Bayesian Inference with Certifiable Adversarial Robustness

Matthew Wicker[*]
Department of Computer Science
University of Oxford

Luca Laurenti[*]
Department of Computer Science
University of Oxford

Andrea Patane[*]
Department of Computer Science
University of Oxford

Zhoutong Chen
Department of ECE
Univ. of California, Santa Barbara

Zheng Zhang
Department of ECE
Univ. of California, Santa Barbara

Marta Kwiatkowska
Department of Computer Science
University of Oxford

Abstract

We consider adversarial training of deep neural networks through the lens of Bayesian learning, and present a principled framework for adversarial training of Bayesian Neural Networks (BNNs) with certifiable guarantees. We rely on techniques from constraint relaxation of non-convex optimisation problems and modify the standard cross-entropy error model to enforce posterior robustness to worst-case perturbations in ϵ-balls around input points. We illustrate how the resulting framework can be combined with methods commonly employed for approximate inference of BNNs. In an empirical investigation, we demonstrate that the presented approach enables training of certifiably robust models on MNIST, FashionMNIST and CIFAR-10 and can also be beneficial for uncertainty calibration. Our method is the first to directly train certifiable BNNs, thus facilitating their deployment in safety-critical applications.

1 INTRODUCTION

Although deep neural networks (NNs) have achieved state-of-the-art performance in a range of learning tasks (Goodfellow et al., 2016), they have recently been shown to be susceptible to adversarial attacks: small, often imperceptible perturbations to their inputs that can trigger a misclassification (Goodfellow et al., 2014; Biggio and Roli, 2018). While retaining the flexibility of standard (deterministic) NNs, Bayesian neural networks (BNNs), i.e., neural networks with distributions placed over their weights and biases, enable principled quantification of their predictions' uncertainty (Neal, 2012). Intuitively, the latter can be used to provide a natural protection against adversarial examples, making BNNs particularly appealing for safety-critical scenarios, in which the safety of the system must be provably guaranteed (McAllister et al., 2017). In fact, not only have BNNs been shown to possess many favorable robustness properties against adversarial attacks (Bekasov and Murray, 2018; Carbone et al., 2020), but their uncertainty has also been investigated as a means of flagging out-of-distribution samples and for robust decision making (Kahn et al., 2017).

However, while guarantees for BNNs have been provided for the true Bayesian posterior and under idealised conditions on the training data and network architecture, the necessary assumptions cannot be checked in practice and exact inference for BNNs is generally infeasible. Indeed, it has been shown that BNNs trained with modern approximate inference methods and on real-world datasets can be easily fooled by adversarial attacks (Grosse et al., 2018; Athalye et al., 2018). Consequently, BNN methodologies need to be strengthened before they can be deployed in practical safety-critical scenarios. However, to the best of our knowledge, there is no general Bayesian approach targeted at the training of BNNs with certifiable robustness against adversarial attacks.

In this paper, we present a principled Bayesian approach for incorporating adversarial robustness in the posterior inference procedure of BNNs. In order to do so, we formulate the robustness requirement as the worst-case prediction over an adversarial input ball of radius $\epsilon \geq 0$ induced by a user-defined probability density function p_ϵ, and extend the standard cross-entropy likelihood model by marginalising the network output over p_ϵ. We refer to the resulting likelihood model as the *robust likelihood*. We show how, for

Proceedings of the 24th International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130. Copyright 2021 by the author(s). * - Authors contributed equally.

any $\epsilon > 0$, certified lower bounds to the robust likelihood can be computed by employing Interval Bound Propagation (IBP) techniques (Ehlers, 2017; Mirman et al., 2018; Gowal et al., 2018). We further demonstrate that, based on a differentiable modification of the likelihood model, the adversarial training procedure we introduce adapts naturally to the main approximate inference techniques employed for training of BNNs, including HMC (Neal et al., 2011), Bayes by Backprop (BBB) (Blundell et al., 2015), Stochastic Weight Averaging - Gaussian (SWAG) (Maddox et al., 2019), NoisyAdam (NA) (Zhang et al., 2018a), and Variational Online Gauss-Newton (VOGN) (Osawa et al., 2019).

We experimentally investigate the suitability of our method for training certifiably robust BNN models on the MNIST, FashionMNIST and CIFAR-10 datasets. We find that training with the robust likelihood enables the first non-trivial adversarial robustness certification on BNNs. In particular, we obtain certified bounds for the robust accuracy of up to 75% on the MNIST test set for $\epsilon = 0.1$, up to 73% on the FashionMNIST test set for $\epsilon = 0.1$, and up to 50% on the CIFAR-10 test set for $\epsilon = 1/255$. We find the maximum certified safe radius when training with the robust likelihood to be double that obtained when training with the standard likelihood model, which suggests that the robust likelihood pushes the BNN posterior distribution toward more robust regions of the parameter space. Furthermore, we analyse the effect that the robust likelihood has on the predictive uncertainty. We qualitatively observe better calibrated uncertainty when predicting out-of-distribution samples compared with standard training.

