECE.

In this work we propose to use the recent framework of Generalized Variational Inference to overcome this and other limitations of the standard VI approach. In the next chapters we introduce this framework and describe our method, stating more differences and similarities with the original presented in [1].

Chapter 3

Generalized Variational Inference

In this chapter we introduce the novel framework of Generalized Variational Inference (GVI) [2], a generalization of the well known Variational Inference method for approximate inference. We closely follow the lineup of the main work in this chapter and refer the reader to it for proofs and more in depth explanations. First, we establish the traditional Bayesian inference paradigm as an optimization problem. This allow us to, then, identify three underlying assumptions behind the Bayesian approach which cannot be fully met in practice but reasonably satisfied [36]. Modern machine learning techniques completely disregard these constraints [2] hindering the standard Bayesian approach. As a means to maintain the virtues of modern models along the desirable properties of the Bayesian paradigm, authors propose a generalization of the standard setting which can account for these violations, the Rule of Three (RoT). Lastly, we introduce some divergences that may be used instead of the Kullback–Leibler divergence (KL) to produce more robust approximations.

3.1 Bayesian Inference as an Optimization Problem

Let $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ be the vector parameter of interest, $\mathcal{D}$ a set of observations, and $p(\mathcal{D}|\boldsymbol{\theta})$ a likelihood function relating both. Given these, bayes rule allows us to compute the posterior distribution $p(\boldsymbol{\theta}|\mathcal{D})$, updating a prior distribution $p(\boldsymbol{\theta})$ which accounts for previous beliefs over $\boldsymbol{\theta}$:

$$p(\boldsymbol{\theta}|\mathcal{D}) \propto p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta}) \tag{3.1}$$

Despite being this one of the most common representations, bayesian posteriors can alternatively be defined as an optimization problem [37, 38].

Let $\{o_n\}_{n=1}^N = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N = \mathcal{D}$ be the set of i.i.d. samples in $\mathcal{D}$. Then, Equation 3.1 can be expressed as:

$$p(\boldsymbol{\theta}|\mathcal{D}) \propto p(\boldsymbol{\theta}) \prod_{n=1}^{N} \exp \log(o_n|\boldsymbol{\theta})$$

In order to better grasp concepts in the following sections, we introduce the following notation heavily inspired on that of the original work. We represent the likelihood as a loss function $l(\boldsymbol{\theta}, o_n) = -\log(o_n|\boldsymbol{\theta})$, i.e. the logarithmic rule. We denote the prior and the posterior distributions as $\pi(\boldsymbol{\theta})$ and $q_B^*(\boldsymbol{\theta})$ respectively, where the B stands for the standard Bayesian setting and the period $*$ implies that it is the solution of an optimization problem. This problem can be formulated as:

$$q_B^*(\boldsymbol{\theta}) = \underset{q \in \mathcal{P}(\Theta)}{\arg\min} \left\{ \mathbb{E}_q \left[\sum_{n=1}^{N} l(\boldsymbol{\theta}, o_n) \right] + D_{\mathrm{KL}}(q \,||\, \pi) \right\}, \tag{3.2}$$

where $\mathcal{P}(\boldsymbol{\Theta})$ is the space of probability measures over $\boldsymbol{\Theta}$ and D_{KL} is the Kullback–Leibler divergence. Notice that this formulation ease the task of identifying certain properties of the Bayesian approach. For instance, the regularization effect of the prior, it can be readily checked that the KL term is preventing $q_B^*(\boldsymbol{\theta})$ from converging to the empiric risk minimizer, a delta on the value of $\boldsymbol{\theta}$ that minimizes the loss l, effectively resembling the point-estimate approach. Although this behaviour may seem appealing at first glance, we should notice that is the variance of the posterior which accounts for the good uncertainty modeling that makes so appealing the Bayesian paradigm. Moreover, is the expected value over the posterior, accounting for every possible value of $\boldsymbol{\theta}$, which avoids making over-confident predictions [39].

3.1.1 Variational Inference

Exact inference of the posterior $q_B^*(\boldsymbol{\theta})$ turns to be intractable in many practical scenarios. Sometimes it can be evaluated up to a normalizing constant, although this is not always the case neither particularly useful if we want to make predictions over new data. There exits sampling methods that allow for asymptotically exact computations, but they incur in a high computational cost that makes them unpractical for complex models like Neural Networks.

Approximate methods recover a parametric distribution $q \in \mathcal{Q}$ supposed to closely resemble the posterior $q_B^*(\boldsymbol{\theta}) \simeq q$, here $\mathcal{Q} = \{q(\boldsymbol{\theta}|\kappa) \mid \kappa \in K\}$ is a κ-parametrized family of distributions. In particular, Variational Inference approximates the posterior $q_B^*(\boldsymbol{\theta}) \simeq q_{\mathrm{VI}}^*(\boldsymbol{\theta})$ by minimizing the KL divergence:

$$q_{\mathrm{VI}}^*(\boldsymbol{\theta}) = \underset{q \in \mathcal{Q}}{\arg\min}\, D_{\mathrm{KL}}(q \,||\, q_B^*) \tag{3.3}$$

It can be easily proved that this objective is equivalent to the one in 3.2 but restricting the space to the family $\mathcal{Q}$.

Zero forcing behaviour

The form of the divergence in Equation 3.3 is also called the Reverse KL since q_B^* is the target distribution but we take the expected value over q. Expressing the divergence in its integral form:

$$D_{\mathrm{KL}}(q \,||\, p) = \int q \log \frac{q}{p} \, du,$$

where q and p are distributions over u, and taking that we want to approximate p with q, we can easily observe the following behaviour. Whenever q is big and p is small, the divergence would incur in a high penalty. On the opposite case, big p and small q, this would not be necessarily the case, in fact pushing $q \to 0$ would minimize the divergence. This means that when minimizing the Reverse KL q tends to over-concentrate avoiding regions of low p. This behaviour is known as *zero forcing* and it is responsible for the variance under-estimation we mention in Section 2.3.2 and that is characteristic of the VI method.

Discrepancy VI

Although not recovered by the GVI framework, it is worth mentioning another generalization of VI which authors call Discrepancy VI. Multiple works generalize the objective in Equation 3.3 by taking other statistical divergences rather than the KL. For instance, the well-known, Expectation Propagation [40] minimizes the direct KL, i.e. $D_{\mathrm{KL}}(q_B^* \,||\, q)$. More recent work includes BB-α [41], which minimizes α-divergences [42], a parametrized family of divergences.

Even though these methods can be proven to yield sub-optimal posteriors in the KL sense (in contrast to standard VI), in practice, they often rise better approximations. This is because, either the initial objective in Equation 3.2 is incorrectly specified, e.g. a faulty likelihood, or the approximating family $\mathcal{Q}$ is poorly chosen in the sense that $q \simeq q_B^*$ does not hold for any $q \in \mathcal{Q}$.

3.2 Violated Assumptions underlying the Bayesian Paradigm

The theoretic properties of Bayesian posteriors makes them appealing for the task of inference. As we implicitly introduce above, the Bayesian paradigm builds upon three fundamental assumptions:

(P) The prior $\pi(\boldsymbol{\theta})$ is correctly specified. This is, it reflects exactly any prior beliefs, including previous observations or domain expertise.

(L) There exists some unique $\boldsymbol{\theta}^*$ such that the likelihood model recovers the true data generating mechanism, i.e. $o_n \sim p(o_n|\boldsymbol{\theta}^*)$.

(C) The complexity of computing the posterior q_B^* can be ignored, this is, it assumes that the computational power is always enough.

The approach taken in many traditional applications can be closely outlined as follows. A simple model would be chosen as an informed guess of the true data generating mechanism. The simplicity implies that the computational requirements rarely exceed the actual capacity, thus **(C)** is hardly a constraint. The method of model selection ensures that both, the prior and the likelihood, closely reflect beliefs about the distribution and the data generating mechanism. Moreover, an iterative procedure is usually taken in order to find a prior and a likelihood such that **(P)** and **(L)** are satisfied in practice [36].

These assumptions are, however, consistently ignored in modern machine learning models.

(P) First, priors are typically chosen for convenience. For instance, the use of conjugate priors restricts the possible distributions. In addition, it is common practice to use over-parametrized likelihoods models, e.g. Neural Networks. Such high number of parameters prevent them to be interpretable at all; hence, we cannot specify prior beliefs in a meaningful way. Even more, priors are conventionally chosen for their regularization effect.

(L) Modern machine learning methods usually specify highly parametrized models that are capable of fitting any collection of samples, overlooking completely the search of the true data generating mechanism. The high number of parameters often results in un-identifiable functions of $\boldsymbol{\theta}^*$, meaning that this is neither interpretable nor unique. Moreover, complex models try to recover the *typical* behaviour of the data. Although this may seem a subtlety, it implies that nuances of the data have an exaggerated impact on the inference since, because of **(L)**, the posterior will try to recover all the data, but modern modeling treat all samples as *typical*. Proper likelihood modeling accounts for every aspect of the population; thus, its capable of discriminating outliers and other discrepancies from the *typical* behaviour.

(C) The increasing complexity of modern statistical models results in intractable posteriors impelling the use of approximations.

3.3 The Rule of Three and Generalized Variational Inference

In order to address the above observations, the authors propose the Rule of Three (RoT). This framework allows to tackle issues concerning each of the three conditions independently by generalizing the formulation in Equation 3.2. The posterior $q^* \in \Pi$ is obtained by solving an optimization problem over some space $\Pi \subseteq \mathcal{P}(\Theta)$ and which objective seeks to minimize two independent criteria:

- The expected sample loss $\sum_{n=1}^{N} l(\boldsymbol{\theta}, o_n)$ under q^*.
- The deviation from the prior $\pi(\boldsymbol{\theta})$ measured by some statistical divergence D.

The RoT constructs, then, the posterior as the solution of an optimization problem with three degrees of freedom, a space of possible posteriors Π, a loss function linking the parameter of interest $\boldsymbol{\theta}$ with the set of observations $\mathcal{D}$, and a divergence D that imposes a cost for deviating from the prior $\pi(\boldsymbol{\theta})$:

$$q^* = \underset{q \in \Pi}{\arg\min} \left\{ \mathbb{E}_q \left[\sum_{n=1}^{N} l(\boldsymbol{\theta}, o_n) \right] + D(q \,||\, \pi) \right\} \tag{3.4}$$

The RoT provides a flexible approach to building posteriors, its modularity allows to address the issues regarding **(P)**, **(L)**, and **(C)** independently:

(P) Changing the divergence D allows to change the way the prior influences the posterior accounting for possible prior misspecification and regularizing uncertainty quantification.

(L) To address model misspecification, this amounts to changing the loss function l.

(C) The space Π can be chosen to address the problems with limited computational power.

The method of GVI simply amounts to applying the RoT constraining the space of possible posteriors to a variational family like in VI. This is, $\Pi = \mathcal{Q}$, where $\mathcal{Q} = \{q(\boldsymbol{\theta}|\kappa) \mid \kappa \in K\}$.

3.4 Robustness to Prior Misspecification through Robust Divergence

In this work we focus on the selection of robust divergences for GVI. In this setting, robust divergences are used as a penalty term quantifying deviation from the prior, this should not be confused with Discrepancy VI,

in which robust divergences are also used, but they quantify, instead, the deviation of the variational posterior from the standard Bayesian posterior. The authors of the original work refer to an overview of robust divergences given in [43]. This work introduces a comprehensive family of parametrized divergences that recovers as special cases many well-known divergences, the $\alpha\beta\gamma$-divergence. Because of its apparent success on the experiments in [2] we study the special case of the Rényi's α-divergence [44]. Taking the parametrization of [43] the Rényi's α-divergence $D_{\mathrm{AR}}^{(\alpha)}$ is given by:

$$D_{\mathrm{AR}}^{(\alpha)}\big(q(\boldsymbol{\theta}) \,||\, \pi(\boldsymbol{\theta})\big) = \frac{1}{\alpha(\alpha-1)} \log\left(\int q(\boldsymbol{\theta})^{\alpha} \pi(\boldsymbol{\theta})^{1-\alpha} d\boldsymbol{\theta} \right), \tag{3.5}$$

where $\alpha \in \mathbb{R} \setminus \{0, 1\}$, parametrizes the divergence. It can be readily seen that for higher values of α we give low weight to the prior, favouring smaller posterior variances towards the empiric risk minimizer; in contrast, lower values of α will penalize more any deviations from the prior. This behavior can be obtained too just by weighting the KL divergence, i.e. βD_{KL}, and varying β in the opposite direction of α for the same effect. However, the weighted KL is not robust against prior misspecification, in which case it imposes a trade-off between a large variance around an incorrect prior and a low variance around the empiric risk minimizer. The α-divergence, on the other hand, produce posteriors that are robust to the prior and account for adequately large variances.

It is worth remarking that in the limit the $D_{\mathrm{AR}}^{(\alpha)}$ recovers the KL and the reverse KL:

$$\lim_{\alpha \to 1} D_{\mathrm{AR}}^{(\alpha)}\big(q(\boldsymbol{\theta}) \,||\, \pi(\boldsymbol{\theta})\big) = D_{\mathrm{KL}}\big(q(\boldsymbol{\theta}) \,||\, \pi(\boldsymbol{\theta})\big)$$
$$\lim_{\alpha \to 0} D_{\mathrm{AR}}^{(\alpha)}\big(q(\boldsymbol{\theta}) \,||\, \pi(\boldsymbol{\theta})\big) = D_{\mathrm{KL}}\big(\pi(\boldsymbol{\theta}) \,||\, q(\boldsymbol{\theta})\big)$$

Chapter 4

Design

In this chapter we describe our method, an extension of the decoupled BNNs proposed in [1], we explicitly remark the differences between our approach and the original one. First, we describe the statistical model for calibration. Then, we show how to obtain better posteriors of the model parameters using GVI, the main contribution of this work. Lastly, we describe the implementation of the optimization problem.

4.1 Model

Following the notation we introduce in Chapter 2, let $\{\hat{\mathbf{z}}_n\}_{n=1}^N$ be a set of uncalibrated logits and $\{\mathbf{y}_n\}_{n=1}^N$ their corresponding ground-truth labels. We want to fit some model f which is $\boldsymbol{\theta}$-parametrized to make new predictions $\hat{\mathbf{q}}$ that are well calibrated. In this work we limit our experiments to logit vectors obtained from the output of DNNs, but the method does not impose such restriction, $\hat{\mathbf{z}}$ may be the output of any probabilistic classifier.

As for the choice of model f, we use multilayer perceptrons. It may seem excessive to use such an expressive model for a task that is typically carried out by the *softmax*, this is mapping logits to probability vectors. However, as is the main claim in the original work [1], calibration can be a complex task, motivating the use of complex models. In order to avoid the typical overconfident behavior of Neural Networks, we take a Bayesian approach. This is, we do not train the weights $\boldsymbol{\theta}$ to a point estimate, instead, we infer the posterior $q^*(\boldsymbol{\theta})$. Ideally, we would compute the standard Bayesian posterior with the Bayes rule:

$$q_B^*(\boldsymbol{\theta}) = \frac{\pi(\boldsymbol{\theta}) \prod_{n=1}^N p((\hat{\mathbf{z}}_n, \mathbf{y}_n)|\boldsymbol{\theta})}{\int \pi(\boldsymbol{\theta}) \prod_{n=1}^N p((\hat{\mathbf{z}}_n, \mathbf{y}_n)|\boldsymbol{\theta}) d\boldsymbol{\theta}}$$

But there are two main problems with this approach. Most importantly, the integral in the denominator is intractable; hence, exact computation is impossible. Even if we were able to compute it, the inherent complexity of

the model means that both, the likelihood and the prior, violate the fundamental assumptions of the Bayesian paradigm as remarked in Chapter 3. These reasons motivate the application of Generalized Variational Inference. First, it provides a framework to build posteriors in a tractable way, and second, it does it accounting for possible misspecifications.

Making predictions

In order to make predictions, i.e. compute the calibrated probability vector $\hat{\mathbf{q}}$ given its associated uncalibrated logit $\hat{\mathbf{z}}$, we need to compute the expectation:

$$\hat{\mathbf{q}} = \int f(\hat{\mathbf{z}}|\boldsymbol{\theta}) q^*(\boldsymbol{\theta}) d\boldsymbol{\theta},$$

where $f(\hat{\mathbf{z}}|\boldsymbol{\theta})$ is the output of the MLP setting its weights to $\boldsymbol{\theta}$. Thus, the prediction can be seen as the average over all the possible outcomes of the MLP weigthed by the posterior probability of each $\boldsymbol{\theta}$. Again, this integral is, in general, intractable. So, we approximate it taking a Monte Carlo estimator:

$$\hat{\mathbf{q}} \simeq \frac{1}{K} \sum_{k=1}^{K} f(\hat{\mathbf{z}}|\boldsymbol{\theta}_k);\ \boldsymbol{\theta}_k \sim q^*(\boldsymbol{\theta}),$$

where K is the number of Monte Carlo (MC) samples.

4.2 Approximating the Posterior with GVI

The GVI framework constructs an optimization problem which solution is a, hopefully, useful posterior. To do so, GVI requires to specify a variational family Π, a loss function l, and a divergence D as defined in the RoT (which implies specifying the prior π too). For reasons that we explain bellow, we maintain the traditional negative log-likelihood as loss function and focus only on the variational family, the prior distribution, and the divergence.

Variational Family and Prior Distribution

We take a Mean Field approach and restrict the posterior to be a factorized multivariate Gaussian, i.e. $\boldsymbol{\theta} \sim \mathcal{N}(\boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}))$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are the mean and variance vectors of the posterior distribution and $\text{diag}(\boldsymbol{\sigma})$ is the square diagonal matrix whose non-zero entries are the elements of the vector $\boldsymbol{\sigma}$. Similarly we set the prior to the standard Gaussian. This is a convenient choice in the computational sense, as it only doubles the number of parameters (a mean μ and a variance σ^2 for each weight). Moreover, the D_{KL} is available in closed form solution, avoiding the need to estimate it via Monte Carlo.

Despite its popularity, this choice of distributions might be not so convenient for computing useful posteriors. The prior is clearly misspecified, we have observed that posteriors close to it output very uniform, thus uncertain, class. predictions (which may be well calibrated but offer no discrimination performance at all, i.e. very low accuracy, close to 1 over the number of classes). On the other hand, this prior is the only source of uncertainty for the posterior, which is an unimodal distribution, this means that down-weighting the prior importance would lead to peaked posteriors around the point-estimate solution, loosing the uncertainty modeling of the Bayesian approach [2].

Divergence

Motivated by this and the theoretical properties of GVI, we propose to use a robust divergence, specifically the α-Rényi divergence. We expect to obtain posteriors that maintain an adequately big variance but deviate enough from the prior so as to make useful predictions. This is, we expect that our method yield predictions that are both, well calibrated and accurate.

The previous approach overcome this limitation by capping the number of MC samples used when sampling from the posterior. This has the effect of virtually augmenting the variance of the posterior, and thus forcing uncertainty modeling. Thus, the number of MC samples, K, had to be estimated in a validation set. We hypothesize that with GVI we will obtain posteriors that already present an adequate variance without the need of tuning the posterior by selecting an optimum K, i.e. posteriors will retain good calibration properties as $K \to \infty$.

The α-Rényi divergence is available in closed form for the multivariate Gaussian, although it is only guaranteed to exist for $0 < \alpha < 1$. For greater values of α the variance of the prior must be much larger than the posterior variance in order to be defined. Since this is a novel framework, there is no much literature on the choice of the hyperparameter α. Therefore, in this work we contribute with a sensitivity study of this parameter basing our range of search on the results given in [2].

4.3 Implementation

The GVI give us a way to construct a posterior via an optimization problem, taking the choices made in the previous Section, this is given by:

$$q^*(\boldsymbol{\theta}) = \underset{q \in \Pi}{\arg\min} \left\{ \mathbb{E}_q \left[\sum_{n=1}^{N} l(\boldsymbol{\theta}, \hat{\mathbf{z}}_n, \mathbf{y}_n) \right] + D_{\mathrm{AR}}^{(\alpha)}\big(q(\boldsymbol{\theta}) \,||\, \pi(\boldsymbol{\theta})\big) \right\}, \tag{4.1}$$

where Π is the family of factorized multivariate Gaussians, and $l(\boldsymbol{\theta}, \hat{\mathbf{z}}_n, \mathbf{y}_n)$ is the NLL. We refer to q and to q^* as the posterior the same way we refer

to f and $f*$ as the model, regardless of whether it has already been fitted or not. First, we initialize the posterior parameters, and the we optimize them to approximate the "true" posterior.

The Reparametrization Trick

To solve this optimization problem we may use a gradient-based method like stochastic gradient descent, but we need to make some tweaks before we can actually apply it. To approximate the expectation of the first term we use a Monte Carlo estimator sampling from the posterior and evaluating the NLL for multiple samples:

$$\mathbb{E}_q\left[\sum_{n=1}^{N} l(\boldsymbol{\theta}, \hat{\mathbf{z}}_n, \mathbf{y}_n)\right] \simeq \frac{1}{K}\sum_{k=1}^{K}\left[\sum_{n=1}^{N} l(\boldsymbol{\theta}_k, \hat{\mathbf{z}}_n, \mathbf{y}_n)\right];\ \boldsymbol{\theta}_k \sim q(\boldsymbol{\theta}), \tag{4.2}$$

where K is the number of MC samples.

Even after making this change, a gradient-based method cannot be directly applied since the gradient can not be propagated through weight samples. Instead we use the reparametrization trick [33].

The reparametrization trick allows to detach the stochasticity of the sampling from the distribution parameters. If one desires to propagate gradients throughout the expected value of some function $f(x)$ over some distribution $p(x)$, take for instance the following case:

$$\nabla_\mu \mathbb{E}_{p(x)}\left[f(x)\right];\ \ x \sim \mathcal{N}(\mu, 1),$$

one may find that there is no analytic solution, since μ is a parameter of $p(x)$. The reparametrization trick defines x as a function of a base distribution, which accounts for the randomness, and the distribution parameters. In the example above:

$$\nabla_\mu \mathbb{E}_{p(z)}\left[f(x)\right] = \mathbb{E}_{p(z)}\left[\nabla_\mu f(x)\right];\ \ x = z + \mu;\ \ z \sim \mathcal{N}(0, 1),$$

so the expectation does not depend on μ, and by linearity of the expectation and the differentiation, the gradient can be taken into the expectation.

More precisely, we use the local reparametrization trick [34], which recognizes that if the parameters $\theta_l = \{W^l, b^l\}$ of a FC layer follow a Gaussian distribution, then the activations $\mathbf{a}^l = W^l\mathbf{x}^{(l-1)} + b^l$ also follow a Gaussian distribution whose parameters can be obtained from those of θ_l. This method reduces the variance of gradients by sampling directly from the activations, instead of sampling from the weights, improving the convergence rate. We always use the local reparametrization trick, in contrast to the previous work where two different implementations are tested, one using the local reparametrization trick and the other the plain reparametrization trick.

Mini-Batch Scaling

It is common practice to estimate gradients with just a subset, or mini-batch, of the whole dataset. Training with mini-batches often improves the convergence rate of the optimization problem and we use it in this work. Notice that the magnitude of the loss estimation and the size of the mini-batch M are directly proportional. If we were seeking to only optimize the loss this would not be an issue since any constant scaling preserves the gradient direction and the step size can be covered by the learning rate. However, our optimization objective presents another term, the divergence, which scale is invariant to the size of the mini-batch. To compensate for this effect we need to rescale the estimate for the loss. Let $\mathcal{MB}$ be the set of indexes of the samples in the mini-batch. We compute the loss estimate as:

$$\mathbb{E}_q\left[\sum_{n=1}^{N} l(\boldsymbol{\theta}, \hat{\mathbf{z}}_n, \mathbf{y}_n)\right] \simeq \frac{1}{K}\sum_{k=1}^{K}\left[\frac{N}{M}\sum_{m\in\mathcal{MB}} l(\boldsymbol{\theta}_k, \hat{\mathbf{z}}_m, \mathbf{y}_m)\right];\ \boldsymbol{\theta}_k \sim q(\boldsymbol{\theta}),$$

where the sampling from the posterior is done with the local reparametrization trick (sampling activations instead of weights), but we maintain this notation for the sake of simplicity.

Finally, we implement the optimization problem as:

$$q^* = \arg\min_{q\in\Pi}\left\{\frac{1}{K}\sum_{k=1}^{K}\left[\frac{N}{M}\sum_{m\in\mathcal{MB}} l(\boldsymbol{\theta}_k, \hat{\mathbf{z}}_m, \mathbf{y}_m)\right] + D_{\mathrm{AR}}^{(\alpha)}\big(q\,||\,\pi\big)\right\};\ \boldsymbol{\theta}_k \sim q, \tag{4.3}$$

where we have omitted the dependence with $\boldsymbol{\theta}$ to avoid confusion with dependence with $\boldsymbol{\theta}_k$

Chapter 5

Experiments

We extend the work in [1]. There, the authors provide a vast range of experiments in which they calibrate multiple CNNs in different image classification datasets. Because of the novelty of GVI, there is not much literature in the application of robust divergences to the task. To this aim, we select a small subset of the experiments in the previous work and make a comprehensive analysis on those. In particular we provide a sensitivity study on the influence of the β and α hyperparameters in the performance of the KL and Rényi's α-divergence respectively. Then, we compare the Rényi's α-divergence with the traditional KL.

Throughout this section we show aggregated results in the form of tables and present some examples in the form of figures to help visualization. These comprise the most relevant results and explain all the conclusions and observations that we make. However, for the interested reader we include the whole bulk of results in the form of tables arranged in the three appendices of this work, but we remark that these are not necessary to follow this work and that all necessary information is included on the main body.

5.1 Set-Up

We have restricted the experiments to the calibration of Densenet-121 [20] in the CIFAR10 and CIFAR100 datasets [45], and the CARS dataset [46]. Each dataset is divided into a train/validation/test split. The Densenet-121 is trained on the train split and then it is used to compute predictions on the three splits. Our training data is, then, composed by the output logits of Densenet-121 for each of the three datasets and divided, in each case, into the three splits train/validation/test.

Following the procedure of the original work, we train a BNN on the train set, select K^* on the validation set as the number of predictive samples obtaining the lowest ECE score, and report results on the test set (see Figure 5.1). Additionally, we perform experiments ignoring the original train set

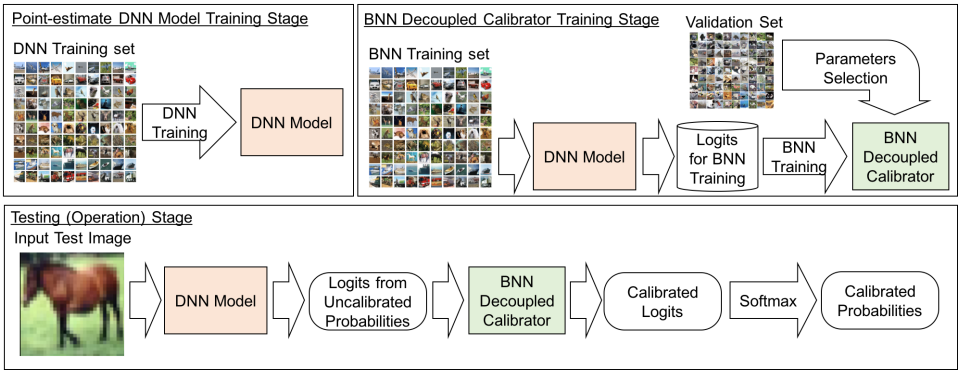

Figure 5.1: Description of the training, validation, and testing stages. Figure taken from [1].

and using the validation set for training. The outline of the two different procedures is:

1. Train on the train set, select optimum K^* on the validation test, and report results on the test set.

2. Discard the training set, split the validation set randomly in a ratio 80/20, 80% for a new train set and 20% for new validation, then apply the first procedure but with the new data splits.

We try different topologies, number of layers and hidden units, for the BNNs and multiple values of β in the optimization objective:

$$q^* = \arg\min_{q \in \Pi} \left\{ \frac{1}{K} \sum_{k=1}^{K} \left[\frac{N}{M} \sum_{m \in \mathcal{MB}} l(\boldsymbol{\theta}_k, \hat{\mathbf{z}}_m, \mathbf{y}_m) \right] + \beta D \right\}; \; \boldsymbol{\theta}_k \sim q, \qquad (5.1)$$

where D originally was the KL divergence but we generalize in this work to other divergences (KL and Rényi's α). We use a different parametrization than that in the original work since there the re-scaling of the loss estimator (see Section 4.3) is subsumed by the β parameter. We, in contrast, scale the estimator so as $\beta = 1$ corresponds to not scaling the divergence at all recovering the original objective.

Initialization of parameters to be optimized

All the parameters, namely the means and logarithmic variances of the weights, are initialized randomly from a standard normal distribution, except when using the Rényi's α-divergence. As we mention in Chapter 3, the divergence is only guaranteed to exist in closed form for $0 < \alpha < 1$. For greater values of α, the prior variance must be much larger than the posterior variance. To this aim we initialize the log-variance of the weights to

$\log \sigma^2 = 1e{-}10$. This initialization scheme is replicated from the one used in [2] and ensures a stable training:

- When $\alpha > 1$ more weight is placed on the posterior (see Equation 3.5). This favors the *zero forcing* property, i.e. the variance under-estimation characteristic of standard VI (see Section 3.1.1), and keeps posterior variances small enough.
- When $0 < \alpha < 1$ the divergence is guaranteed to exist regardless of the relative size of the prior and posterior variances.

On the other hand, the variance under-estimation is one of the VI drawbacks we want to avoid the in first place. However, the toy experiments in [2] show that the best performing values of α lay in the range $1 < \alpha \leq 2.5$; hence, we contemplate values of $\alpha > 1$ in this work as well as values of $0 < \alpha < 1$.

Training

We use the Adam optimizer [47] with a learning rate of 0.01 and a batch size of 100, we train for 1000 epochs when using the KL divergence and for 2000 epochs when using the Rényi's α-divergence as it shows a slower convergence rate. We use this configuration regardless of whether we train on the train set or on the validation set.

Measuring Performance

We focus primarily on two different metrics. ECE as a measure of calibration, and accuracy to monitor discrimination. A BNN learning to predict the empirical class prior, which in this case is uniform, would achieve an almost-perfect ECE score, but the accuracy will drop to one over the number of classes. Conversely, the goal is to improve calibration without giving away the good accuracy of Densenet-121. We report performance measures taking K^* predictive samples on the test set, where K^* has been selected in the validation set as described above. In addition, we also report results using $K = 20000$ to observe the aforementioned ill behaviour of variance under-estimation when $K \to \infty$.

Visualization

We use bar plots to visualize the accuracy and the ECE for multiple values of β or α depending on divergence used in each experiment. We use different bar colors to indicate the data split on which the model was trained, blue for the train set (namely "trained on train") and orange for the validation set (namely "trained on train")(see Figure 5.2, described bellow). We compare with the uncalibrated network (blue dashed line) and with Temp-Scaling

(red dashed line). Notice that Temp-Scaling does not change accuracy so we only use Temp-Scaling as reference for performance results with accuracy. We indicate the topology of a network between square brackets, for instance, [32-32] refer to a model with 2 hidden layers of 32 units each and [128] to a single hidden-layer net with 128 units.

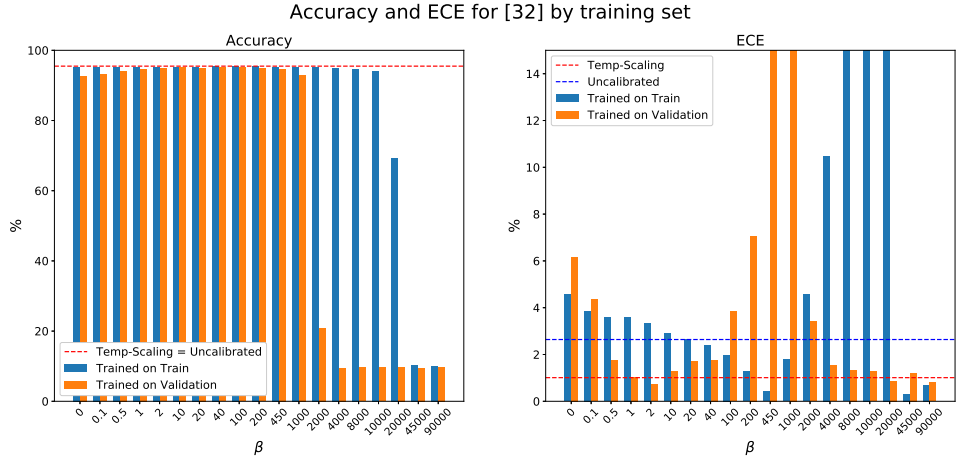

Figure 5.2: Test set performance of a single 32-unit (indicated between brackets in the title) hidden-layer BNN calibrating CIFAR10. Accuracy(left) and ECE(right) in the CIFAR10 test set after calibrating with multiple values of β and different training sets.

In the last section, when comparing divergences, we change the color scheme to avoid confusion because they will indicate the number of MC predictive samples, purple means best K on validation and brown $K = 20000$. The comparison is done against Temp-Scaling and the KL divergence based method (see Figure 5.5, described bellow).

5.2 Sensitivity Study

In this section we describe the experiments related to the sensitivity study of the hyperparameters α and β using the Rényi's α-divergence and the KL divergence respectively. We refer to sensitivity study as the analysis of how some hyperparameter impacts the test performance. First we train several models with the KL divergence varying, in each case, the value of β. This way we can relate changes in test performance to changes in β as we can fix any of the tested configurations of topology and training set (train or validation) and check results for different values of β. We, then, repeat a similar procedure but training with the Rényi's α-divergence and varying the α parameter.

5.2.1 KL divergence

We first try several configurations using the KL divergence, varying the topology of the net and the β parameter in the range $[0, 90000]$. Due to the differences in dimensionality between CIFAR10, on the one side, and CIFAR100 and CARS, on the other, we use a different combination of architectures in the smaller CIFAR10. We try the following configurations:

- **CIFAR10:** A 0-hidden layer model, a 1-hidden layer model with 32 units, and two 2-hidden layers models with 32 and 512 hidden units.
- **CIFAR100 and CARS:** A 0-hidden layer model, and models with 1 and 2 hidden layers with 128, 512, 1024, and 2048 units.

Influence of the dataset size

The first thing we notice when we compare the results after training on the train set with those after training on the validation set is that the influence of β is directly related to the size of the training set. This can be easily observed in Figure 5.2, specially in the accuracy plot, where degradation begins for higher values of β in the trained on train scenario; on the ECE plot we can also note how results after training on the train set with high values of β present a similar behaviour to those after training on the validation set with low values of β. The difference in β values is about x10, the same difference in size between the train and validation sets. This result is no surprise, the additive nature of the likelihood term in Equation 5.1 implies that its magnitude scales linearly with the number of samples. This difference in magnitude is compensated with a change in value of β since the divergence does not depend on the dataset size. Another way of seeing this is that, as the number of samples increases so does the evidence reinforcing the confidence of the model; hence, one must increase the weight of the divergence in order to achieve the same level of uncertainty.

Dimensionality

It is also worth noting that calibrating CIFAR10 seems to be a much easier task than calibrating CIFAR100 and CARS. In the first case almost all configurations of topology and β manage to obtain high accuracy, i.e. succeeds to minimize the NLL (see Table 5.1). Notice that we refer to succeded experiments as those obtaining relatively high accuracy despite its ECE score; hence, we do not mean that they report state-of-the-art calibration properties, just that this configuration recovers discrimination performance. In the latter cases, we find more settings in which the BNN is not able to recover any discrimination performance at all (see Table 5.2). Furthermore, almost every configuration with 2-hidden gets the minimum of the cost by obtaining

a posterior that outputs the prior distribution of the classes, obtaining vary low ECE but also very low accuracy.

Table 5.1: Proportion of succeeded experiments in CIFAR10 training with the KL divergence.

Topology	Succeed (%)
[-]	78.95
[32]	73.68
[32-32]	39.47
[512-512]	89.47

Table 5.2: Proportion of succeeded experiments in CIFAR100 and CARS training with the KL divergence.