In summary, this paper makes the following main contributions:

- We introduce a robust likelihood function based on an adversarial generalisation of the cross-entropy error model for the training of certifiably robust BNNs.
- We demonstrate how IBP techniques can be employed to compute a certified lower bound to the robust likelihood for commonly employed approximate Bayesian inference techniques.
- We show how our methods allow us to train, for the first time, certified BNNs on MNIST, FashionMNIST and CIFAR-10 with non-trivial robustness. We empirically find that it also leads to improvements in the calibration of uncertainty.[1]

Related Works Robustness to adversarial examples has been a central topic for both machine learning theorists and practitioners since the seminal work of Szegedy et al. (2013). Since their popularization (Biggio and Roli, 2018), many methods have been developed to generate adversarial attacks for neural networks (Goodfellow et al., 2014; Madry et al., 2017; Carlini and Wagner, 2016) and also specifically for BNNs (Carbone et al., 2020; Yuan et al., 2021). In addition to this, several other methods have been proposed for computing guarantees on the absence of adversarial attacks in the neighbourhood of a given test point (Katz et al., 2017; Boopathy et al., 2019), including works that focus on quantifying the robustness to adversarial perturbations of BNNs (Cardelli et al., 2019a; Wicker et al., 2020) and Gaussian processes (Cardelli et al., 2019b; Blaas et al., 2020).

Adversarial training, on the other hand, seeks to directly train neural networks that are robust to them (Madry et al., 2017). Most of the adversarial training methods have been developed for deterministic (i.e., non-Bayesian) neural networks, where the common approach is that of solving a min-max optimization problem obtained by modifying the loss. Based on a similar approach, (Liu et al., 2018) have developed a method for adversarial training of BNNs trained with Gaussian Variational Inference (VI). However, this method cannot be directly extended to other approximate inference algorithms and relies on gradient-based attacks (i.e., PGD (Madry et al., 2017)) to approximate worst-case perturbations in the neighbourhood of each data point, which, as we show in Section 5, may fail to generalize to other attacks. Similarly, Ye and Zhu (2018), assumes a prior distribution around each data point, which is then used to sample adversarial examples. However, by assuming a distribution over the adversarial examples, the resulting approach is not worst-case. In contrast, the framework presented in this paper is based on a principled Bayesian foundation (i.e., we modify the standard likelihood model to account for adversarial examples). As a consequence, it can be formulated directly for any approximate inference method. Furthermore, by explicitly maximizing a lower bound of the robust likelihood instead of an approximation obtained by employing gradient-based attacks, it allows us to obtain non-trivial certifiable worst-case guarantees for BNNs as illustrated in Section 5.

2 BAYESIAN INFERENCE WITH NEURAL NETWORKS

We consider a generic neural network (NN) architecture, $f^w : \mathbb{R}^n \mapsto \mathbb{R}^C$, parameterised by a vector of weights $w \in \mathbb{R}^{n_w}$, where C is the number of classes.[2] In a Bayesian setting, one assumes a prior distribution over the weights, $p(w)$, that induces a distribution over the network outputs. Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n_\mathcal{D}}$ our belief

[1]All of the source code to reproduce the experiments can be found at https://github.com/matthewwicker/CertifiableBayesianInference. All the experiments were run on a single NVIDIA 2080Ti GPU with a 20-core Intel Core Xeon 6230.

[2]We discuss how to generalise to the regression case in Remark 1

over the weight distribution is updated through Bayes' theorem, so as to obtain the posterior weight distribution as $p(w|\mathcal{D}) \propto p(\mathcal{D}|w)p(w)$. In the latter equation, $p(\mathcal{D}|w)$ is the data *likelihood* and, under the assumption that data in $\mathcal{D}$ are drawn independently from the same distribution, we have $p(\mathcal{D}|w) = \prod_{i=1}^{n_{\mathcal{D}}} p(y_i|x_i, w)$. In the standard classification model, the likelihood for a point (x_i, y_i), i.e., $p(y_i|x_i, w)$, is defined as the multinoulli distribution with the probability for each class given as the softmax of the neural network final logits (Bishop et al., 1995). In what follows, we will refer to thus defined $p(y_i|x_i, w)$ as the *standard likelihood*. In Section 3 we will discuss how the standard likelihood can be modified in order to model adversarial perturbations. Given the posterior, $p(\mathcal{D}|w)$, the predictive distribution over a test point x^* is computed as the expected value over the softmax output. In practice, this is computed via an empirical estimator:

$$\mathbb{E}_{w \sim p(w|\mathcal{D})}[\sigma(f^w(x))] \approx \hat{\mathbb{E}}^N(x) := \frac{1}{N}\sum_{i=1}^{N} \sigma\left(f^{w_i}(x)\right), \qquad (1)$$

where $w_1, \dots, w_N$ are N random samples from $p(w|\mathcal{D})$ and $\sigma(\cdot)$ is the softmax function. The output class for a test point x^* is then computed as $\arg\max_{c \in \{1,\dots,C\}} \hat{\mathbb{E}}^N(x^*)$.

Given the predictor, a local adversarial example for the BNN at a given test point x^* with associated class label c^* is defined as a point $\bar{x} \in \mathbb{R}^n$ such that $|x^* - \bar{x}| \leq \epsilon$ and $\arg\max_{c \in \{1,\dots,C\}} \hat{\mathbb{E}}^N(\bar{x}) \neq c^*$, where $|\cdot|$ is a given norm, that is, a point in the neighborhood of x^* that gets misclassified by the BNN predictor. Given a set of test points $\{x_i^*\}_{i=1,\dots,m}$, we define the *robust accuracy* (denoted $\mathcal{R}_\epsilon$) for $\epsilon > 0$ as the ratio between the number of points x_i^* for which no adversarial example exists within ϵ radius, and the total number of test points, that is:

$$\mathcal{R}_\epsilon = \frac{1}{m}\sum_{i=1}^{m} \mathbb{I}\Bigg[\forall \bar{x}\ s.t.\ |x_i^* - \bar{x}| \leq \epsilon, \quad \arg\max_{c \in \{1,\dots,C\}} \hat{\mathbb{E}}^N(\bar{x}) = c_i^*\Bigg], \qquad (2)$$

where $\mathbb{I}[\cdot]$ evaluates to 1 if the expression inside the brackets is true, and to zero otherwise.

Note that the computation of the predictive distribution relies on the computation of the posterior distribution over the weights, $p(w|\mathcal{D})$. Exact computation of the latter for BNNs is infeasible and approximate methods are employed for its estimation. In the following, we briefly describe those employed in this paper.

Hamiltonian Monte Carlo Hamiltonian Monte Carlo (HMC) approximates the posterior by defining a Markov chain whose stationary distribution is $p(w|\mathcal{D})$, and relies on Hamiltonian dynamics to improve efficiency of the exploration. This is achieved by alternating between sampling from the potential function $U(w) = -\log(p(w))$, and moving around the weight space by following the dynamics described by a kinetic function, $K(v) = \sum_{i=1}^{n_w} v_i^2/(2m_i)$, given over the auxiliary variable v. The hyper-parameters m_i trade-off exploration with exploitation of the weight space (Neal et al., 2011).

Despite its scalability issues, HMC is considered to be the gold standard of Bayesian inference for neural networks. This is due to the fact that, in the limit of the number of simulations of the Markov chain, the HMC approximation converges to the true posterior distribution.

Variational Inference (VI) Variational methods assume that the posterior can be approximated by a distribution belonging to a given parametric family, that is: $p(w|\mathcal{D}) \approx q(w|\theta)$. The inference problem then boils down to the that of computing a parameterisation θ that minimises a given variational objective. While finding the optimal θ does not guarantee convergence to the true posterior of the BNN, variational methods have significant scalability advantages (Osawa et al., 2019). A number of variational methods were developed in the literature. We briefly review the ones that we employ in this paper below. Blundell et al. (2015) introduced Bayes by Backprop (BBB), a stochastic gradient method to update the parameters of the variational distribution using KL divergence. VI methods with improved scalability have been obtained by relying on the natural gradient (i.e. the gradient w.r.t. sampled weights) in order to update the variational parameters. This, in conjunction with momentum, has been shown in many instances to be an efficient method for performing VI at scale, as implemented by VOGN (Lin et al., 2020) and NA (Zhang et al., 2018a). A method presented by Maddox et al. (2019) (SWAG) generates a variational distribution by moment matching a Gaussian distribution to stochastic gradient descent iterates.

3 A BAYESIAN APPROACH FOR ADVERSARIAL TRAINING

In this section we present a method for the adversarial training of BNNs that relies on embedding adversarial robustness in the standard cross-entropy error model. We first introduce the following notation. Given a data point (x, y), with $x \in \mathbb{R}^n$ and y an associated class label, an $\epsilon > 0$, and a fixed weight $w \in \mathbb{R}^{n_w}$, we define $f_{\min}^{w,\epsilon}(x)$ to be the vector of logits corresponding to the minimizer of the softmax of class y for any input point in an $\epsilon-$ball around x:

$$\sigma_y(f_{\min}^{w,\epsilon}(x)) = \min_{x' : |x - x'| \leq \epsilon} \sigma_y(f^w(x')), \qquad (3)$$

where $\sigma_y(\cdot)$ is the softmax relative to class y. In the robust training scenario we would like the likelihood $p(y|x,w)$ to be influenced not only by the accuracy of the prediction, but also by the robustness of the BNN in the neighbourhood of a point. We model this by assuming that the likelihood of a prediction, given an input point and a weight vector, also depends on the worst-case perturbation in an $\epsilon-$ball around the input point. In particular, we assume that ϵ is sampled from a non-negative random variable with distribution p_ϵ, and for each ϵ in the support of p_ϵ we consider the softmax likelihood computed for $f^{w,\epsilon}_{\text{min}}(x)$. Then, by marginalizing over p_ϵ, we obtain the following likelihood function, which we call *robust likelihood*:

$$\begin{aligned} p(y|x,w) = &\int_{\mathbb{R}_{\geq 0}} \sigma_y(f^{w,\epsilon}_{\text{min}}(x)) p_\epsilon(\epsilon) d\epsilon \\ =&\mathbb{E}_{\epsilon \sim p_\epsilon}[\sigma_y(f^{w,\epsilon}_{\text{min}}(x))]. \end{aligned} \tag{4}$$

That is, in the robust likelihood for each ϵ we consider the worst-case of the standard likelihood for all the points in an ϵ-ball around x (note that $\sigma_y(f^{w,\epsilon}_{\text{min}}(x)$ is a monotonically decreasing function of ϵ) and then compute the average with respect to p_ϵ. We should stress that Eqn. (4) defines a marginal probability, and hence is a well defined probability. Furthermore, we also note that for $p_\epsilon = \delta_0$, the delta function centered in the origin, we recover the standard cross-entropy likelihood model.