Topology	Succeed (%)	
	CIFAR100	CARS
[-]	55.26	36.84
[128]	57.89	0.00
[128-128]	0.00	0.00
[512]	57.89	36.84
[512-512]	0.00	0.00
[1024]	57.89	47.37
[1024-1024]	42.11	0.00
[2048]	60.53	47.37
[2048-2048]	44.74	0.00

Proportion of succeeded experiments when training with the KL divergence. We consider an experiment has succeed when accuracy degradation is no more than 20% despite its ECE score. Proportions are computed by averaging over all configurations of β and train set (on Train or on Validation).

Number of MC samples

We notice that the number of predictive MC samples K has, in general, the conjectured behaviour. The higher the K the higher the ECE and the accuracy. This is because as $K \to \infty$ the Monte Carlo estimator converges to the expected value over the posterior, which, as we state above, underestimates the variance and concentrates around a delta at a local optima. This produces over-confident predictions degrading the ECE score. We show some examples in Table 5.3, where we can see the differences between using a high number of MC samples ($K = 20000$) and choosing the optimum value in a validation set.

5.2.2 Rényi's α-divergence

We perform similar experiments but using the Rényi's α-divergence instead of the KL and setting $\beta = 1$, this way we do not weight the robust divergence and focus exclusively on the influence of they hyperparameter α. We use the same configurations of topology and training set as in the previous

Configuration	Accuracy (best K)	Accuracy ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)
CIFAR100 Trained on Train, [512]	56.29	58.12	9.14	11.42
CIFAR100 Trained on Val, [512]	39.44	41.13	7.65	9.64
CIFAR100 Trained on Train, [2048]	56.81	59.08	9.47	12.37
CIFAR100 Trained on Val, [2048]	40.26	42.20	7.67	10.01
CARS Trained on Train, [1024]	51.57	54.27	5.84	9.12
CARS Trained on Val, [1024]	38.45	41.20	4.44	7.76
CIFAR10 Trained on Train, [32]	84.88	86.06	6.77	8.23
CIFAR10 Trained on Val, [32]	84.88	86.06	6.77	8.23

Table 5.3: Influence of the number of MC samples. Metrics are computed averaging over all values of β, *Train* or *Val* refers to the set in which the BNN was trained, and the topology is specified between square brackets.

section and vary the value of α in the range $[0.1, 2.5]$, this choice of values is motivated by the configurations used in [2].

The typical behaviour across experiments is that of using a low value of β in the previous set of experiments, high accuracy and ECE. Most of the experiments succeed in the sense that they recover discrimination performance (see Table 5.4 and Table 5.5). However, in most settings the model becomes even more over-confident after the calibration (see Figure 5.3 for an example).

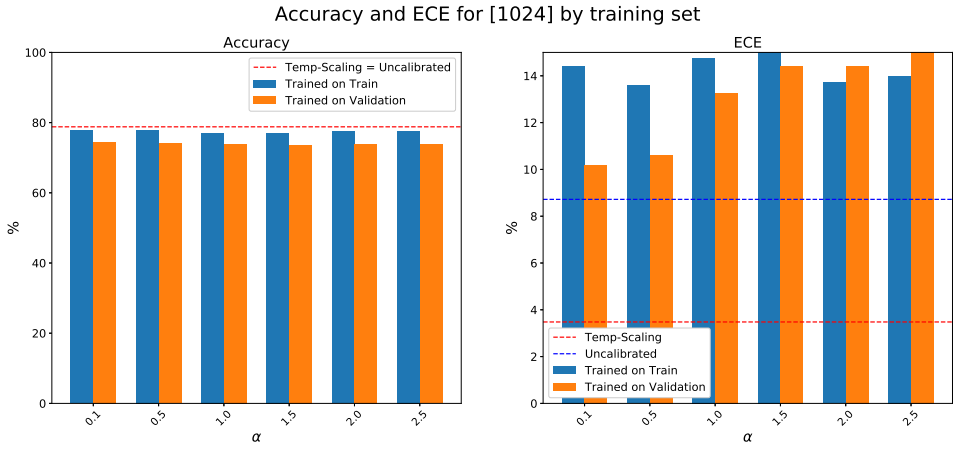

Figure 5.3: Test set performance of a one 1024-unit hidden-layer (indicated between brackets in the title) BNN calibrating CIFAR100. Accuracy(left) and ECE(right) in the CIFAR100 test set after calibrating with multiple values of α and different training sets.

Although this is not always the case, some configurations seem to perform reasonable well, for instance in Figure 5.4 all configurations trained on the validation set present good performance. In general, it seems that

Table 5.4: Proportion of succeeded experiments in CIFAR10 training with the Rényi's α-divergence.

Topology	Succeed (%)
[-]	100.00
[32]	100.00
[32-32]	100.00
[512-512]	100.00

Table 5.5: Proportion of succeeded experiments in CIFAR100 and CARS training with the Rényi's α-divergence.

Topology	Succeed (%)	
	CIFAR100	CARS
[-]	100.00	100.00
[128]	91.67	58.33
[128-128]	66.67	0.00
[512]	100.00	100.00
[512-512]	83.33	83.33
[1024]	100.00	100.00
[1024-1024]	91.67	83.33
[2048]	100.00	100.00
[2048-2048]	91.67	83.33

Proportion of succeeded experiments when training with the Rényi's α-divergence. We consider an experiment has succeed when accuracy degradation is no more than 20% despite its ECE score. Proportions are computed by averaging over all configurations of β and train set (on Train or on Validation).

$\alpha = 0.5$ is the best performing setting across experiments.

The fact that most experiments yield high accuracy but overconfident predictions means that we may not be introducing enough uncertainty with the divergence term. To this aim a similar approach to the weighted KL can be taken, properly addressing the prior misspecification with the introduction of a robust divergence but also accounting for greater uncertainty quantification, i.e. $\beta > 1$.

5.3 Comparison

One way of interpreting β is that of uncertainty quantifier. This is, we can control how much uncertainty does the divergence introduce in the posterior by changing the value of β. The results of the first round of experiments show that careful tuning of this parameter allows for good uncertainty modeling without high accuracy degradation. We can apply this balancing effect to the Rényi's α-divergence as well. The purpose of this set of experiments is that of comparing both divergences, the KL and the Rényi's α-divergence, in a controlled set of configurations. In this case the configuration encompass the topology, the training set (on train or on validation), and the uncertainty quantifier β as well, so we can compare both divergences for the same level

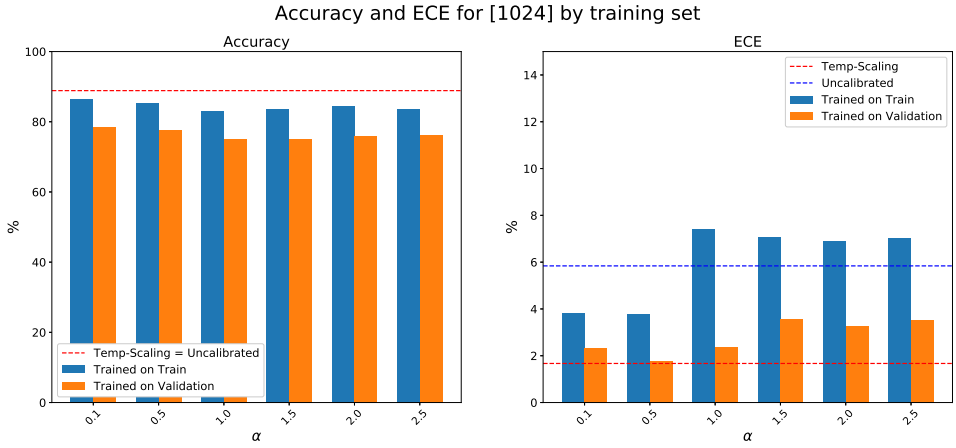

Figure 5.4: Test set performance of a one 1024-unit hidden-layer (indicated between brackets in the title) BNN calibrating CARS. Accuracy(left) and ECE(right) in the CARS test set after calibrating with multiple values of α and different training sets.

of *uncertainty quantification*.

Representative Cases

Form the first round of experiments we select a subset of configurations in which compare both divergences. To this aim we select experiments on the three datasets, including configurations where the method in [1], i.e. training with the weighted KL, provides state-of-the-art calibration performance, but also vicious configurations where it presents some performance degradation, either in accuracy, in ECE, or in both. All selected configurations where trained on the train set. We show these in Table 5.6, top-performing configurations are displayed in the third row and vicious configurations in the last row. In addition, to measure sensibility to the β hyperparameter we select configurations with close values of β to those that report state-of-the-art performance.

Results

In settings where the KL presents state-of-the-art performance we see that the $D_{\mathrm{AR}}^{(\alpha)}$ reports close performance (see Figure 5.5) and that this performance presents low degradation with the increment of K (brown against purple bars). In cases where the KL does not perform so well (vicious configurations), the $D_{\mathrm{AR}}^{(\alpha)}$ is able to improve performance. We can observe this in Figure 5.6, where in most settings the robust divergence yields posteriors with both, lower accuracy degradation and better ECE. In this case

CIFAR10			CIFAR100			CARS		
Model	Accuracy	ECE	Model	Accuracy	ECE	Model	Accuracy	ECE
Uncalibrated	95.49	2.64	Uncalibrated	78.80	8.72	Uncalibrated	88.87	5.84
Temp-Scaling	95.49	1.01	Temp-Scaling	78.80	3.48	Temp-Scaling	88.87	1.67
$\beta = 200$, [32]	95.47	1.27	$\beta = 100$, [128]	77.36	3.47	$\beta = 10$, [512]	82.47	3.43
$\beta = 450$, [32]	95.32	0.42	$\beta = 200$, [128]	77.21	1.23	$\beta = 20$, [512]	82.34	1.66
$\beta = 1000$, [32]	95.3	1.80	$\beta = 450$, [128]	75.89	4.97	$\beta = 40$, [512]	82.06	2.20
$\beta = 1000$, [-]	95.07	3.66	$\beta = 2000$, [512]	72.21	19.38	$\beta = 450$, [2048]	73.79	14.57

Table 5.6: Selected configurations for divergence comparison, the topology is specified between square brackets and all configurations were trained on the train set. We include as reference the performance of the uncalibrated network and after Temp-Scaling.

we notice some degradation in the ECE score as we use a high number of MC samples, but the increment in accuracy performance greatly compensates this supporting a higher value of K. Having that a higher value of K is desired means that the constructed posterior is well suited since is this posterior we want to achieve when increasing K.

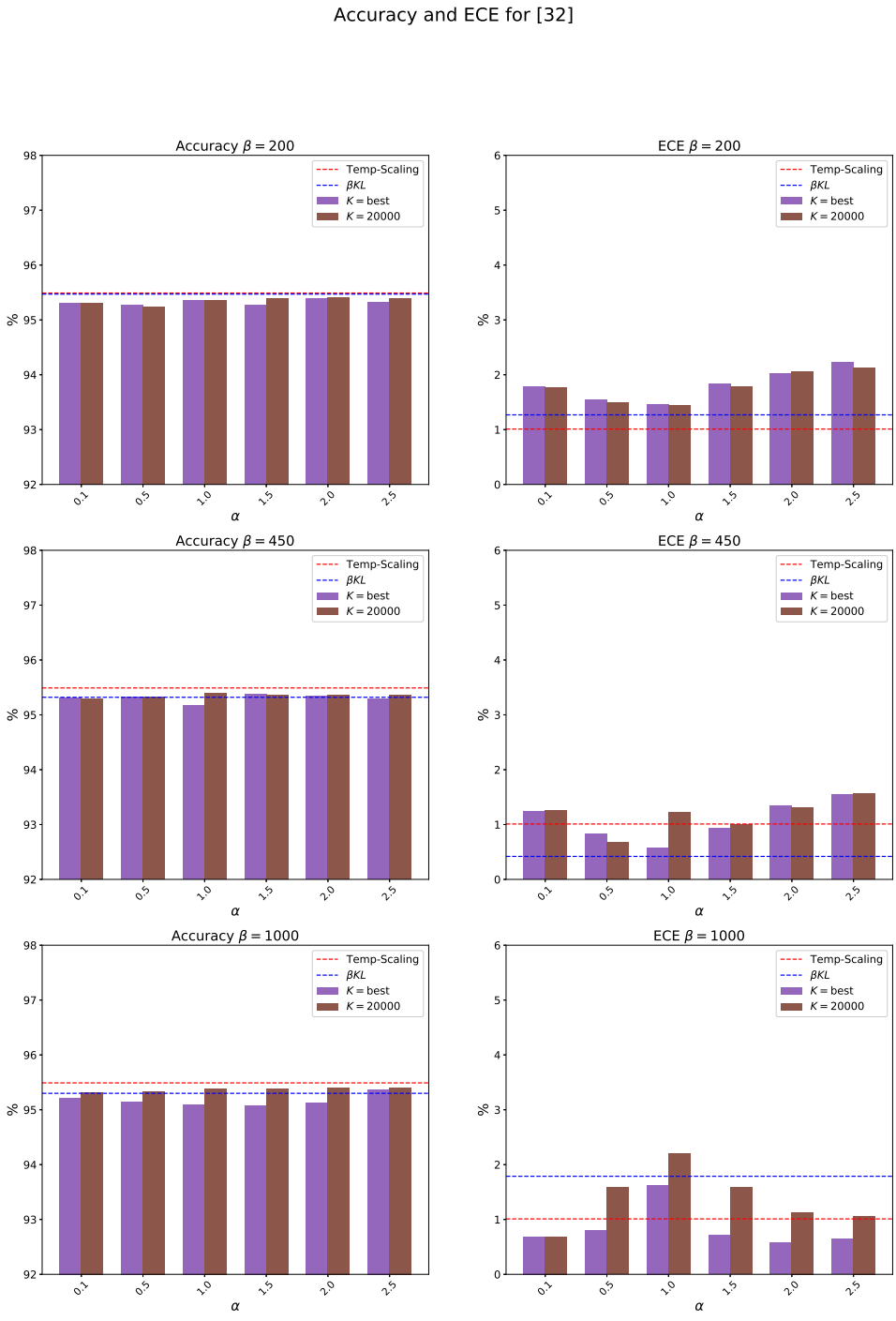

Figure 5.5: Performance of BNNs trained with $D_{\text{AR}}^{(\alpha)}$ in settings where KL reports state-of-the-art results. Accuracy(left) and ECE(right) in the CIFAR10 test set after calibrating with multiple values of α.

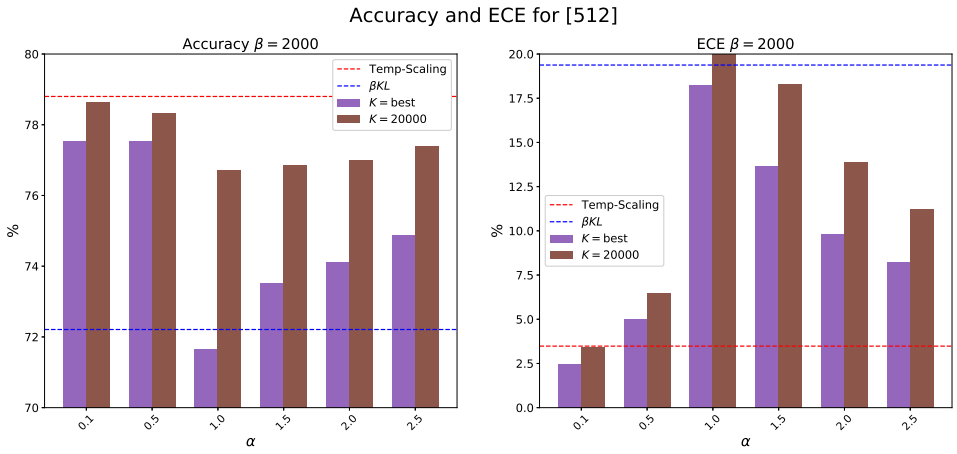

Figure 5.6: Peformance of BNNs trained with $D_{\text{AR}}^{(\alpha)}$ in settings where KL reports moderate results. Accuracy(left) and ECE(right) in the CIFAR100 test set after calibrating with multiple values of α.

Chapter 6

Conclusions and Future Work

In this work we show that the novel framework of GVI can be applied to the task of calibration with Bayesian Neural Networks in order to obtain better posteriors.

We obtain an easier way to incorporate uncertainty in the posterior by using a weighted robust divergence. When compared to the traditional KL, the calibration performance is less sensitive to hyperparameter configuration, providing good calibration properties for a wider range of β. This reinforce the hypothesis that the prior misspecefication characteristic of modern approximations vitiates the Bayesian approach and that such violations shall be taken into consideration. Moreover, the modularity of the GVI seems to provide an effective way of building posteriors that account for such violations.

Future Work

In this work, we only tackle prior misspecification and try just one robust divergence obtaining promising results. This motivates as future work the use of other robust divergences and addressing likelihood misspecification via different loss functions, further exploiting the versatility of GVI.

Likelihood misspecification can also be addressed by assuming a different posterior family. To this aim more flexible posteriors can be employed like Rank-1 Factors BNNs [29] or Multiplicative Normalizing Flows [48].

Likewise we can revise the choice of prior. One way of doing this is by learning useful prior distributions with the Empirical Bayes method [49].

In addition, it would be interesting to evaluate the performance of this method, Decoupled Bayesian Neural Networks, on others datasets like BIRDS [50] and ADIENCE [51], as the authors do in [1].

Bibliography

[1] Juan Maroñas, Roberto Paredes, and Daniel Ramos. "Calibration of Deep Probabilistic Models with Decoupled Bayesian Neural Networks". In: (2019). arXiv: 1908.08972.

[2] Jeremias Knoblauch, Jack Jewson, and Theodoros Damoulas. "Generalized Variational Inference: Three arguments for deriving new Posteriors". In: (2019). arXiv: 1904.02063.

[3] Olga Russakovsky et al. "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3 (2015), pp. 211–252. DOI: 10.1007/s11263-015-0816-y.

[4] K. Xu et al. "Show, attend and tell: Neural image caption generation with visual attention". In: *32nd International Conference on Machine Learning, ICML 2015*. Vol. 3. 2015, pp. 2048–2057. ISBN: 9781510810587.

[5] A. Graves, A.-R. Mohamed, and G. Hinton. "Speech recognition with deep recurrent neural networks". In: *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*. 2013, pp. 6645–6649. ISBN: 9781479903566. DOI: 10.1109/ICASSP.2013.6638947.