In the particular case of deterministic NNs, adversarial training is generally implemented by balancing between adversarial robustness at a fixed $\epsilon > 0$ and accuracy (that is, at $\epsilon = 0$) (Goodfellow et al., 2014; Lakshminarayanan et al., 2017). For $0 \leq \lambda \leq 1$ and $\eta > 0$, this can be obtained in our setting by considering the following discrete distribution for ϵ:

$$p_\epsilon(\epsilon) = \begin{cases} \lambda & \text{if } \epsilon = 0 \\ 1-\lambda & \text{if } \epsilon = \eta \end{cases}. \tag{5}$$

This leads to the following form for the robust likelihood:

$$p(y|x,w) = \lambda \cdot \sigma_y(f^w(x)) + (1-\lambda) \cdot \sigma_y(f^{w,\eta}_{\text{min}}(x)),$$

which is a weighted sum of two softmax functions, one given by the standard likelihood and the other accounting for adversarial robustness.

By assuming the statistical independence of the training labels given input and weights (which is the standard assumption for classification (Bishop et al., 1995)), we obtain the following negative log-likelihood for our model:

$$E = -\sum_{i=1}^{n_\mathcal{D}} log\big(\mathbb{E}_{\epsilon \sim p_\epsilon}[\sigma_{y^{(i)}}(f^{w,\epsilon}_{\text{min}}(x^{(i)}))]\big). \tag{6}$$

Notice that this has a trivial absolute minimum when $\mathbb{E}_{\epsilon \sim p_\epsilon}[\sigma_{y^{(i)}}(f^{w,\epsilon}_{\text{min}}(x^{(i)}))] = 1$ for all $(x^{(i)}, y^{(i)}) \in \mathcal{D}$. Hence, the absolute minimum of the negative log-likelihood (which would correspond to the maximum likelihood estimation) is reached for the set of weights w^*, if it exists, such that for any $(x^{(i)}, y^{(i)}) \in \mathcal{D}$ almost surely $f^{w^*}(x^{(i)})$ has no adversarial examples in an $\epsilon-$ball around $x^{(i)}$, for any ϵ in the support of p_ϵ.

Therefore, in order to evaluate our robust likelihood model we need to be able to compute $\sigma_y(f^{w,\epsilon}_{\text{min}}(x))$ as defined by Eqn. (3). This is discussed in the next section. First, we briefly discuss the case of regression models.

Remark 1. *The above analysis concerns a classification framework. For a regression problem everything follows similarly except that the likelihood is a Gaussian distribution with variance Σ (Bishop et al., 1995). In particular, assuming for simplicity and without lost of generality that $C = 1$ – i.e., single output regression – call $f^{w,\epsilon}_{\text{max}}(x) = \max_{x':|x-x'|\leq\epsilon} f^w(x')$ and $f^{w,\epsilon}_{\text{min}}(x) = \min_{x':|x-x'|\leq\epsilon} f^w(x')$. That is, $f^{w,\epsilon}_{\text{max}}(x)$ and $f^{w,\epsilon}_{\text{min}}(x)$ are the maximum and minimum of f^w for all the points in an $\epsilon-$ball centered around x. Then, in the regression case, for $x \in \mathbb{R}^n, y \in \mathbb{R}$ the robust likelihood is:*

$$\begin{aligned} p(y|x,w) = \frac{1}{\sqrt{2\pi\Sigma}} exp\big(- \frac{1}{2\Sigma} \max\{ \\ (\mathbb{E}_{\epsilon \sim p_\epsilon}[f^{w,\epsilon}_{\text{max}}(x)] - y)^2, (\mathbb{E}_{\epsilon \sim p_\epsilon}[f^{w,\epsilon}_{\text{min}}(x)] - y)^2\}\big). \end{aligned}$$

4 ADVERSARIAL TRAINING THROUGH INTERVAL BOUND PROPAGATION

We now describe our implementation of the robust likelihood introduced in Section 3. For a data point (x,y), we need to compute $\sigma_y(f^{w,\epsilon}_{\text{min}}(x))$, that is, the minimum of the softmax of class y for all the points in an $\epsilon-$ball around x. Notice that, the likelihood function, $p(y|x,w)$, is defined over given values of the weight vector w (and input point x), so that, even in the Bayesian settings, we need to deal with a single deterministic NN (sampled from the BNN) at a time. Unfortunately still, the computation of this quantity poses an NP-complete problem for deterministic NNs (Katz et al., 2017). In this section, we review how Interval Bound Propagation (IBP) can be used to compute a safe lower bound on $\sigma_y(f^{w,\epsilon}_{\text{min}}(x))$ (Ehlers, 2017; Mirman et al., 2018; Gowal et al., 2018) and then discuss extensions needed to provide certification for BNN posterior predictions.

Interval Bound Propagation IBP works by propagating bounding boxes through the network's layers. Given a weight vector w, the computation through the network layers on an input point x is denoted by:

$$\begin{aligned} f^w_{(k)} &= W^w_{k-1}\phi^w_{(k-1)} + b^w_{k-1} & k &= 1,\dots,K \\ \phi^w_{(k)} &= h(f^w_{(k)}) & k &= 1,\dots,K-1, \end{aligned}$$

with $\phi^w_0 = x$, where K is the number of layers, W^w_k and b^w_k are the weight matrix and bias vector corresponding to the weight vector w for the kth layer of the network,

h is the activation function, $f^w_{(k)}$ is the kth pre-activation and $\phi^w_{(k)}$ is the kth activation vector. Notice that $f^w_{(K)}$ is the network latent vector, whose softmax provides the final output of the network. Following the procedure discussed by Gowal et al. (2018), we consider an $\ell_\infty-$ball of radius ϵ in the input space that we denote as $[\phi_0^{w,L}, \phi_0^{w,U}]$. We then iteratively propagate the latter through each layer, for $k = 1, \dots, K$. This can be done efficiently by the introduction of the auxiliary variables, $\hat{\mu}_k, \mu_k, \hat{r}_k$ and r_k, as it follows:

$$\begin{aligned}
\hat{\mu}_k &= \frac{\phi^{w,U}_{(k-1)} + \phi^{w,L}_{(k-1)}}{2}, & \hat{r}_k &= \frac{\phi^{w,U}_{(k-1)} - \phi^{w,L}_{(k-1)}}{2} \\
\mu_k &= W^w_{k-1}\hat{\mu}_k + b^w_{k-1}, & r_k &= |W^w_{k-1}|\hat{r}_k \\
f^{w,U}_{(k)} &= \mu_k + r_k, & f^{w,L}_{(k)} &= \mu_k - r_k \\
\phi^{w,U}_{(k)} &= h(f^{w,U}_{(k)}), & \phi^{w,L}_{(k)} &= h(f^{w,L}_{(k)}).
\end{aligned}$$

The resulting latent values for $k = K$, i.e., $f^{w,U,\epsilon} := f^{w,U}_{(K)}$ and $f^{w,L,\epsilon} := f^{w,L}_{(K)}$, yield a valid bounding box for the network output, that is, they are such that $f^w(x) \in [f^{w,L,\epsilon}, f^{w,U,\epsilon}]$ for every x in the considered $\epsilon-$ball (Gowal et al., 2018).

Bounding the Robust Likelihood Consider a generic input point x with associated class y, and an $\epsilon > 0$ from the support of p_ϵ. Let $f^{w,L,\epsilon}$ and $f^{w,U,\epsilon}$ be the extrema of the bounding box computed by IBP as described in the previous paragraph. For $j = 1, \dots, C$, we define the vector $f^{w,\epsilon}_{LB}(x)$ as follows:

$$f^{w,\epsilon}_{LB,j}(x) = \begin{cases} f^{w,U,\epsilon}_j & \text{if } j \neq y \\ f^{w,L,\epsilon}_j & \text{if } j = y. \end{cases} \tag{7}$$

Since the softmax function for the class y is monotonically increasing along the y coordinate and monotonically decreasing along any other direction, we have that $\sigma_y(f^{w,\epsilon}_{LB}(x)) \leq \sigma_y(f^{w,\epsilon}_{\min}(x))$. That is, $\sigma_y(f^{w,\epsilon}_{LB}(x))$ provides a lower bound for $\sigma_y(f^{w,\epsilon}_{\min}(x))$. By propagating this bound through the computation of the robust likelihood, we obtain the following proposition

Proposition 1. *Consider a point x with class y, and a distribution over ϵ, p_ϵ. Given a weight vector w we have that:*

$$p(y|x,w) \geq \mathbb{E}_{\epsilon\sim p_\epsilon}[\sigma_y(f^{w,\epsilon}_{LB}(x))] := p_{IBP}(y|x,w).$$

Proof Sketch. For any ϵ in the support of p_ϵ we have, by the construction described above, that $\sigma_y(f^{w,\epsilon}_{\min}(x)) \geq \sigma_y(f^{w,\epsilon}_{LB}(x))$. Hence, since expected values respect inequalities, we have that:

$$\begin{aligned} p(y|x,w) &= \mathbb{E}_{\epsilon\sim p_\epsilon}[\sigma_y(f^{w,\epsilon}_{\min}(x))] \geq \mathbb{E}_{\epsilon\sim p_\epsilon}[\sigma_y(f^{w,\epsilon}_{LB}(x))] \\ &= p_{\text{IBP}}(y|x,w). \end{aligned}$$

□

Proposition 1 guarantees that the robust likelihood function can be lower bounded by using IBP computations. This provides us with a worst-case analysis for adversarial attacks, in the sense that if the network is safe according to IBP then it is necessarily safe (while the converse is not true). In fact, while $p(y|x,w)$ could be approximated by means of adversarial attack methods (e.g., FGSM or PGD), the resulting approximation would not have any guarantees with respect to the actual values of the $p(y|x,w)$. Since $p_{\text{IBP}}(y|x,w)$ is non-negative and a formal lower bound to $p(y|x,w)$, by employing it we obtain a (non-negative) lower bound to Equation (6) as well, so that the property of absence of adversarial examples at the optimal value of $p(y|x,w)$ is maintained by $p_{\text{IBP}}(y|x,w)$. However, the same does not hold for an approximation of $p(y|x,w)$ given by adversarial attacks, similarly to how maximising a lower bound yields guarantees on the final result, whereas nothing can be said about the maximisation of an upper bound. In Section 5 we empirically compare IBP with PGD and find these observations to be confirmed in practice.