[6] Chuan Guo et al. "On calibration of modern neural networks". In: *34th International Conference on Machine Learning, ICML 2017* 3 (2017), pp. 2130–2143. arXiv: 1706.04599.

[7] Allan H. Murphy and Edward S. Epstein. "Verification of Probabilistic Predictions: A Brief Review". In: *Journal of Applied Meteorology* 6.5 (Oct. 1967), pp. 748–755. ISSN: 0021-8952. DOI: 10.1175/1520-0450(1967)006<0748:voppab>2.0.co;2.

[8] Morris H DeGroot and Stephen E Fienberg. "The Comparison and Evaluation of Forecasters". In: *Journal of the Royal Statistical Society. Series D (The Statistician)* 32.1/2 (1983), pp. 12–22. ISSN: 00390526, 14679884.

[9] John Platt et al. "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods". In: *Advances in large margin classifiers* 10.3 (1999), pp. 61–74.

[10] Bianca Zadrozny and Charles Elkan. "Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers". In: *Icml* (2001), pp. 1–8.

[11] Dan Hendrycks et al. "AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty". In: (2019). arXiv: 1912.02781.

[12] Yarin Gal and Zoubin Ghahramani. "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning". In: *33rd International Conference on Machine Learning, ICML 2016*. Vol. 3. ICML'16. JMLR.org, 2016, pp. 1651–1660. ISBN: 9781510829008. arXiv: 1506.02142.

[13] J A Nelder and R W M Wedderburn. "Generalized Linear Models". In: *Journal of the Royal Statistical Society. Series A (General)* 135.3 (June 1972), pp. 370–384. ISSN: 00359238. DOI: 10.2307/2344614.

[14] Henry J Kelley. "Gradient theory of optimal flight paths". In: *Ars Journal* 30.10 (1960), pp. 947–954.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks". In: *Communications of the ACM* 60.6 (2017), pp. 84–90. ISSN: 15577317. DOI: 10.1145/3065386.

[16] David E Rumelhart and James L McClelland. "Learning Internal Representations by Error Propagation". In: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*. MITP, 1987, pp. 318–362. ISBN: 9780262291408.

[17] Ashish Vaswani et al. "Attention is all you need". In: *Advances in Neural Information Processing Systems* 2017-December (June 2017), pp. 5999–6009. ISSN: 10495258. arXiv: 1706.03762.

[18] Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. "Neural machine translation by jointly learning to align and translate". In: *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. 2015. arXiv: 1409.0473.

[19] Kaiming He et al. "Deep residual learning for image recognition". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* 2016-December (Dec. 2016), pp. 770–778. ISSN: 10636919. DOI: 10.1109/CVPR.2016.90. arXiv: 1512.03385.

[20] Gao Huang et al. "Densely connected convolutional networks". In: *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017* 2017-January (Aug. 2017), pp. 2261–2269. DOI: 10.1109/CVPR.2017.243. arXiv: 1608.06993.

[21] Jeremy Nixon et al. *Measuring Calibration in Deep Learning*. Apr. 2019. arXiv: 1904.01685.

[22] Ananya Kumar, Percy Liang, and Tengyu Ma. *Verified Uncertainty Calibration*. Sept. 2019. arXiv: 1909.10155.

[23] Niko Brümmer and Johan Adam Du Preez. "Measuring, refining and calibrating speaker and language information extracted from speech". PhD thesis. Stellenbosch : University of Stellenbosch, 2010.

[24] Daniel Ramos et al. "Deconstructing cross-entropy for probabilistic binary classifiers". In: *Entropy* 20.3 (2018). ISSN: 10994300. DOI: 10.3390/e20030208.

[25] Sunil Thulasidasan et al. *On Mixup Training: Improved Calibration and Predictive Uncertainty for Deep Neural Networks*. 2019. arXiv: 1905.11001.

[26] Yixin Wang, Alp Kucukelbir, and David M Blei. "Robust probabilistic modeling with Bayesian data reweighting". In: *34th International Conference on Machine Learning, ICML 2017*. Vol. 7. 2017, pp. 5564–5589. ISBN: 9781510855144. arXiv: 1606.03860.

[27] Gabriel Pereyra et al. "Regularizing neural networks by penalizing confident output distributions". In: *5th International Conference on Learning Representations, ICLR 2017 - Workshop Track Proceedings* (Jan. 2019). arXiv: 1701.06548.

[28] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. "Simple and scalable predictive uncertainty estimation using deep ensembles". In: *Advances in Neural Information Processing Systems*. Vol. 2017-December. Neural information processing systems foundation, 2017, pp. 6403–6414. arXiv: 1612.01474.

[29] Michael W. Dusenberry et al. "Efficient and Scalable Bayesian Neural Nets with Rank-1 Factors". In: (2020). arXiv: 2005.07186.

[30] Bianca Zadrozny and Charles Elkan. *Transforming classifier scores into accurate multiclass probability estimates*. Tech. rep. 2002, pp. 694–699. DOI: 10.1145/775107.775151.

[31] Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. "Obtaining well calibrated probabilities using Bayesian Binning". In: *Proceedings of the National Conference on Artificial Intelligence* 4 (2015), pp. 2901–2907. ISSN: 2159-5399.

[32] Yaniv Ovadia et al. "Can You Trust Your Model's Uncertainty? Evaluating Predictive Uncertainty Under Dataset Shift". In: (June 2019). arXiv: 1906.02530.

[33] Diederik P Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: (Dec. 2013). arXiv: 1312.6114.

[34] Diederik P. Kingma, Tim Salimans, and Max Welling. "Variational dropout and the local reparameterization trick". In: *Advances in Neural Information Processing Systems*. Vol. 2015-January. Neural information processing systems foundation, 2015, pp. 2575–2583. arXiv: 1506.02557.

[35] Christopher M. Bishop. "Machine Learning and Pattern Recoginiton". In: *Information Science and Statistics*. Springer-Verlag New York, 2006. Chap. 10. ISBN: 9780387310732.

[36] George E P Box. "Sampling and Bayes' Inference in Scientific Modelling and Robustness". In: *Journal of the Royal Statistical Society. Series A (General)* 143.4 (June 1980), pp. 383–430. ISSN: 00359238. DOI: 10.2307/2982063.

[37] I Csiszar. "I-Divergence Geometry of Probability Distributions and Minimization Problems". In: *The Annals of Probability* 3.1 (1975), pp. 146–158. ISSN: 00911798.

[38] Arnold Zellner. "Optimal Information Processing and Bayes's Theorem". In: *The American Statistician* 42.4 (June 1988), pp. 278–280. ISSN: 00031305. DOI: 10.2307/2685143.

[39] James Aitchison. "Goodness of prediction fit". In: *Biometrika* 62.3 (Dec. 1975), pp. 547–554. ISSN: 00063444. DOI: 10.1093/biomet/62.3.547.

[40] Thomas P Minka. "Expectation Propagation for Approximate Bayesian Inference". In: *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence*. UAI'01. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2001, pp. 362–369. ISBN: 1558608001.

[41] José Miguel Hernández-Lobato et al. "Black-Box α-Divergence Minimization". In: *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. New York, NY, USA: JMLR.org, 2016, pp. 1511–1520.

[42] Shun-ichi Amari. "α-Divergence and α-Projection in Statistical Manifold". In: *Differential-Geometrical Methods in Statistics*. New York, NY: Springer New York, 1985, pp. 66–103. ISBN: 978-1-4612-5056-2. DOI: 10.1007/978-1-4612-5056-2_3.

[43] Andrzej Cichocki and Shun ichi Amari. "Families of alpha- beta- and gamma- divergences: Flexible and robust measures of Similarities". In: *Entropy* 12.6 (June 2010), pp. 1532–1568. ISSN: 10994300. DOI: 10.3390/e12061532.

[44] Alfréd Rényi. "On measures of entropy and information". In: *Fourth Berkeley Symposium on Mathematical Statistics and Probability*. Vol. 1. Fourth Berkeley Symposium on Mathematical Statistics and Probability. Berkeley, Calif.: University of California Press, 1961, pp. 547–561. DOI: 10.1021/jp106846b. arXiv: 1101.3070.

[45] Alex Krizhevsky and G Hinton. *Learning multiple layers of features from tiny images*. Tech. rep. 2009.

[46] Jonathan Krause et al. "3D Object Representations for Fine-Grained Categorization". In: *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*. Sydney, Australia, 2013.

[47] Diederik P. Kingma and Jimmy Lei Ba. "Adam: A method for stochastic optimization". In: *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings* (Dec. 2015). arXiv: 1412.6980.

[48] Christos Louizos and Max Welling. "Multiplicative normalizing flows for variational Bayesian neural networks". In: *34th International Conference on Machine Learning, ICML 2017*. Vol. 5. Mar. 2017, pp. 3480–3489. ISBN: 9781510855144. arXiv: 1703.01961.

[49] Herbert Robbins. "An Empirical Bayes Approach to Statistics". In: *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. Third Berkeley Symposium on Mathematical Statistics and Probability. Berkeley, Calif.: University of California Press, 1956, pp. 157–163. ISBN: 0097-0433.

[50] P. Welinder et al. *Caltech-UCSD Birds 200*. Tech. rep. CNS-TR-2010-001. California Institute of Technology, 2010.

[51] Eran Eidinger, Roee Enbar, and Tal Hassner. "Age and gender estimation of unfiltered faces". In: *IEEE Transactions on Information Forensics and Security* 9.12 (2014), pp. 2170–2179. ISSN: 15566013. DOI: 10.1109/TIFS.2014.2359646.

Appendix A

Results Sensitivity Study KL Divergence

In this appendix we show the results for the experiments with the KL in the sensitivity study.

A.1 Results on CIFAR10

A.2 Results on CIFAR100

A.3 Results on CARS

Results on CIFAR10, BNN trained on train set with topology: [] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	95.16	95.16	4.50	4.50	13.00
0.10	95.24	95.25	3.57	3.56	10000.00
0.50	95.35	95.32	3.17	3.19	42.00
1.00	95.42	95.31	2.95	3.03	31.00
2.00	95.40	95.35	2.81	2.84	23.00
10.00	95.43	95.40	2.41	2.26	42.00
20.00	95.40	95.47	2.01	1.97	562.00
40.00	95.36	95.46	1.88	1.66	13.00
100.00	95.48	95.49	1.11	1.17	749.00
200.00	95.34	95.48	0.45	0.77	23.00
450.00	95.18	95.46	1.33	1.77	13.00
1000.00	95.07	95.44	3.66	4.29	10.00
2000.00	94.97	95.51	7.74	8.57	10.00
4000.00	94.82	95.47	14.51	15.66	10.00
8000.00	94.46	95.48	25.55	27.07	10.00
10000.00	94.08	95.44	29.86	31.75	10.00
20000.00	88.71	95.40	40.85	48.64	10.00
45000.00	61.31	95.45	28.91	67.11	10.00
90000.00	34.36	95.45	7.04	76.52	10.00

Results on CIFAR10, BNN trained on validation set with topology: [] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	95.15	95.15	1.23	1.23	237.00
0.10	95.11	95.14	1.11	1.15	42.00
0.50	95.04	95.10	1.00	1.07	13.00
1.00	95.04	95.13	0.97	0.94	23.00
2.00	95.14	95.16	0.87	0.97	31.00
10.00	95.11	95.21	0.45	0.66	13.00
20.00	95.11	95.24	0.61	1.12	10.00
40.00	95.14	95.30	1.38	1.95	10.00
100.00	95.06	95.40	4.35	4.98	10.00
200.00	94.89	95.41	8.57	9.43	10.00
450.00	94.62	95.43	17.67	18.85	10.00
1000.00	93.64	95.46	32.47	34.71	10.00
2000.00	85.16	95.46	40.79	52.45	10.00
4000.00	59.17	95.42	27.07	67.64	10.00
8000.00	33.18	95.38	6.28	76.72	10.00
10000.00	27.62	95.28	1.65	78.44	10.00
20000.00	20.57	95.20	1.98	81.90	17.00
45000.00	17.27	94.60	0.42	83.16	42.00
90000.00	14.76	88.80	0.40	78.08	100.00

Results on CIFAR10, BNN trained on train set with topology: [32] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	95.22	95.22	4.57	4.57	1778.00
0.10	95.27	95.28	3.87	3.86	4216.00
0.50	95.26	95.25	3.58	3.58	177.00
1.00	95.18	95.25	3.59	3.49	31.00
2.00	95.26	95.30	3.34	3.29	42.00
10.00	95.30	95.29	2.90	2.89	237.00
20.00	95.33	95.32	2.64	2.65	7498.00
40.00	95.46	95.38	2.42	2.40	56.00
100.00	95.47	95.48	1.97	1.97	562.00
200.00	95.47	95.49	1.27	1.32	4216.00
450.00	95.32	95.46	0.42	1.05	31.00
1000.00	95.30	95.48	1.79	2.17	23.00
2000.00	95.19	95.44	4.56	5.05	17.00
4000.00	95.00	95.41	10.48	10.98	10.00
8000.00	94.66	95.40	21.57	22.94	10.00
10000.00	94.18	95.37	26.94	28.82	10.00
20000.00	69.32	88.56	31.67	54.54	10.00
45000.00	10.40	10.74	0.31	0.17	7498.00
90000.00	10.06	9.98	0.70	0.66	7498.00

Results on CIFAR10, BNN trained on validation set with topology: [32] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	92.60	92.59	6.16	6.17	10.00
0.10	93.13	93.08	4.35	4.39	56.00
0.50	94.16	94.18	1.77	1.75	31.00
1.00	94.67	94.66	1.05	0.93	23.00
2.00	94.86	94.92	0.71	0.51	13.00
10.00	95.17	95.26	1.30	1.30	13.00
20.00	95.03	95.25	1.71	1.47	13.00
40.00	95.12	95.18	1.76	2.01	100.00
100.00	95.08	95.20	3.83	3.90	13.00
200.00	94.81	95.25	7.07	7.74	10.00
450.00	94.65	95.31	15.44	16.68	10.00
1000.00	92.94	95.26	32.87	36.07	10.00
2000.00	20.86	30.93	3.40	16.99	31.00
4000.00	9.47	10.21	1.53	0.47	4216.00
8000.00	9.61	9.85	1.33	0.78	4216.00
10000.00	9.66	10.09	1.27	0.52	4216.00
20000.00	9.77	9.98	0.87	0.55	10000.00
45000.00	9.51	10.24	1.21	0.22	5623.00
90000.00	9.84	10.22	0.81	0.32	10000.00

Results on CIFAR10, BNN trained on train set with topology: [32-32] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	95.15	95.14	4.58	4.59	177.00
0.10	85.90	85.90	3.42	3.44	7498.00
0.50	86.25	86.24	3.07	3.08	1778.00
1.00	86.24	86.21	2.94	2.98	100.00
2.00	86.12	86.14	2.83	2.78	23.00
10.00	76.58	76.59	2.55	2.53	562.00
20.00	76.46	76.53	2.52	2.37	74.00
40.00	76.66	76.52	2.06	2.02	23.00
100.00	76.53	76.58	1.68	1.76	74.00
200.00	76.43	76.52	1.90	2.01	31.00
450.00	66.64	67.12	1.53	2.22	23.00
1000.00	66.87	67.14	1.96	3.17	31.00
2000.00	66.64	67.10	3.08	5.45	17.00
4000.00	48.10	48.44	4.43	6.62	42.00
8000.00	10.37	10.51	0.31	0.11	3162.00
10000.00	10.10	9.81	1.08	0.60	1000.00
20000.00	10.39	9.92	0.46	0.55	3162.00
45000.00	10.26	9.92	0.50	0.62	5623.00
90000.00	9.78	9.97	0.90	0.56	7498.00

Results on CIFAR10, BNN trained on validation set with topology: [32-32] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	92.77	92.77	4.28	4.28	31.00
0.10	93.53	93.55	2.75	2.74	421.00
0.50	94.42	94.39	1.33	1.01	10.00
1.00	94.58	94.65	0.91	0.99	74.00
2.00	94.31	94.30	2.61	2.64	17.00
10.00	57.86	57.72	3.96	3.83	421.00
20.00	48.41	48.28	2.68	2.58	316.00
40.00	48.45	48.33	3.52	4.75	10.00
100.00	19.68	19.96	2.74	2.43	1333.00
200.00	10.64	10.46	0.92	0.11	177.00
450.00	10.25	9.99	0.28	0.44	5623.00
1000.00	10.39	10.22	0.54	0.19	1778.00
2000.00	10.07	9.86	0.95	0.66	2371.00
4000.00	10.02	10.07	1.07	0.63	3162.00
8000.00	9.95	9.84	1.33	0.93	2371.00
10000.00	10.05	10.18	1.24	0.61	2371.00
20000.00	9.96	10.00	1.11	0.78	4216.00
45000.00	10.08	9.99	1.28	0.69	1778.00
90000.00	10.28	9.66	0.82	0.86	2371.00

Results on CIFAR10, BNN trained on train set with topology: [512-512] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	95.15	95.16	4.54	4.54	316.00
0.10	95.09	95.15	4.38	4.30	17.00
0.50	95.04	95.03	4.34	4.35	1778.00
1.00	95.13	95.12	4.21	4.22	5623.00
2.00	95.04	95.06	4.29	4.19	17.00
10.00	95.26	95.24	3.51	3.54	316.00
20.00	95.33	95.30	3.27	3.29	56.00
40.00	95.30	95.35	3.28	3.13	23.00
100.00	95.31	95.32	3.00	3.00	237.00
200.00	95.32	95.34	2.82	2.76	100.00
450.00	95.42	95.39	2.42	2.41	74.00
1000.00	95.30	95.29	2.11	2.11	4216.00
2000.00	95.39	95.42	1.44	1.46	56.00
4000.00	95.23	95.38	0.67	0.91	23.00
8000.00	95.18	95.44	0.77	1.65	17.00
10000.00	95.16	95.44	1.24	2.08	17.00
20000.00	95.19	95.41	4.48	4.64	23.00
45000.00	94.82	95.36	12.93	13.60	13.00
90000.00	93.39	95.34	29.76	32.39	10.00