Certification of BNNs After training, we can employ IBP to compute the certified robust accuracy of the BNN on a test set. In order to do so, instead of performing IBP on the single extracted network from the BNN, we have to propagate the input bounding box through the posterior predictor. To see that, consider a predictor for the BNN output $\hat{\mathbb{E}}^N$ defined as in Eqn. (1). Let $w_1, \dots, w_N$ be the sampled weights related to the latter, consider a given test point x with true label y, and let $\epsilon > 0$ be the adversarial perturbation strength that we want to provide certification against. By performing IBP N times we obtain a family of lower bounds such that $\sigma_y(f^{w_i,\epsilon}_{LB}(x)) \leq \sigma_y(f^{w_i,\epsilon}_{\min}(x))$, for $i = 1, \dots, N$. By averaging these out we obtain a lower bound on the given empirical predictor for class y in x, which we denote as $\hat{\mathbb{E}}^{N,\epsilon}_{y,LB} := \frac{1}{N}\sum \sigma_y(f^{w_i,\epsilon}_{LB}(x))$. Similarly, we can proceed to compute an upper bound on the predictor for classes $c' \neq y$. For this purpose we define:

$$f^{w,\epsilon,c'}_{UB,j}(x) = \begin{cases} f^{w,L,\epsilon}_j & \text{if } j \neq c' \\ f^{w,U,\epsilon}_j & \text{if } j = c', \end{cases}$$

which we average out to obtain an upper bound on the predictive probability for class c' as $\hat{\mathbb{E}}^{N,\epsilon}_{c',UB} := \frac{1}{N}\sum \sigma_{c'}(f^{w_i,\epsilon,c'}_{UB}(x))$. By defining the worst-case predictor for the true class y according to IBP for x as the vector $\hat{\mathbb{E}}^N_{IBP,\epsilon}(x) \in \mathbb{R}^C$ such that its $y-$th entry is equal to $\hat{\mathbb{E}}^{N,\epsilon}_{y,LB}$, while the other entries for $c' \neq y$ are set to $\hat{\mathbb{E}}^{N,\epsilon}_{c',UB}$, we obtain the following theorem.

Theorem 1. *Consider a point x^* with associated class y^*. Assume that* $\arg\max_{c\in\{1,\dots,C\}} \hat{\mathbb{E}}^N_{IBP,\epsilon}(x^*) = y^*$, *then:*

$$\arg\max_{c\in\{1,\dots,C\}} \hat{\mathbb{E}}^N(\bar{x}) = y^* \quad \forall \bar{x} \;\; s.t. \;\; |x^* - \bar{x}| \leq \epsilon.$$

As a consequence, given a set of test points x_i^* with associated class labels y_i^*, for $i = 1, \dots, m$, we have the following certified bound for the robust accuracy at $\epsilon > 0$:

$$\mathcal{R}_\epsilon \geq \frac{1}{m}\sum_{i=1}^{m} \mathbb{I}\left[\arg\max_{c \in \{1,\dots,C\}} \hat{\mathbb{E}}^N_{IBP,\epsilon}(x_i^*) = y_i^*\right] := \mathcal{R}_\epsilon^{IBP}.$$

Proof Sketch. By construction we have that, $\forall \bar{x}$ s.t. $|x^* - \bar{x}| \leq \epsilon$ it holds that:

$$\hat{\mathbb{E}}^N_{IBP,\epsilon}(x^*)_{y^*} \leq \hat{\mathbb{E}}^N_\epsilon(\bar{x})_{y^*}, \quad \hat{\mathbb{E}}^N_{IBP,\epsilon}(x^*)_c \quad \geq \hat{\mathbb{E}}^N_\epsilon(\bar{x})_c,$$

for $c \neq y^*$, $c \in \{1, \dots, C\}$. Hence, if the $\arg\max$ of $\hat{\mathbb{E}}^N_{IBP,\epsilon}(x^*)$ is y^*, then necessary the $\arg\max$ of $\hat{\mathbb{E}}^N_\epsilon(\bar{x})$ will be y^* as well (of course, the converse is not necessarily true). This can be re-stated as:

$$\mathbb{I}\left[\forall \bar{x} \ s.t. \ |x_i^* - \bar{x}| \leq \epsilon, \arg\max_{c \in \{1,\dots,C\}} \hat{\mathbb{E}}^N(\bar{x}) = c_i^*\right]$$
$$\geq \mathbb{I}\left[\arg\max_{c \in \{1,\dots,C\}} \hat{\mathbb{E}}^N_{IBP,\epsilon}(x_i^*) = y_i^*\right].$$

The theorem statement then follows by the definitions of $\mathcal{R}_\epsilon$ and $\mathcal{R}_\epsilon^{IBP}$. □

Theorem 1 ensures that a lower bound on the certified robust accuracy of the BNN can be computed by employing IBP on a given predictor. Observe that, while the guarantees when computing the likelihood in Proposition 1 are given for a single weight realisation, so that robust training has the effect of penalising at inference time the weights which are not robust, the certified robust accuracy is computed for an empirical average of predictions over weights sampled from the posterior. We remark that, while in this paper we only considered training with IBP, linear bound propagation techniques (LBP) (Zhang et al., 2018b) could be similarly employed. LBP could potentially yield better bounds at the price of a less efficient training process.

Remark 2. *The certified bounds are provided for a given empirical predictor, and as shown in Theorem 1, if the empirical predictor is known, the bounds are exact. In the case in which the set of weights $w_1, \dots, w_N$ changes at every prediction, one can rely on standard concentration inequalities, such as Chernoff's bound, to obtain confidence intervals over the certified behaviour (Vapnik, 2013).*

4.1 APPROXIMATE INFERENCE WITH IBP

As mentioned in Section 2, unfortunately, training BNNs is an intractable problem, and remains so when the robust likelihood is used, so that the exact posterior cannot be computed. Hence, even if a model that perfectly minimises Equation (6) were to exist, we would not be guaranteed to find it (similarly to why no optimality can be claimed when training with the standard likelihood). Crucially, however, the likelihood bound computed by IBP is differentiable. This allows us to adapt commonly used approximate Bayesian inference methods to the robust likelihood settings. We now give the details for selected variational and Monte Carlo methods. We remark that, when computing the certified robust accuracy, we assume that the model has already been trained, so that the inference method does not introduce any error in the certification that we provide.

ROBUST VI In Algorithm 1 we highlight the changes needed for the implementation of the robust likelihood in the case of natural gradient variational inference. We maintain the standard form of VI (Lin et al., 2020), except for lines 6-9, highlighted in red, that emphasize the changes needed in the case in which p_ϵ takes the form of Eqn. (8). The case of a general p_ϵ can be tackled by iteratively sampling from the distribution (i.e., by adding a for loop around lines 6-9), the cost of this will, however, be an increased computational time. Notice, that since the IBP bound is differentiable, so is l defined in line 9 of the algorithm. We remark that, by changing the parameter update on line 10 with approximations to the Hessian, computing the gradient wrt μ, s, or by introducing momentum parameters, this algorithm can be converted to any of the gradient and natural gradient variational inference methods which have been proposed in recent years, including those of Graves (2011); Blundell et al. (2015); Khan et al. (2018) and Osawa et al. (2019).

Algorithm 1 Robust Natural Grad. Variational Inference

Input: Prior mean and precision: $\mu_{\text{prior}}, s_{\text{prior}}$, NN architecture: f, Dataset: $\mathcal{D}$, Learning rate: α, Iterations: T, Mini-batch Size: m, ϵ and λ parameters of p_ϵ.
Output: Mean and precision of Gaussian approximate posterior.

```
 1: $s \leftarrow s_{\text{prior}}$; $\mu \leftarrow \mu_{\text{prior}}$
 2: for $t = 1, \dots, T$ do
 3:     $\{X, Y\} \leftarrow \{x_i, y_i\}_{i=0}^m$ {Sample Batch}
 4:     $w = \mu + ((n_{\mathcal{D}} s)^{-1/2} \mathcal{N}(0, I))$
 5:     $Y_{\text{clean}} \leftarrow \sigma(f^w(X))$
 6:     $f^{w,L,\epsilon}(X), f^{w,U,\epsilon}(X) \leftarrow \text{IBP}(f, w, X)$
 7:     $f^{w,\epsilon}_{LB}(x) \leftarrow$ Eqn. (7) for $f^{w,L,\epsilon}(X), f^{w,U,\epsilon}(X)$
 8:     $Y_{\text{worst}} \leftarrow \sigma(f^{w,\epsilon}_{LB}(x))$
 9:     $l \leftarrow -Y log(\lambda Y_{\text{clean}} + (1 - \lambda) Y_{\text{worst}})$
            $+ \mathbb{D}_{\text{KL}}(\mathcal{N}(\mu_{\text{prior}}, 1/s_{\text{prior}}) \mid \mathcal{N}(\mu, 1/s))$
10:     $s \leftarrow (1 - \alpha)s + \alpha \nabla^2_w l$;   $\mu \leftarrow \mu - \alpha s^{-1} \nabla_w l$
11: end for
12: return $(\mu, s)$
```

ROBUST HMC A similar modification needs to be made in the case of Hamiltonian Monte Carlo inference. When computing the potential energy function the same