Results on CIFAR10, BNN trained on validation set with topology: [512-512] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	91.62	91.63	3.45	3.45	562.00
0.10	93.04	93.05	2.98	3.06	17.00
0.50	94.02	94.24	1.22	1.36	17.00
1.00	94.15	94.18	2.13	2.14	133.00
2.00	93.61	93.97	2.97	3.46	10.00
10.00	91.32	91.36	7.83	7.67	13.00
20.00	93.32	93.39	5.51	5.80	17.00
40.00	94.68	94.58	3.85	3.79	100.00
100.00	94.81	95.09	1.47	2.14	17.00
200.00	95.13	95.18	1.03	2.00	17.00
450.00	95.24	95.24	2.11	2.49	17.00
1000.00	95.09	95.23	4.22	4.38	17.00
2000.00	95.12	95.25	7.81	8.14	17.00
4000.00	94.63	95.35	15.74	16.46	10.00
8000.00	80.26	86.61	25.10	34.49	10.00
10000.00	9.67	9.78	2.23	1.48	1333.00
20000.00	10.14	10.05	1.19	1.17	5623.00
45000.00	9.86	10.07	1.45	1.21	10000.00
90000.00	10.11	10.02	1.21	1.21	5623.00

Results on CIFAR100, BNN trained on train set with topology: [] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	75.36	75.35	23.46	23.45	17.00
0.10	78.00	78.02	12.45	12.42	7498.00
0.50	78.32	78.28	10.11	10.13	421.00
1.00	78.38	78.37	9.03	9.05	1333.00
2.00	78.47	78.43	7.67	7.67	421.00
10.00	78.18	78.26	4.32	4.27	1333.00
20.00	78.44	78.39	2.27	2.33	133.00
40.00	78.35	78.29	1.50	1.55	177.00
100.00	77.52	78.22	4.53	5.77	23.00
200.00	76.11	78.17	8.29	11.38	10.00
450.00	74.72	78.02	16.29	21.46	10.00
1000.00	71.30	77.90	27.51	36.51	10.00
2000.00	60.66	77.53	32.29	52.73	10.00
4000.00	33.71	76.83	15.87	65.46	10.00
8000.00	12.94	75.85	1.46	71.00	13.00
10000.00	10.27	75.32	2.20	71.51	17.00
20000.00	5.32	71.78	1.00	69.65	42.00
45000.00	3.41	60.22	0.22	58.77	133.00
90000.00	2.37	38.14	0.19	36.89	316.00

Results on CIFAR100, BNN trained on validation set with topology: [] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	68.91	68.94	28.44	28.41	13.00
0.10	71.15	71.21	18.99	18.60	13.00
0.50	72.53	72.84	13.59	12.78	13.00
1.00	73.64	74.17	9.87	8.69	13.00
2.00	75.66	75.76	4.19	3.94	133.00
10.00	75.68	77.82	5.17	7.75	10.00
20.00	75.00	77.96	10.61	14.92	10.00
40.00	73.77	78.03	19.44	25.60	10.00
100.00	66.19	77.82	30.89	45.47	10.00
200.00	46.34	76.48	24.30	59.47	10.00
450.00	16.68	73.52	3.15	66.97	10.00
1000.00	7.55	69.45	1.95	66.59	23.00
2000.00	5.45	62.64	0.69	60.84	74.00
4000.00	4.18	49.36	1.59	47.96	316.00
8000.00	2.22	29.67	0.11	28.43	421.00
10000.00	1.87	22.35	0.45	21.14	421.00
20000.00	1.48	8.26	0.48	7.08	749.00
45000.00	1.27	2.67	0.33	1.50	1778.00
90000.00	1.09	1.73	0.51	0.56	1778.00

Results on CIFAR100, BNN trained on train set with topology: [128] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	76.01	76.00	22.56	22.57	562.00
0.10	76.88	76.96	15.51	15.42	1000.00
0.50	76.84	76.83	14.23	14.23	7498.00
1.00	77.23	77.24	13.19	13.11	562.00
2.00	77.27	77.25	12.19	12.20	1333.00
10.00	77.30	77.36	9.74	9.66	3162.00
20.00	77.25	77.28	8.26	8.23	5623.00
40.00	77.42	77.45	6.41	6.37	7498.00
100.00	77.36	77.44	3.46	3.33	316.00
200.00	77.21	77.15	1.23	1.43	2371.00
450.00	75.89	76.87	4.97	5.50	13.00
1000.00	74.23	76.71	11.63	15.22	10.00
2000.00	70.78	76.11	22.84	29.83	10.00
4000.00	41.33	70.60	15.45	51.18	10.00
8000.00	1.23	1.00	0.70	0.63	1778.00
10000.00	1.18	0.96	0.54	0.61	4216.00
20000.00	1.16	1.06	0.71	0.48	1778.00
45000.00	1.15	1.18	0.54	0.30	3162.00
90000.00	1.17	1.06	0.54	0.36	2371.00

Results on CIFAR100, BNN trained on validation set with topology: [128] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	71.72	71.69	26.23	26.20	31.00
0.10	72.82	72.79	15.77	15.79	316.00
0.50	72.70	72.81	13.73	13.47	74.00
1.00	72.38	72.46	12.43	12.31	421.00
2.00	72.52	72.55	9.06	9.03	5623.00
10.00	74.85	74.85	2.79	2.95	133.00
20.00	74.69	75.93	5.14	7.43	17.00
40.00	73.29	76.23	9.97	14.36	10.00
100.00	67.58	75.20	19.13	29.72	10.00
200.00	47.99	66.91	17.30	41.87	10.00
450.00	9.60	14.12	2.06	8.11	31.00
1000.00	0.98	0.98	0.82	0.77	7498.00
2000.00	1.10	1.02	0.57	0.61	10000.00
4000.00	1.05	0.94	0.56	0.63	10000.00
8000.00	1.07	0.85	0.60	0.62	3162.00
10000.00	0.98	0.88	0.60	0.54	4216.00
20000.00	0.99	1.02	0.49	0.34	5623.00
45000.00	1.13	1.02	0.33	0.26	4216.00
90000.00	1.01	1.02	0.41	0.20	4216.00

Results on CIFAR100, BNN trained on train set with topology: [128-128] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	1.00	1.00	0.17	0.17	7498.00
0.10	1.00	1.00	0.17	0.17	10.00
0.50	1.05	1.00	0.12	0.17	10.00
1.00	0.98	1.00	0.20	0.17	42.00
2.00	0.95	1.00	0.23	0.17	42.00
10.00	1.06	1.00	0.16	0.17	74.00
20.00	1.13	1.01	0.14	0.15	74.00
40.00	1.08	0.95	0.37	0.21	23.00
100.00	1.03	0.97	0.16	0.18	749.00
200.00	0.98	0.97	0.21	0.17	1000.00
450.00	1.12	0.88	0.02	0.25	4216.00
1000.00	1.17	1.01	0.03	0.10	1778.00
2000.00	1.00	1.13	0.12	0.04	7498.00
4000.00	0.92	1.01	0.31	0.08	3162.00
8000.00	0.90	1.02	0.29	0.10	7498.00
10000.00	0.88	0.97	0.29	0.15	10000.00
20000.00	0.97	0.97	0.37	0.19	4216.00
45000.00	0.90	0.99	0.37	0.21	10000.00
90000.00	1.11	0.93	0.24	0.32	7498.00

Results on CIFAR100, BNN trained on validation set with topology: [128-128] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	1.00	1.00	0.28	0.28	31.00
0.10	1.00	1.00	0.28	0.28	3162.00
0.50	0.97	1.00	0.33	0.28	133.00
1.00	0.93	1.00	0.39	0.27	100.00
2.00	0.98	1.00	0.40	0.27	56.00
10.00	0.97	1.00	0.26	0.22	7498.00
20.00	0.92	0.97	0.28	0.21	2371.00
40.00	0.93	1.11	0.22	0.03	5623.00
100.00	0.90	1.00	0.51	0.10	562.00
200.00	0.95	1.15	0.36	0.06	1333.00
450.00	0.88	0.94	0.68	0.16	749.00
1000.00	1.00	1.01	0.54	0.12	1333.00
2000.00	1.02	1.16	0.47	0.01	2371.00
4000.00	0.94	0.91	0.70	0.29	1778.00
8000.00	0.85	0.90	0.47	0.36	10000.00
10000.00	0.96	0.96	0.73	0.33	1778.00
20000.00	0.86	1.13	0.78	0.22	2371.00
45000.00	0.94	0.97	0.72	0.42	2371.00
90000.00	0.98	1.00	0.53	0.34	4216.00

Results on CIFAR100, BNN trained on train set with topology: [512] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	75.79	75.78	22.68	22.70	316.00
0.10	77.06	77.03	16.12	16.10	74.00
0.50	77.25	77.30	14.77	14.70	1778.00
1.00	77.40	77.40	13.73	13.73	10000.00
2.00	77.38	77.45	12.77	12.68	316.00
10.00	77.72	77.76	9.95	9.89	4216.00
20.00	77.37	77.48	9.08	8.94	177.00
40.00	77.41	77.59	7.32	7.14	100.00
100.00	77.29	77.31	4.55	4.51	562.00
200.00	77.50	77.49	1.48	1.25	100.00
450.00	76.89	77.36	3.75	4.13	17.00
1000.00	74.82	77.23	9.62	12.73	10.00
2000.00	72.21	76.73	19.38	25.47	10.00
4000.00	57.77	75.06	24.76	45.77	10.00
8000.00	10.16	19.97	1.03	14.68	31.00
10000.00	2.40	2.34	0.21	0.21	5623.00
20000.00	1.13	1.02	0.72	0.79	10000.00
45000.00	0.99	1.00	0.88	0.85	10000.00
90000.00	0.97	1.02	0.91	0.78	4216.00

Results on CIFAR100, BNN trained on validation set with topology: [512] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	72.63	72.62	25.45	25.44	42.00
0.10	73.57	73.60	16.50	16.47	1778.00
0.50	73.44	73.46	15.09	15.07	1000.00
1.00	73.57	73.55	13.88	13.90	10000.00
2.00	73.82	74.01	12.29	11.75	23.00
10.00	75.26	75.31	5.29	5.24	4216.00
20.00	76.29	76.29	2.14	2.20	1000.00
40.00	74.55	76.98	6.18	8.50	10.00
100.00	71.49	76.11	15.69	22.27	10.00
200.00	60.43	73.38	21.30	37.84	10.00
450.00	16.83	28.38	2.70	18.51	17.00
1000.00	0.90	0.93	1.39	0.98	1333.00
2000.00	1.09	1.08	1.05	0.88	2371.00
4000.00	0.92	0.99	0.98	0.87	7498.00
8000.00	0.90	0.94	1.81	0.77	421.00
10000.00	0.85	0.99	1.08	0.71	2371.00
20000.00	1.02	1.01	0.94	0.74	1778.00
45000.00	0.95	0.96	1.04	0.53	1000.00
90000.00	0.87	0.90	0.61	0.42	4216.00

Results on CIFAR100, BNN trained on train set with topology: [512-512] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	1.00	1.00	0.17	0.17	749.00
0.10	1.00	1.00	0.17	0.17	7498.00
0.50	0.98	1.00	0.20	0.17	23.00
1.00	0.96	1.00	0.24	0.17	10.00
2.00	0.99	1.00	0.21	0.17	23.00
10.00	1.00	1.00	0.17	0.17	7498.00
20.00	0.95	1.00	0.23	0.17	1000.00
40.00	0.87	0.99	0.29	0.17	5623.00
100.00	0.92	0.94	0.25	0.22	3162.00
200.00	0.91	1.05	0.26	0.10	2371.00
450.00	1.14	1.02	0.03	0.11	3162.00
1000.00	1.16	1.06	0.02	0.04	7498.00
2000.00	1.16	1.26	0.02	0.15	5623.00
4000.00	1.11	1.00	0.07	0.13	10000.00
8000.00	1.05	0.96	0.17	0.20	10000.00
10000.00	1.02	0.92	0.22	0.25	10000.00
20000.00	0.97	1.02	0.30	0.18	10000.00
45000.00	0.93	1.16	0.50	0.15	5623.00
90000.00	1.01	0.86	0.57	0.65	5623.00

Results on CIFAR100, BNN trained on validation set with topology: [512-512] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	1.00	1.00	0.28	0.28	1000.00
0.10	1.00	1.00	0.28	0.28	10000.00
0.50	1.06	1.00	0.29	0.28	56.00
1.00	1.07	1.00	0.33	0.27	56.00
2.00	1.02	1.01	0.25	0.26	1000.00
10.00	0.95	1.00	0.27	0.22	5623.00
20.00	0.99	0.93	0.21	0.26	5623.00
40.00	0.99	1.09	0.31	0.06	1000.00
100.00	1.02	0.92	0.26	0.19	1778.00
200.00	1.12	0.84	0.14	0.27	3162.00
450.00	0.89	1.08	0.34	0.06	7498.00
1000.00	0.86	1.09	0.39	0.09	10000.00
2000.00	0.87	0.93	0.41	0.27	10000.00
4000.00	0.95	0.96	0.39	0.32	10000.00
8000.00	0.89	1.20	0.63	0.24	7498.00
10000.00	0.99	0.98	0.56	0.52	10000.00
20000.00	0.90	1.16	0.79	0.48	10000.00
45000.00	1.02	1.01	0.98	0.81	2371.00
90000.00	1.03	1.11	0.88	0.65	3162.00

Results on CIFAR100, BNN trained on train set with topology: [1024] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	76.36	76.38	22.09	22.06	562.00
0.10	76.57	76.57	17.47	17.47	4216.00
0.50	77.20	77.19	15.04	15.03	562.00
1.00	76.85	76.85	14.37	14.37	1778.00
2.00	77.00	77.01	13.62	13.61	7498.00
10.00	77.32	77.30	10.62	10.64	316.00
20.00	77.20	77.23	9.11	9.07	1778.00
40.00	77.45	77.44	7.31	7.31	10000.00
100.00	77.33	77.32	4.86	4.88	1778.00
200.00	77.60	77.57	1.87	1.79	5623.00
450.00	76.43	77.37	2.98	3.74	17.00
1000.00	74.87	77.24	8.91	11.67	10.00
2000.00	72.45	76.77	17.35	23.71	10.00
4000.00	60.80	75.52	25.39	43.61	10.00
8000.00	12.94	24.14	2.73	17.50	31.00
10000.00	4.65	5.51	0.54	2.58	237.00
20000.00	1.00	1.01	1.08	1.04	10000.00
45000.00	0.95	0.99	1.09	0.99	7498.00
90000.00	0.85	1.03	1.01	0.79	10000.00

Results on CIFAR100, BNN trained on validation set with topology: [1024] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	72.65	72.61	25.38	25.35	23.00
0.10	73.82	73.73	16.54	16.60	562.00
0.50	74.00	74.09	14.29	14.19	1778.00
1.00	73.56	73.61	14.22	13.67	23.00
2.00	73.98	73.99	12.19	12.16	4216.00
10.00	75.22	75.21	5.42	5.43	2371.00
20.00	75.99	76.00	1.77	1.82	133.00
40.00	74.78	76.43	5.23	6.85	13.00
100.00	71.61	76.23	13.97	20.33	10.00
200.00	63.00	73.30	21.08	34.52	10.00
450.00	19.73	39.92	3.73	27.93	10.00
1000.00	2.16	2.06	0.66	0.70	2371.00
2000.00	1.09	1.02	1.37	1.38	3162.00
4000.00	1.02	0.96	1.35	1.33	3162.00
8000.00	1.07	0.98	1.15	1.16	4216.00
10000.00	1.02	0.97	1.22	1.22	4216.00
20000.00	1.11	1.01	1.28	1.16	1000.00
45000.00	1.04	0.99	0.85	0.77	2371.00
90000.00	1.05	1.05	0.42	0.37	10000.00

Results on CIFAR100, BNN trained on train set with topology: [1024-1024] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	76.06	76.11	21.82	21.69	23.00
0.10	75.98	75.94	18.92	18.95	2371.00
0.50	76.09	76.07	18.10	18.11	749.00
1.00	76.24	76.29	17.03	16.97	2371.00
2.00	76.38	76.40	16.43	16.37	316.00
10.00	76.51	76.60	14.53	14.39	237.00
20.00	75.90	75.93	14.60	14.52	237.00
40.00	75.93	76.07	13.96	13.69	74.00
100.00	75.71	75.77	12.62	12.54	237.00
200.00	76.18	76.18	10.88	10.88	5623.00
450.00	75.81	75.99	8.75	8.53	316.00
1000.00	75.81	75.74	6.20	6.24	421.00
2000.00	74.88	75.03	4.49	4.35	4216.00
4000.00	74.76	74.79	2.25	2.22	562.00
8000.00	71.40	73.57	5.31	6.60	10.00
10000.00	71.10	73.61	7.32	9.97	10.00
20000.00	1.06	0.92	0.23	0.30	10000.00
45000.00	1.03	1.01	0.37	0.34	10000.00
90000.00	0.87	1.00	0.86	0.59	4216.00

Results on CIFAR100, BNN trained on validation set with topology: [1024-1024] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	1.00	1.00	0.28	0.28	1000.00
0.10	1.04	1.00	0.24	0.28	1778.00
0.50	1.04	1.00	0.26	0.28	316.00
1.00	1.09	1.00	0.19	0.27	749.00
2.00	1.00	0.99	0.27	0.28	7498.00
10.00	1.01	1.04	0.22	0.18	10000.00
20.00	1.14	0.99	0.14	0.20	1333.00
40.00	1.05	0.96	0.11	0.19	10000.00
100.00	0.98	1.19	0.23	0.08	4216.00
200.00	0.87	0.96	0.33	0.16	7498.00
450.00	1.26	0.91	0.09	0.25	4216.00
1000.00	1.14	1.07	0.21	0.12	5623.00
2000.00	0.95	1.16	0.97	0.07	1000.00
4000.00	0.86	1.07	0.57	0.32	10000.00
8000.00	1.00	1.05	0.71	0.64	10000.00
10000.00	1.07	0.99	0.94	0.76	1778.00
20000.00	0.84	1.06	1.11	0.73	3162.00
45000.00	0.96	0.95	0.99	0.96	10000.00
90000.00	0.95	0.98	1.00	0.96	10000.00

Results on CIFAR100, BNN trained on train set with topology: [2048] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	76.64	76.65	21.85	21.83	133.00
0.10	76.51	76.49	18.33	18.35	1778.00
0.50	76.53	76.53	17.13	17.13	562.00
1.00	76.67	76.66	15.01	15.02	3162.00
2.00	76.84	76.85	14.36	14.34	1778.00
10.00	77.10	77.19	11.07	10.97	749.00
20.00	77.42	77.35	9.32	9.36	1333.00
40.00	77.16	77.17	7.82	7.80	5623.00
100.00	77.31	77.51	5.08	4.85	562.00
200.00	77.22	77.25	2.23	2.21	562.00
450.00	76.71	77.29	2.46	3.16	31.00
1000.00	74.49	77.11	7.84	10.69	10.00
2000.00	72.57	76.76	15.98	22.45	10.00
4000.00	61.78	75.42	24.66	41.75	10.00
8000.00	14.94	30.26	2.00	22.72	23.00
10000.00	6.56	12.96	0.80	8.90	56.00
20000.00	0.96	0.95	1.44	1.36	4216.00
45000.00	0.96	1.08	1.30	1.12	7498.00
90000.00	0.96	1.09	1.24	1.07	7498.00

Results on CIFAR100, BNN trained on validation set with topology: [2048] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	72.98	72.99	24.87	24.85	177.00
0.10	73.74	73.77	17.43	17.37	749.00
0.50	73.98	74.01	14.41	14.34	421.00
1.00	74.06	74.13	13.45	13.31	133.00
2.00	74.12	74.41	12.29	11.83	74.00
10.00	75.13	75.16	6.30	6.27	4216.00
20.00	75.71	76.02	2.01	1.70	100.00
40.00	76.32	76.53	5.60	5.89	100.00
100.00	72.35	76.37	13.35	18.85	10.00
200.00	63.94	73.86	20.11	33.26	10.00
450.00	23.02	44.98	4.56	31.28	10.00
1000.00	2.31	2.38	0.67	0.59	10000.00
2000.00	1.00	1.00	1.87	1.87	10000.00
4000.00	1.00	1.00	1.72	1.72	10000.00
8000.00	1.00	1.00	1.97	1.97	10000.00
10000.00	0.99	1.00	2.10	2.08	2371.00
20000.00	1.00	1.00	1.83	1.83	5623.00
45000.00	1.06	1.02	0.78	0.78	5623.00
90000.00	1.22	1.08	0.33	0.43	10000.00

Results on CIFAR100, BNN trained on train set with topology: [2048-2048] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	75.94	75.94	22.04	22.03	133.00
0.10	76.09	76.08	19.59	19.59	3162.00
0.50	75.89	75.87	19.11	19.12	1333.00
1.00	76.30	76.30	17.83	17.83	1333.00
2.00	76.25	76.26	17.83	17.82	7498.00
10.00	75.81	75.77	16.58	16.61	3162.00
20.00	76.28	76.32	14.85	14.80	749.00
40.00	76.49	76.50	13.97	13.96	3162.00
100.00	75.93	75.89	13.23	13.26	1778.00
200.00	76.01	76.03	11.89	11.88	10000.00
450.00	76.12	76.13	10.06	10.05	1778.00
1000.00	75.82	75.72	8.08	8.15	133.00
2000.00	75.61	75.61	5.67	5.67	10000.00
4000.00	75.33	75.35	2.70	2.67	10000.00
8000.00	73.99	74.44	3.09	3.51	31.00
10000.00	73.94	74.30	4.61	5.20	31.00
20000.00	69.93	73.02	9.90	14.33	10.00
45000.00	0.97	0.98	0.55	0.51	10000.00
90000.00	1.00	1.00	0.97	0.95	5623.00

Results on CIFAR100, BNN trained on validation set with topology: [2048-2048] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	1.89	1.90	0.33	0.31	23.00
0.10	0.95	1.00	0.34	0.28	316.00
0.50	1.07	1.00	0.21	0.28	1000.00
1.00	1.03	0.99	0.25	0.28	1333.00
2.00	1.07	0.96	0.22	0.31	749.00
10.00	0.94	1.01	0.33	0.22	1778.00
20.00	1.12	1.02	0.08	0.17	7498.00
40.00	1.04	0.99	0.17	0.16	4216.00
100.00	1.07	1.18	0.12	0.05	7498.00
200.00	1.09	1.19	0.22	0.05	4216.00
450.00	1.15	0.97	0.38	0.21	2371.00
1000.00	5.75	6.48	0.56	2.09	421.00
2000.00	0.97	0.83	0.46	0.47	5623.00
4000.00	1.03	1.01	0.59	0.59	10000.00
8000.00	0.97	1.01	1.04	0.98	5623.00
10000.00	1.08	1.05	0.97	0.99	10000.00
20000.00	0.84	1.06	1.34	1.07	7498.00
45000.00	0.99	1.04	1.45	1.20	3162.00
90000.00	0.94	1.07	1.40	1.24	10000.00

Results on CARS, BNN trained on train set with topology: [] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	82.37	82.37	17.01	16.96	17.00
0.10	86.19	86.07	5.18	5.29	3162.00
0.50	86.47	86.47	2.27	2.38	4216.00
1.00	86.42	86.56	1.73	1.83	133.00
2.00	86.93	86.94	3.00	2.73	177.00
10.00	85.10	87.11	8.96	10.69	10.00
20.00	84.21	87.03	13.63	16.97	10.00
40.00	81.98	86.86	19.63	25.78	10.00
100.00	74.86	85.74	27.69	41.08	10.00
200.00	62.31	83.85	28.70	53.85	10.00
450.00	38.32	78.92	17.03	63.93	10.00
1000.00	16.33	65.15	2.41	58.70	13.00
2000.00	8.67	45.60	1.53	42.26	23.00
4000.00	5.59	27.48	0.44	25.48	56.00
8000.00	3.35	18.99	0.24	17.66	133.00
10000.00	2.99	15.16	0.30	13.96	177.00
20000.00	1.76	9.47	0.21	8.55	316.00
45000.00	1.05	7.17	0.47	6.42	562.00
90000.00	1.10	3.93	0.14	3.25	2371.00

Results on CARS, BNN trained on validation set with topology: [] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	69.56	69.51	23.83	23.71	23.00
0.10	79.61	80.22	2.40	2.72	42.00
0.50	79.41	80.99	5.29	7.09	17.00
1.00	78.77	81.85	7.53	11.05	10.00
2.00	78.28	82.04	11.20	15.64	10.00
10.00	73.49	81.60	21.83	31.79	10.00
20.00	67.74	80.91	25.80	42.18	10.00
40.00	56.21	79.12	24.96	52.56	10.00
100.00	30.64	71.60	10.44	59.02	10.00
200.00	15.43	62.23	2.55	55.96	13.00
450.00	8.30	46.31	1.83	43.49	31.00
1000.00	5.11	32.17	1.30	30.65	100.00
2000.00	2.86	21.21	0.21	20.19	177.00
4000.00	1.53	11.99	0.67	11.17	237.00
8000.00	1.79	6.71	0.65	5.97	1333.00
10000.00	1.10	5.59	0.41	4.87	562.00
20000.00	0.64	2.68	0.86	2.01	562.00
45000.00	1.02	1.22	0.27	0.56	7498.00
90000.00	0.67	0.94	0.35	0.28	1778.00

Results on CARS, BNN trained on train set with topology: [128] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	28.05	28.02	9.78	9.77	31.00
0.10	33.91	34.39	7.26	7.27	10.00
0.50	32.45	32.49	5.22	5.05	100.00
1.00	29.26	29.58	3.67	3.61	17.00
2.00	25.11	25.62	2.74	2.93	42.00
10.00	11.71	12.47	1.42	2.69	42.00
20.00	8.42	8.73	0.82	1.59	74.00
40.00	8.73	8.93	1.50	1.90	316.00
100.00	9.59	9.87	2.60	2.83	4216.00
200.00	7.60	8.63	3.77	3.22	56.00
450.00	3.98	3.96	2.46	2.35	10000.00
1000.00	1.28	1.12	1.58	1.69	7498.00
2000.00	0.58	0.58	1.26	1.13	7498.00
4000.00	0.69	0.61	0.80	0.81	5623.00
8000.00	0.72	0.71	0.59	0.55	7498.00
10000.00	0.67	0.77	0.57	0.44	10000.00
20000.00	0.64	0.59	0.49	0.49	5623.00
45000.00	0.62	0.53	0.28	0.34	10000.00
90000.00	0.58	0.56	0.28	0.26	10000.00

Results on CARS, BNN trained on validation set with topology: [128] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	24.47	24.50	15.09	15.09	316.00
0.10	28.35	28.55	7.96	8.17	177.00
0.50	27.84	27.82	4.56	4.55	5623.00
1.00	27.04	27.76	3.31	3.13	23.00
2.00	26.64	27.40	1.78	1.96	42.00
10.00	21.72	23.88	3.48	4.89	13.00
20.00	16.61	16.71	4.91	4.74	133.00
40.00	12.93	13.62	5.24	5.07	100.00
100.00	6.02	6.28	3.64	3.44	7498.00
200.00	2.06	1.86	1.85	1.94	5623.00
450.00	0.64	0.56	0.72	0.75	7498.00
1000.00	0.62	0.56	0.61	0.65	10000.00
2000.00	0.54	0.48	0.52	0.52	7498.00
4000.00	0.33	0.43	0.57	0.43	10000.00
8000.00	0.59	0.53	0.69	0.25	1000.00
10000.00	0.56	0.53	0.70	0.23	1000.00
20000.00	0.49	0.38	0.28	0.34	10000.00
45000.00	0.39	0.46	0.37	0.25	10000.00
90000.00	0.56	0.33	0.66	0.36	1000.00

Results on CARS, BNN trained on train set with topology: [128-128] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	0.95	0.95	0.11	0.11	10000.00
0.10	0.95	0.95	0.11	0.11	4216.00
0.50	0.95	0.95	0.11	0.11	10000.00
1.00	0.95	0.95	0.12	0.12	10000.00
2.00	0.95	0.95	0.12	0.12	10000.00
10.00	0.87	0.95	0.05	0.17	23.00
20.00	0.89	0.95	0.10	0.21	42.00
40.00	0.66	0.95	0.15	0.27	42.00
100.00	0.94	0.95	0.32	0.34	1778.00
200.00	0.87	0.97	0.30	0.40	7498.00
450.00	0.76	0.81	0.19	0.26	5623.00
1000.00	0.46	0.46	0.16	0.08	1778.00
2000.00	0.61	0.49	0.04	0.05	1778.00
4000.00	0.43	0.46	0.46	0.10	562.00
8000.00	0.53	0.62	0.09	0.05	7498.00
10000.00	0.53	0.56	0.15	0.02	4216.00
20000.00	0.61	0.67	0.07	0.06	7498.00
45000.00	0.69	0.74	0.08	0.08	5623.00
90000.00	0.69	0.79	0.22	0.10	3162.00

Results on CARS, BNN trained on validation set with topology: [128-128] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	0.54	0.54	0.11	0.11	133.00
0.10	0.51	0.54	0.14	0.11	562.00
0.50	0.59	0.56	0.05	0.09	10000.00
1.00	0.56	0.54	0.08	0.10	10000.00
2.00	0.48	0.53	0.39	0.10	10.00
10.00	0.54	0.56	0.20	0.02	133.00
20.00	0.54	0.61	0.25	0.05	133.00
40.00	0.36	0.46	0.37	0.09	316.00
100.00	0.53	0.79	0.49	0.25	133.00
200.00	0.44	0.53	0.61	0.01	177.00
450.00	0.39	0.48	0.59	0.07	316.00
1000.00	0.69	0.43	0.03	0.14	1778.00
2000.00	0.66	0.66	0.09	0.07	2371.00
4000.00	0.58	0.90	0.20	0.29	3162.00
8000.00	0.67	0.46	0.05	0.18	7498.00
10000.00	0.72	0.76	0.14	0.10	3162.00
20000.00	0.49	0.66	0.31	0.02	5623.00
45000.00	0.69	0.53	0.08	0.16	7498.00
90000.00	0.66	0.38	0.26	0.30	3162.00

Results on CARS, BNN trained on train set with topology: [512] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	82.29	82.31	16.28	16.26	421.00
0.10	82.52	82.49	9.65	9.63	177.00
0.50	83.23	83.26	7.28	7.27	1000.00
1.00	83.08	83.00	6.30	6.38	4216.00
2.00	82.93	83.14	6.02	5.78	237.00
10.00	82.47	82.54	3.43	3.36	2371.00
20.00	82.34	82.35	1.66	1.59	10000.00
40.00	82.06	82.40	2.20	2.34	56.00
100.00	79.99	82.54	5.69	8.59	10.00
200.00	76.55	81.02	9.94	16.11	10.00
450.00	65.52	74.94	15.41	28.03	10.00
1000.00	38.35	59.28	9.80	37.54	10.00
2000.00	6.63	8.24	2.72	3.59	74.00
4000.00	1.55	1.73	0.98	0.55	1000.00
8000.00	0.81	0.94	0.80	0.58	2371.00
10000.00	0.84	0.86	0.65	0.54	3162.00
20000.00	0.89	1.02	0.45	0.23	3162.00
45000.00	0.72	0.67	0.47	0.49	10000.00
90000.00	0.41	0.49	0.70	0.56	7498.00

Results on CARS, BNN trained on validation set with topology: [512] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	69.81	69.96	19.53	19.38	1000.00
0.10	73.43	73.38	6.61	6.63	5623.00
0.50	74.36	74.18	3.88	3.69	177.00
1.00	73.33	73.97	3.69	2.71	56.00
2.00	72.01	73.06	4.20	2.29	13.00
10.00	70.99	72.36	2.69	3.67	17.00
20.00	69.77	72.26	4.00	7.27	10.00
40.00	67.23	71.98	7.20	13.43	10.00
100.00	58.48	68.44	12.53	26.39	10.00
200.00	34.71	53.38	6.86	30.70	10.00
450.00	8.37	11.73	3.62	6.87	100.00
1000.00	1.32	1.18	0.44	0.53	7498.00
2000.00	0.36	0.31	0.82	0.75	4216.00
4000.00	0.61	0.53	0.45	0.47	4216.00
8000.00	0.56	0.69	0.39	0.13	4216.00
10000.00	0.51	0.59	0.58	0.21	1778.00
20000.00	0.56	0.48	0.53	0.48	3162.00
45000.00	0.49	0.49	0.39	0.32	7498.00
90000.00	0.49	0.38	0.32	0.40	10000.00

Results on CARS, BNN trained on train set with topology: [512-512] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	0.95	0.95	0.11	0.11	13.00
0.10	0.95	0.95	0.11	0.11	1778.00
0.50	0.95	0.95	0.11	0.11	5623.00
1.00	0.95	0.95	0.12	0.12	1333.00
2.00	0.95	0.95	0.12	0.12	1333.00
10.00	0.92	0.95	0.13	0.17	237.00
20.00	0.95	0.95	0.20	0.21	316.00
40.00	0.79	0.95	0.02	0.27	100.00
100.00	0.67	0.95	0.05	0.33	237.00
200.00	0.87	0.95	0.29	0.37	7498.00
450.00	0.43	0.69	0.29	0.13	562.00
1000.00	0.46	0.72	0.20	0.17	1333.00
2000.00	0.38	0.54	0.19	0.01	10000.00
4000.00	0.38	0.81	0.33	0.23	2371.00
8000.00	0.79	0.79	0.05	0.17	3162.00
10000.00	0.86	0.79	0.19	0.15	10000.00
20000.00	0.92	0.66	0.07	0.08	4216.00
45000.00	0.82	0.58	0.22	0.30	3162.00
90000.00	0.39	0.59	0.53	0.20	3162.00

Results on CARS, BNN trained on validation set with topology: [512-512] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	0.54	0.54	0.11	0.11	13.00
0.10	0.61	0.54	0.04	0.11	10000.00
0.50	0.58	0.56	0.08	0.09	562.00
1.00	0.53	0.56	0.11	0.08	10000.00
2.00	0.56	0.56	0.07	0.07	10000.00
10.00	0.51	0.44	0.09	0.14	3162.00
20.00	0.43	0.51	0.36	0.05	177.00
40.00	0.38	0.53	0.34	0.02	421.00
100.00	0.51	0.53	0.09	0.02	2371.00
200.00	0.58	0.53	0.02	0.02	10000.00
450.00	0.41	0.54	0.65	0.01	316.00
1000.00	0.64	0.86	0.10	0.27	2371.00
2000.00	0.76	1.00	0.04	0.38	2371.00
4000.00	0.76	0.76	0.02	0.06	10000.00
8000.00	0.64	0.58	0.26	0.24	5623.00
10000.00	0.64	0.64	0.29	0.21	5623.00
20000.00	0.53	0.54	0.28	0.20	10000.00
45000.00	0.62	0.61	0.57	0.24	1333.00
90000.00	0.48	0.56	0.40	0.26	7498.00

Results on CARS, BNN trained on train set with topology: [1024] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	81.93	81.96	16.67	16.63	100.00
0.10	83.88	83.95	9.16	9.08	316.00
0.50	82.62	82.90	7.98	7.59	56.00
1.00	82.96	83.11	7.42	7.20	100.00
2.00	83.49	83.49	5.90	5.88	749.00
10.00	83.37	83.41	3.34	3.45	562.00
20.00	83.70	83.88	1.75	1.67	562.00
40.00	83.51	83.70	2.14	2.45	74.00
100.00	81.96	83.85	4.80	7.24	17.00
200.00	78.51	82.26	8.76	13.15	10.00
450.00	72.49	79.74	15.84	25.31	10.00
1000.00	53.74	69.59	16.53	37.23	10.00
2000.00	21.33	42.74	4.71	31.28	13.00
4000.00	3.80	3.60	0.55	0.38	7498.00
8000.00	0.76	0.77	0.97	0.91	7498.00
10000.00	0.61	0.76	1.15	0.90	3162.00
20000.00	0.43	0.48	1.24	1.17	5623.00
45000.00	0.43	0.48	1.08	0.97	4216.00
90000.00	0.36	0.51	0.92	0.70	5623.00

Results on CARS, BNN trained on validation set with topology: [1024] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	73.80	74.10	18.20	17.83	56.00
0.10	76.83	76.94	5.21	5.08	562.00
0.50	75.74	75.84	3.14	3.04	4216.00
1.00	75.22	75.22	2.39	2.28	5623.00
2.00	74.25	74.71	2.55	2.25	74.00
10.00	72.31	74.07	2.74	3.82	17.00
20.00	73.11	74.31	6.18	7.91	31.00
40.00	69.66	74.59	7.96	13.99	10.00
100.00	64.13	72.13	12.80	23.98	10.00
200.00	50.19	66.91	13.69	35.30	10.00
450.00	18.78	37.71	4.26	27.49	17.00
1000.00	2.93	2.93	0.66	0.73	5623.00
2000.00	0.56	0.58	0.65	0.52	4216.00
4000.00	0.43	0.44	0.59	0.50	5623.00
8000.00	0.59	0.46	0.50	0.50	3162.00
10000.00	0.56	0.49	0.60	0.55	3162.00
20000.00	0.44	0.48	0.83	0.78	5623.00
45000.00	0.41	0.38	0.59	0.54	5623.00
90000.00	0.59	0.48	0.74	0.41	1000.00

Results on CARS, BNN trained on train set with topology: [1024-1024] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	0.95	0.95	0.11	0.11	23.00
0.10	0.95	0.95	0.11	0.11	2371.00
0.50	0.95	0.95	0.11	0.11	5623.00
1.00	0.95	0.95	0.12	0.12	5623.00
2.00	0.95	0.95	0.12	0.12	2371.00
10.00	0.97	0.95	0.18	0.17	749.00
20.00	0.64	0.95	0.36	0.21	10.00
40.00	0.95	0.95	0.27	0.27	1333.00
100.00	0.59	0.95	0.09	0.34	421.00
200.00	0.48	0.95	0.14	0.37	1333.00
450.00	0.51	0.56	0.19	0.00	749.00
1000.00	0.53	0.79	0.09	0.23	3162.00
2000.00	0.48	0.76	0.37	0.19	749.00
4000.00	0.61	0.82	0.41	0.21	562.00
8000.00	0.76	0.82	0.01	0.11	5623.00
10000.00	0.90	0.76	0.37	0.02	562.00
20000.00	1.04	0.59	0.24	0.38	1000.00
45000.00	0.66	0.56	1.23	0.57	421.00
90000.00	0.72	0.49	0.43	0.42	2371.00

Results on CARS, BNN trained on validation set with topology: [1024-1024] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	0.54	0.54	0.11	0.11	7498.00
0.10	0.66	0.54	0.01	0.11	10000.00
0.50	0.56	0.59	0.08	0.05	10000.00
1.00	0.58	0.51	0.06	0.13	10000.00
2.00	0.46	0.61	0.17	0.02	10000.00
10.00	0.44	0.35	0.16	0.24	2371.00
20.00	0.77	0.54	0.20	0.02	7498.00
40.00	0.59	0.58	0.02	0.03	4216.00
100.00	0.64	0.54	0.16	0.00	421.00
200.00	0.59	0.74	0.02	0.19	7498.00
450.00	0.69	0.71	0.33	0.14	421.00
1000.00	0.66	1.10	0.35	0.50	749.00
2000.00	0.84	0.77	0.12	0.08	1333.00
4000.00	0.89	0.61	0.26	0.26	1333.00
8000.00	0.86	0.62	0.50	0.44	1333.00
10000.00	0.84	0.53	0.47	0.58	1778.00
20000.00	0.46	0.46	0.36	0.35	10000.00
45000.00	0.58	0.53	0.33	0.37	10000.00
90000.00	0.54	0.51	0.38	0.37	10000.00

Results on CARS, BNN trained on train set with topology: [2048] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	82.55	82.57	16.11	16.00	23.00
0.10	84.25	84.20	9.30	9.33	562.00
0.50	83.39	83.34	7.99	8.04	177.00
1.00	83.46	83.64	6.92	6.73	133.00
2.00	83.52	83.62	6.45	6.35	1333.00
10.00	84.21	84.11	3.43	3.55	421.00
20.00	84.03	84.46	2.53	2.48	23.00
40.00	83.11	84.25	2.77	2.00	13.00
100.00	81.71	83.59	4.25	5.39	10.00
200.00	79.87	83.19	7.73	11.59	10.00
450.00	73.79	80.92	14.57	23.07	10.00
1000.00	59.58	74.08	18.60	36.64	10.00
2000.00	27.04	51.52	5.14	35.90	10.00
4000.00	6.82	8.57	0.74	3.74	100.00
8000.00	1.35	1.28	1.01	1.06	10000.00
10000.00	0.48	0.48	1.55	1.54	10000.00
20000.00	0.49	0.48	1.42	1.44	10000.00
45000.00	0.59	0.53	1.09	1.04	4216.00
90000.00	0.66	0.64	0.97	0.96	10000.00

Results on CARS, BNN trained on validation set with topology: [2048] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	71.68	71.70	20.95	20.87	74.00
0.10	78.06	78.10	5.56	5.58	10000.00
0.50	76.30	76.34	3.77	3.68	2371.00
1.00	75.27	75.25	3.39	3.33	1333.00
2.00	74.71	75.28	2.79	1.78	42.00
10.00	72.36	74.77	2.94	3.53	10.00
20.00	72.13	74.71	3.97	6.86	13.00
40.00	69.89	73.84	5.69	11.47	10.00
100.00	66.80	73.87	13.24	23.22	10.00
200.00	55.78	70.42	16.26	34.94	10.00
450.00	23.94	48.00	5.67	34.64	13.00
1000.00	5.30	5.41	2.09	2.40	1333.00
2000.00	1.00	0.84	0.42	0.52	5623.00
4000.00	0.51	0.58	0.68	0.59	10000.00
8000.00	0.43	0.49	0.95	0.86	10000.00
10000.00	0.43	0.53	1.08	0.90	3162.00
20000.00	0.49	0.38	1.05	1.06	3162.00
45000.00	0.44	0.35	0.73	0.75	5623.00
90000.00	0.48	0.49	0.93	0.51	1000.00

Results on CARS, BNN trained on train set with topology: [2048-2048] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	0.95	0.95	0.11	0.11	17.00
0.10	0.97	0.95	0.13	0.11	7498.00
0.50	0.95	0.95	0.11	0.11	1333.00
1.00	0.95	0.95	0.10	0.12	74.00
2.00	0.90	0.95	0.07	0.12	316.00
10.00	0.87	0.95	0.05	0.17	133.00
20.00	0.94	0.95	0.19	0.21	2371.00
40.00	0.99	0.95	0.29	0.26	1000.00
100.00	0.71	0.95	0.07	0.33	1000.00
200.00	0.95	0.97	0.37	0.39	10000.00
450.00	0.53	0.79	0.12	0.23	1333.00
1000.00	0.62	1.12	0.15	0.55	749.00
2000.00	0.69	1.04	0.12	0.43	1000.00
4000.00	0.92	0.66	0.14	0.19	7498.00
8000.00	0.79	0.69	0.84	0.83	3162.00
10000.00	0.89	0.62	0.85	1.02	3162.00
20000.00	0.71	0.56	1.32	1.40	2371.00
45000.00	0.56	0.62	0.97	0.83	5623.00
90000.00	0.43	0.41	0.72	0.64	5623.00

Results on CARS, BNN trained on validation set with topology: [2048-2048] and DKL divergence

β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.00	0.54	0.54	0.11	0.11	316.00
0.10	0.67	0.54	0.02	0.11	10000.00
0.50	0.44	0.51	0.24	0.13	316.00
1.00	0.54	0.39	0.11	0.24	1333.00
2.00	0.59	0.58	0.05	0.05	1778.00
10.00	0.46	0.48	0.30	0.10	237.00
20.00	0.67	0.44	0.39	0.12	74.00
40.00	0.58	0.53	0.18	0.03	421.00
100.00	0.56	0.62	0.00	0.08	10000.00
200.00	0.51	0.77	0.06	0.22	10000.00
450.00	0.97	0.87	0.36	0.29	10000.00
1000.00	0.81	0.89	0.09	0.21	7498.00
2000.00	1.02	0.59	0.41	0.33	562.00
4000.00	0.76	0.59	0.85	0.64	1000.00
8000.00	0.62	0.49	1.05	0.76	1000.00
10000.00	0.43	0.53	1.22	0.75	1000.00
20000.00	0.36	0.51	0.55	0.33	10000.00
45000.00	0.26	0.59	0.85	0.33	2371.00
90000.00	0.44	0.43	0.76	0.58	1778.00

Appendix B

Results Sensitivity Study Rényi's α-divergence

In this appendix we show the results for the experiments with the Rényi's α-divergence in the sensitivity study.

B.1 Results on CIFAR10

B.2 Results on CIFAR100

B.3 Results on CARS

Results on CIFAR10, BNN trained on train set with topology: [] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	95.30	95.29	3.16	3.17	5623.00
0.50	95.39	95.33	2.85	2.88	23.00
1.00	95.42	95.31	2.95	3.03	31.00
1.50	95.37	95.33	3.06	3.13	42.00
2.00	95.35	95.31	3.14	3.17	42.00
2.50	95.35	95.33	3.23	3.22	31.00

Results on CIFAR10, BNN trained on validation set with topology: [] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	95.17	95.15	0.56	0.56	100.00
0.50	94.99	95.17	0.98	0.91	10.00
1.00	95.04	95.12	0.97	0.98	23.00
1.50	95.11	95.11	1.01	1.01	31.00
2.00	95.04	95.10	1.00	1.08	13.00
2.50	95.05	95.11	1.08	1.01	10.00

Results on CIFAR10, BNN trained on train set with topology: [32] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	95.32	95.31	3.28	3.28	3162.00
0.50	95.31	95.34	3.28	3.24	133.00
1.00	95.33	95.31	3.32	3.37	1000.00
1.50	95.28	95.29	3.48	3.46	237.00
2.00	95.28	95.29	3.55	3.54	2371.00
2.50	95.28	95.24	3.62	3.59	42.00

Results on CIFAR10, BNN trained on validation set with topology: [32] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	95.20	95.18	1.88	1.90	7498.00
0.50	94.92	94.99	0.79	0.76	13.00
1.00	94.43	94.32	1.16	1.19	10.00
1.50	94.18	94.18	1.46	1.48	1333.00
2.00	93.78	93.82	2.42	2.35	23.00
2.50	93.52	93.54	2.94	2.93	1333.00

Results on CIFAR10, BNN trained on train set with topology: [32-32] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	95.19	95.19	3.62	3.62	3162.00
0.50	95.21	95.21	3.61	3.61	316.00
1.00	95.21	95.21	3.87	3.84	749.00
1.50	95.33	95.33	3.77	3.77	562.00
2.00	95.11	95.12	4.06	4.04	10000.00
2.50	95.25	95.27	3.91	3.90	421.00

Results on CIFAR10, BNN trained on validation set with topology: [32-32] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	94.71	94.94	3.03	3.30	10.00
0.50	94.75	94.89	1.07	1.17	13.00
1.00	94.37	94.36	0.54	0.55	42.00
1.50	93.40	93.39	2.02	2.04	23.00
2.00	93.28	93.31	2.87	2.90	177.00
2.50	93.49	93.53	2.91	2.85	316.00

Results on CIFAR10, BNN trained on train set with topology: [512-512] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	95.15	95.16	4.01	3.99	31.00
0.50	95.29	95.28	3.80	3.82	1000.00
1.00	95.07	95.06	4.10	4.12	1778.00
1.50	95.21	95.21	3.86	3.85	10000.00
2.00	95.31	95.30	3.88	3.89	1333.00
2.50	95.29	95.28	3.84	3.86	1778.00

Results on CIFAR10, BNN trained on validation set with topology: [512-512] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	90.34	90.68	8.17	8.75	10.00
0.50	94.36	94.64	4.70	5.02	10.00
1.00	93.88	93.91	1.85	1.89	42.00
1.50	93.59	93.72	1.34	1.58	13.00
2.00	93.89	93.89	1.13	1.07	5623.00
2.50	93.70	93.70	2.01	2.00	237.00

Results on CIFAR100, BNN trained on train set with topology: [] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	78.62	78.66	12.09	12.05	4216.00
0.50	78.35	78.33	10.09	10.10	2371.00
1.00	78.37	78.46	9.02	8.95	2371.00
1.50	78.38	78.41	9.40	9.33	2371.00
2.00	78.53	78.51	9.68	9.71	1333.00
2.50	78.37	78.34	10.09	10.11	1778.00

Results on CIFAR100, BNN trained on validation set with topology: [] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	75.72	75.70	3.43	3.45	4216.00
0.50	75.36	75.36	4.98	4.99	3162.00
1.00	73.94	74.07	9.03	8.78	133.00
1.50	73.33	73.29	11.14	11.16	237.00
2.00	72.72	72.72	12.80	12.80	5623.00
2.50	72.34	72.38	13.95	13.91	5623.00

Results on CIFAR100, BNN trained on train set with topology: [128] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	77.50	77.54	14.19	14.11	316.00
0.50	76.86	76.89	12.01	11.98	10000.00
1.00	77.13	77.16	13.28	13.23	421.00
1.50	77.02	77.06	13.73	13.67	421.00
2.00	76.99	77.07	14.26	14.12	100.00
2.50	76.92	76.92	14.29	14.28	1333.00

Results on CIFAR100, BNN trained on validation set with topology: [128] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	50.28	53.56	11.57	16.13	10.00
0.50	71.77	71.76	2.67	2.55	74.00
1.00	72.74	72.77	11.93	11.90	10000.00
1.50	72.20	72.05	14.67	14.72	133.00
2.00	72.70	72.70	14.50	14.51	7498.00
2.50	72.46	72.40	15.17	15.22	237.00

Results on CIFAR100, BNN trained on train set with topology: [128-128] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	77.20	77.08	15.00	15.10	177.00
0.50	76.29	76.28	11.84	11.82	133.00
1.00	0.99	1.00	0.19	0.17	23.00
1.50	75.89	75.93	12.70	12.65	237.00
2.00	75.76	75.77	13.38	13.38	1333.00
2.50	76.23	76.15	13.90	13.88	177.00

Results on CIFAR100, BNN trained on validation set with topology: [128-128] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	0.96	1.00	0.39	0.25	23.00
0.50	39.97	41.43	5.11	6.96	10.00
1.00	1.03	1.00	0.25	0.27	562.00
1.50	71.63	71.58	14.83	14.88	7498.00
2.00	71.79	71.89	15.51	15.35	100.00
2.50	71.75	71.87	16.28	15.84	31.00

Results on CIFAR100, BNN trained on train set with topology: [512] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	77.83	77.82	14.15	14.12	237.00
0.50	77.35	77.30	12.99	12.99	237.00
1.00	77.37	77.31	13.58	13.64	5623.00
1.50	76.81	76.80	14.82	14.83	5623.00
2.00	77.27	77.28	14.69	14.68	4216.00
2.50	77.51	77.52	13.68	13.65	1333.00

Results on CIFAR100, BNN trained on validation set with topology: [512] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	74.28	74.19	9.13	9.14	133.00
0.50	73.83	74.16	10.61	9.89	31.00
1.00	73.64	73.79	13.65	13.45	100.00
1.50	73.98	73.97	14.08	14.03	177.00
2.00	73.48	73.60	15.05	14.79	42.00
2.50	73.86	73.87	14.85	14.81	316.00

Results on CIFAR100, BNN trained on train set with topology: [512-512] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	76.90	76.87	15.91	15.94	562.00
0.50	77.24	77.23	15.29	15.31	2371.00
1.00	0.91	1.00	0.28	0.17	23.00
1.50	76.28	76.29	16.66	16.64	562.00
2.00	76.30	76.30	15.88	15.81	100.00
2.50	76.31	76.33	15.51	15.49	562.00

Results on CIFAR100, BNN trained on validation set with topology: [512-512] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	74.02	73.99	11.30	11.09	56.00
0.50	73.08	73.12	12.46	12.40	421.00
1.00	1.08	1.00	0.33	0.27	42.00
1.50	72.52	72.52	15.95	15.91	316.00
2.00	73.01	73.03	15.97	15.95	2371.00
2.50	72.80	72.78	16.12	16.14	1778.00

Results on CIFAR100, BNN trained on train set with topology: [1024] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	77.77	77.76	14.42	14.42	749.00
0.50	77.91	77.88	13.58	13.59	562.00
1.00	76.93	76.91	14.75	14.75	1333.00
1.50	77.11	77.07	15.32	15.28	100.00
2.00	77.49	77.53	13.72	13.67	5623.00
2.50	77.55	77.54	13.98	14.00	10000.00

Results on CIFAR100, BNN trained on validation set with topology: [1024] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	74.46	74.49	10.19	10.15	2371.00
0.50	74.07	74.21	10.60	10.38	177.00
1.00	73.94	73.94	13.27	13.26	5623.00
1.50	73.70	73.65	14.43	14.48	1000.00
2.00	73.95	73.94	14.43	14.39	562.00
2.50	73.74	73.76	15.51	15.17	23.00

Results on CIFAR100, BNN trained on train set with topology: [1024-1024] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	76.87	76.96	16.59	16.48	237.00
0.50	77.08	77.11	15.95	15.92	5623.00
1.00	76.26	76.30	17.80	17.72	100.00
1.50	76.28	76.35	16.24	16.18	237.00
2.00	76.58	76.67	15.88	15.76	74.00
2.50	76.86	76.88	15.08	15.06	749.00

Results on CIFAR100, BNN trained on validation set with topology: [1024-1024] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	74.02	74.02	12.98	12.95	316.00
0.50	73.38	73.38	13.08	13.09	4216.00
1.00	1.04	1.00	0.23	0.27	4216.00
1.50	72.83	72.81	16.04	16.06	7498.00
2.00	73.03	72.98	16.20	16.25	5623.00
2.50	72.99	73.08	16.39	16.08	42.00

Results on CIFAR100, BNN trained on train set with topology: [2048] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	77.55	77.54	15.17	15.18	1000.00
0.50	77.41	77.35	15.25	15.31	562.00
1.00	77.10	77.09	15.30	15.30	421.00
1.50	76.61	76.71	16.03	15.87	133.00
2.00	77.12	77.17	14.41	14.38	10000.00
2.50	77.23	77.24	14.75	14.74	1778.00

Results on CIFAR100, BNN trained on validation set with topology: [2048] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	74.51	74.53	11.09	11.04	562.00
0.50	74.40	74.40	10.82	10.83	7498.00
1.00	73.93	73.89	13.60	13.63	316.00
1.50	74.17	74.21	13.80	13.75	562.00
2.00	73.90	73.92	14.71	14.68	5623.00
2.50	74.04	74.06	15.03	15.03	5623.00

Results on CIFAR100, BNN trained on train set with topology: [2048-2048] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	76.88	76.84	17.24	17.26	237.00
0.50	76.96	76.98	16.74	16.71	2371.00
1.00	76.15	76.11	18.76	18.67	56.00
1.50	76.79	76.75	16.37	16.40	421.00
2.00	76.78	76.78	16.37	16.35	177.00
2.50	76.61	76.62	15.62	15.60	10000.00

Results on CIFAR100, BNN trained on validation set with topology: [2048-2048] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	73.70	73.70	14.05	14.03	1333.00
0.50	73.73	73.78	13.77	13.70	237.00
1.00	1.10	1.00	0.29	0.27	133.00
1.50	73.13	73.13	16.49	16.42	100.00
2.00	73.25	73.20	16.20	16.23	1333.00
2.50	73.23	73.19	16.32	16.36	2371.00

Results on CARS, BNN trained on train set with topology: [] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	87.63	87.83	1.89	1.36	133.00
0.50	87.70	87.75	1.92	1.83	4216.00
1.00	86.43	86.48	1.90	1.38	42.00
1.50	86.42	86.53	1.98	2.00	316.00
2.00	86.55	86.61	2.25	2.08	100.00
2.50	86.35	86.38	2.77	2.56	42.00

Results on CARS, BNN trained on validation set with topology: [] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	81.53	82.77	7.60	8.78	31.00
0.50	80.66	83.13	8.88	11.51	13.00
1.00	78.69	81.80	7.57	11.00	10.00
1.50	79.84	81.98	6.96	9.37	13.00
2.00	80.68	82.06	6.35	8.01	31.00
2.50	80.27	81.88	5.23	6.95	17.00

Results on CARS, BNN trained on train set with topology: [128] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	84.16	84.25	3.33	3.24	421.00
0.50	82.09	82.04	5.69	5.74	1333.00
1.00	29.26	29.60	3.67	3.60	17.00
1.50	81.98	81.86	7.97	8.10	2371.00
2.00	82.59	82.50	7.72	7.79	749.00
2.50	81.96	81.96	8.53	8.50	421.00

Results on CARS, BNN trained on validation set with topology: [128] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	74.71	76.43	4.13	4.84	13.00
0.50	73.03	73.00	2.09	2.20	316.00
1.00	27.04	27.71	3.31	3.27	23.00
1.50	70.83	70.86	5.24	5.04	133.00
2.00	70.63	70.61	6.07	6.02	2371.00
2.50	70.20	70.33	6.89	6.73	562.00

Results on CARS, BNN trained on train set with topology: [128-128] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	0.97	0.95	0.16	0.14	13.00
0.50	0.95	0.95	0.12	0.12	10000.00
1.00	0.95	0.95	0.12	0.12	10000.00
1.50	0.95	0.95	0.11	0.11	10000.00
2.00	0.95	0.95	0.11	0.11	3162.00
2.50	0.95	0.95	0.11	0.11	10000.00

Results on CARS, BNN trained on validation set with topology: [128-128] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	0.41	0.53	0.47	0.09	10.00
0.50	0.56	0.54	0.08	0.09	1333.00
1.00	0.56	0.53	0.08	0.11	10000.00
1.50	23.86	23.83	2.87	2.70	2371.00
2.00	27.91	28.79	3.81	4.16	23.00
2.50	33.50	33.94	4.89	5.12	42.00

Results on CARS, BNN trained on train set with topology: [512] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	86.47	86.47	3.00	2.63	100.00
0.50	84.15	84.18	4.64	4.60	1778.00
1.00	83.08	83.05	6.30	6.33	4216.00
1.50	83.62	83.52	6.84	6.93	421.00
2.00	83.47	83.52	7.80	7.75	1000.00
2.50	83.00	83.11	7.35	7.21	177.00

Results on CARS, BNN trained on validation set with topology: [512] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	78.59	79.00	4.07	4.09	56.00
0.50	76.09	76.09	2.12	2.68	562.00
1.00	73.33	73.92	3.69	2.66	56.00
1.50	73.82	74.33	4.28	3.53	42.00
2.00	74.64	74.68	3.63	3.60	316.00
2.50	74.12	74.38	5.12	4.59	74.00

Results on CARS, BNN trained on train set with topology: [512-512] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	84.81	84.97	5.80	5.48	31.00
0.50	84.72	84.81	5.42	5.36	133.00
1.00	0.95	0.95	0.12	0.12	1333.00
1.50	82.73	82.75	7.55	7.58	316.00
2.00	83.28	83.29	7.75	7.74	10000.00
2.50	83.18	83.11	7.94	7.94	237.00

Results on CARS, BNN trained on validation set with topology: [512-512] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	75.05	75.60	4.13	3.42	56.00
0.50	75.09	75.46	4.16	3.55	177.00
1.00	0.53	0.49	0.11	0.15	10000.00
1.50	74.76	74.74	5.37	5.35	3162.00
2.00	74.40	74.66	6.18	5.90	133.00
2.50	75.97	76.01	4.23	4.49	316.00

Results on CARS, BNN trained on train set with topology: [1024] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	86.40	86.83	3.83	3.13	23.00
0.50	85.35	85.36	3.80	3.77	421.00
1.00	82.96	83.16	7.42	7.16	100.00
1.50	83.75	83.93	7.06	6.88	1333.00
2.00	84.46	84.44	6.91	6.88	421.00
2.50	83.72	83.72	7.02	7.01	7498.00

Results on CARS, BNN trained on validation set with topology: [1024] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	78.56	79.69	2.35	3.00	17.00
0.50	77.70	77.59	1.77	2.00	2371.00
1.00	75.22	75.23	2.39	2.31	5623.00
1.50	75.20	75.07	3.56	3.68	421.00
2.00	75.88	75.91	3.28	3.35	1778.00
2.50	76.30	76.27	3.50	3.53	10000.00

Results on CARS, BNN trained on train set with topology: [1024-1024] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	85.30	85.48	5.98	5.75	177.00
0.50	84.56	84.56	6.48	6.49	5623.00
1.00	0.95	0.95	0.12	0.12	5623.00
1.50	83.34	83.34	7.17	7.16	7498.00
2.00	84.11	84.02	7.10	7.05	74.00
2.50	83.85	83.83	7.06	7.07	562.00

Results on CARS, BNN trained on validation set with topology: [1024-1024] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	76.16	75.93	4.52	4.70	237.00
0.50	76.35	76.42	4.75	4.67	2371.00
1.00	0.58	0.54	0.06	0.09	10000.00
1.50	75.83	75.88	4.65	4.56	1000.00
2.00	76.12	76.06	4.42	4.48	10000.00
2.50	77.31	77.32	4.01	4.02	5623.00

Results on CARS, BNN trained on train set with topology: [2048] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	86.66	86.70	4.21	4.15	562.00
0.50	85.79	85.86	3.96	3.89	177.00
1.00	83.46	83.65	6.92	6.71	133.00
1.50	84.16	84.16	6.94	6.89	237.00
2.00	83.95	84.00	7.69	7.64	421.00
2.50	84.79	84.89	6.62	6.49	177.00

Results on CARS, BNN trained on validation set with topology: [2048] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	80.04	80.28	1.99	1.96	56.00
0.50	78.44	78.61	1.74	1.64	133.00
1.00	75.27	75.28	3.39	3.33	1333.00
1.50	76.01	76.11	3.49	3.16	56.00
2.00	77.27	77.29	2.98	2.97	10000.00
2.50	75.55	75.88	5.89	5.30	56.00

Results on CARS, BNN trained on train set with topology: [2048-2048] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	85.10	85.00	6.82	6.85	100.00
0.50	85.31	85.35	6.33	6.29	1000.00
1.00	0.95	0.95	0.10	0.12	74.00
1.50	83.06	83.05	8.05	8.06	7498.00
2.00	84.13	84.25	6.78	6.67	562.00
2.50	83.87	83.82	7.02	7.06	4216.00

Results on CARS, BNN trained on validation set with topology: [2048-2048] and DAR divergence

α	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)	Best K
0.10	75.81	76.25	6.30	5.64	56.00
0.50	76.53	76.58	5.49	5.57	1000.00
1.00	0.54	0.53	0.11	0.11	1333.00
1.50	77.01	77.03	4.42	4.36	4216.00
2.00	75.51	76.11	6.85	5.55	23.00
2.50	75.43	75.66	6.81	6.59	421.00

Appendix C

Results Divergences Comparison

In this appendix we show the results of all the experiments regarding the comparison between divergences. For a reference of the chosen configurations see Table 5.6.

C.1 Results on CIFAR10

C.2 Results on CIFAR100

C.3 Results on CARS

Divergences comparison on CIFAR10 with state-of-the-art performing configuration

Divergence	β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)
D_{KL}	200	95.47	95.49	1.27	1.32
D_{KL}	450	95.32	95.46	0.42	1.05
D_{KL}	1000	95.30	95.48	1.79	2.17
$D_{\mathrm{AR}}(\alpha = 0.1)$	200	95.30	95.31	1.78	1.77
$D_{\mathrm{AR}}(\alpha = 0.5)$	200	95.28	95.24	1.55	1.50
$D_{\mathrm{AR}}(\alpha = 1.0)$	200	95.35	95.36	1.47	1.44
$D_{\mathrm{AR}}(\alpha = 1.5)$	200	95.28	95.39	1.84	1.79
$D_{\mathrm{AR}}(\alpha = 2.0)$	200	95.39	95.41	2.02	2.07
$D_{\mathrm{AR}}(\alpha = 2.5)$	200	95.33	95.40	2.23	2.13
$D_{\mathrm{AR}}(\alpha = 0.1)$	450	95.31	95.30	1.24	1.26
$D_{\mathrm{AR}}(\alpha = 0.5)$	450	95.33	95.34	0.83	0.67
$D_{\mathrm{AR}}(\alpha = 1.0)$	450	95.18	95.40	0.56	1.22
$D_{\mathrm{AR}}(\alpha = 1.5)$	450	95.38	95.37	0.94	1.00
$D_{\mathrm{AR}}(\alpha = 2.0)$	450	95.35	95.36	1.35	1.31
$D_{\mathrm{AR}}(\alpha = 2.5)$	450	95.30	95.37	1.54	1.57
$D_{\mathrm{AR}}(\alpha = 0.1)$	1000	95.21	95.31	0.69	0.69
$D_{\mathrm{AR}}(\alpha = 0.5)$	1000	95.15	95.33	0.81	1.58
$D_{\mathrm{AR}}(\alpha = 1.0)$	1000	95.10	95.39	1.63	2.20
$D_{\mathrm{AR}}(\alpha = 1.5)$	1000	95.08	95.39	0.71	1.59
$D_{\mathrm{AR}}(\alpha = 2.0)$	1000	95.12	95.40	0.57	1.13
$D_{\mathrm{AR}}(\alpha = 2.5)$	1000	95.36	95.40	0.65	1.07

Divergences comparison on CIFAR10 with poor performing configuration ($\beta = 1000$).

Divergence	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)
D_{KL}	95.07	95.44	3.66	4.29
$D_{\mathrm{AR}}(\alpha = 0.1)$	95.27	95.45	1.85	2.39
$D_{\mathrm{AR}}(\alpha = 0.5)$	94.99	95.45	5.09	5.93
$D_{\mathrm{AR}}(\alpha = 1.0)$	95.00	95.43	3.40	4.07
$D_{\mathrm{AR}}(\alpha = 1.5)$	95.06	95.44	2.25	2.81
$D_{\mathrm{AR}}(\alpha = 2.0)$	95.29	95.46	1.77	2.07
$D_{\mathrm{AR}}(\alpha = 2.5)$	95.21	95.45	1.12	1.69

Divergences comparison on CIFAR100 with state-of-the-art performing configuration

Divergence	β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)
D_{KL}	100	77.36	77.44	3.46	3.33
D_{KL}	200	77.21	77.15	1.23	1.43
D_{KL}	450	75.89	76.87	4.97	5.50
$D_{\mathrm{AR}}(\alpha = 0.1)$	100	0.94	1.03	0.19	0.09
$D_{\mathrm{AR}}(\alpha = 0.5)$	100	77.53	77.55	3.38	3.35
$D_{\mathrm{AR}}(\alpha = 1.0)$	100	77.20	77.25	3.83	3.71
$D_{\mathrm{AR}}(\alpha = 1.5)$	100	77.53	77.44	4.83	4.91
$D_{\mathrm{AR}}(\alpha = 2.0)$	100	77.45	77.50	5.63	5.57
$D_{\mathrm{AR}}(\alpha = 2.5)$	100	77.60	77.61	6.12	6.11
$D_{\mathrm{AR}}(\alpha = 0.1)$	200	0.92	1.01	0.20	0.11
$D_{\mathrm{AR}}(\alpha = 0.5)$	200	77.68	77.74	1.88	1.87
$D_{\mathrm{AR}}(\alpha = 1.0)$	200	77.43	77.46	0.87	0.99
$D_{\mathrm{AR}}(\alpha = 1.5)$	200	77.16	77.20	2.21	2.19
$D_{\mathrm{AR}}(\alpha = 2.0)$	200	77.34	77.42	3.34	3.26
$D_{\mathrm{AR}}(\alpha = 2.5)$	200	77.62	77.62	3.83	3.82
$D_{\mathrm{AR}}(\alpha = 0.1)$	450	78.53	78.48	1.90	2.12
$D_{\mathrm{AR}}(\alpha = 0.5)$	450	77.21	77.81	2.10	2.09
$D_{\mathrm{AR}}(\alpha = 1.0)$	450	76.42	77.12	4.65	5.71
$D_{\mathrm{AR}}(\alpha = 1.5)$	450	77.04	77.30	2.36	3.14
$D_{\mathrm{AR}}(\alpha = 2.0)$	450	77.05	77.42	1.79	1.72
$D_{\mathrm{AR}}(\alpha = 2.5)$	450	77.52	77.43	0.96	0.81

Divergences comparison on CIFAR100 with poor performing configuration ($\beta = 2000$).

Divergence	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)
D_{KL}	72.21	76.73	19.38	25.47
$D_{\mathrm{AR}}(\alpha = 0.1)$	77.52	78.63	2.48	3.41
$D_{\mathrm{AR}}(\alpha = 0.5)$	77.53	78.32	4.98	6.46
$D_{\mathrm{AR}}(\alpha = 1.0)$	71.66	76.71	18.26	25.14
$D_{\mathrm{AR}}(\alpha = 1.5)$	73.51	76.84	13.66	18.29
$D_{\mathrm{AR}}(\alpha = 2.0)$	74.11	77.00	9.82	13.88
$D_{\mathrm{AR}}(\alpha = 2.5)$	74.86	77.38	8.23	11.21

Divergences comparison on CARS with state-of-the-art performing configuration

Divergence	β	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)
D_{KL}	10	82.47	82.54	3.43	3.36
D_{KL}	20	82.34	82.35	1.66	1.59
D_{KL}	40	82.06	82.40	2.20	2.34
$D_{\mathrm{AR}}(\alpha = 0.1)$	10	86.42	86.65	2.56	2.15
$D_{\mathrm{AR}}(\alpha = 0.5)$	10	84.76	84.64	1.55	1.63
$D_{\mathrm{AR}}(\alpha = 1.0)$	10	83.03	83.01	3.48	3.20
$D_{\mathrm{AR}}(\alpha = 1.5)$	10	82.31	82.32	4.93	4.90
$D_{\mathrm{AR}}(\alpha = 2.0)$	10	82.73	82.68	5.49	5.54
$D_{\mathrm{AR}}(\alpha = 2.5)$	10	82.16	82.01	6.07	6.18
$D_{\mathrm{AR}}(\alpha = 0.1)$	20	86.60	86.70	2.47	2.42
$D_{\mathrm{AR}}(\alpha = 0.5)$	20	84.39	84.48	2.39	2.43
$D_{\mathrm{AR}}(\alpha = 1.0)$	20	83.37	83.31	1.81	1.94
$D_{\mathrm{AR}}(\alpha = 1.5)$	20	82.73	82.68	2.52	2.59
$D_{\mathrm{AR}}(\alpha = 2.0)$	20	82.40	82.42	4.28	4.23
$D_{\mathrm{AR}}(\alpha = 2.5)$	20	82.57	82.72	4.51	4.43
$D_{\mathrm{AR}}(\alpha = 0.1)$	40	87.39	87.34	4.09	4.17
$D_{\mathrm{AR}}(\alpha = 0.5)$	40	83.31	84.76	3.51	3.49
$D_{\mathrm{AR}}(\alpha = 1.0)$	40	81.58	82.83	2.69	2.45
$D_{\mathrm{AR}}(\alpha = 1.5)$	40	82.11	82.85	2.27	1.88
$D_{\mathrm{AR}}(\alpha = 2.0)$	40	82.39	82.42	2.55	2.18
$D_{\mathrm{AR}}(\alpha = 2.5)$	40	82.52	82.50	2.91	2.94

Divergences comparison on CARS with poor performing configuration ($\beta = 450$).

Divergence	Acc (best K)	Acc ($K = 20000$)	ECE (best K)	ECE ($K = 20000$)
D_{KL}	73.79	80.92	14.57	23.07
$D_{\mathrm{AR}}(\alpha = 0.1)$	79.77	82.86	9.01	12.43
$D_{\mathrm{AR}}(\alpha = 0.5)$	83.62	86.29	9.83	13.14
$D_{\mathrm{AR}}(\alpha = 1.0)$	74.66	80.12	14.47	22.04
$D_{\mathrm{AR}}(\alpha = 1.5)$	77.44	81.61	10.97	16.33
$D_{\mathrm{AR}}(\alpha = 2.0)$	79.13	82.65	8.76	12.85
$D_{\mathrm{AR}}(\alpha = 2.5)$	80.13	83.14	7.27	10.46