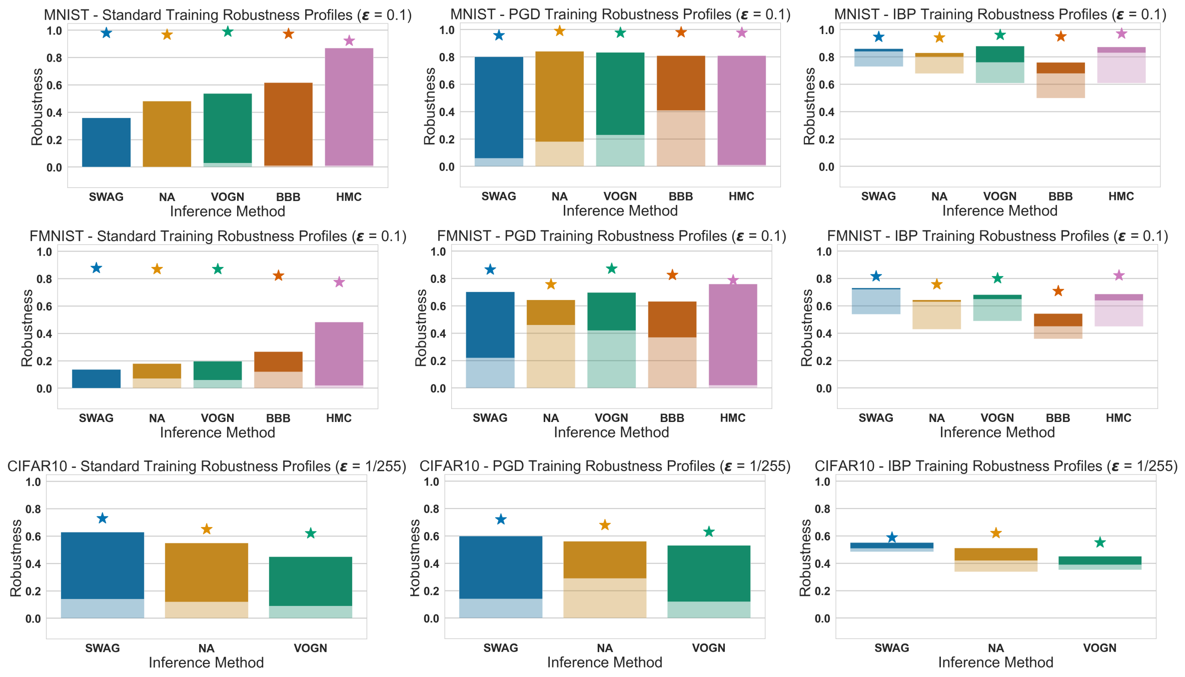

Figure 1: Accuracy (plotted as star points), an empirical estimation of $\mathcal{R}_\epsilon$ obtained using PGD (upper bound of each bar), $\mathcal{R}_\epsilon^{LBP}$ (lower bound of each bar), and $\mathcal{R}_\epsilon^{IBP}$ (shaded lower bound of each bar) obtained for $\epsilon = 0.1$ on the MNIST dataset (top row) and FMNIST (middle row) as well as for $\epsilon = 1/255$ on the CIFAR-10 dataset (bottom row). Each bar refers to a different approximate Bayesian inference technique. **Left Column:** results for the standard likelihood. **Centre Column:** results for approximation of robust likelihood using PGD. **Right Column:** results for training with formal IBP lower bound of robust likelihood (Eq (6)). With our method we obtain up to 75% certified robust accuracy on MNIST and up to 50% on CIFAR-10.

procedure outlined in lines 6-9 is employed. Namely, we have $U(w) = -log(p(w)) - log(\lambda Y_{\text{clean}} + (1-\lambda)Y_{\text{worst}})$, where the vectors Y_{clean} and Y_{worst} are defined as in Algorithm 1. Observe that the kinetic function given in Section 2 remains unaltered, as it is not related to the weight distribution.

5 EXPERIMENTS

We conduct an empirical evaluation of our framework considering various approximate inference methods for BNNs, including SWAG (Maddox et al., 2019), NoisyAdam (NA) (Zhang et al., 2018a), Variational Online Gauss Newton (VOGN) (Khan et al., 2018), Bayes by Backprop (BBB) (Blundell et al., 2015), and Hamiltonian Monte Carlo (HMC) (Neal et al., 2011).

We first evaluate the robust accuracy of networks trained with our robust likelihood compared to standard training on the MNIST benchmark (LeCun et al., 1998). We then perform analogous analyses on the CIFAR-10 (Krizhevsky et al., 2009) dataset, and, finally, we empirically study the effect of our robust likelihood on the predictive uncertainty. Further, training parameters, including prior distribution and architecture for each of the BNNs, can be found in the Supplementary Material.

For HMC, we set the initial weight to be a sample from the prior when performing standard training and set the initial weight to a pre-trained SGD iterate when performing inference with robust likelihood; this is to enforce that the starting point of the algorithm is closer to the target distribution.

EVALUATION ON MNIST We use p_ϵ introduced in Eqn (8) with $\eta = 0.1$ and $\lambda = 0.25$ (an empirical evaluation of the effect of changing these parameters, as well as using different forms for p_ϵ, is reported in the Supplementary Material). We train a single hidden layer BNN with 512 neurons on the full MNIST dataset (further training parameters are given in the Supplementary Material).

In Figure 1 (top row) we analyze how different training methods affect the accuracy, robustness to PGD attacks ($\mathcal{R}_{0.1}^{PGD}$), as well as the certified lower bounds using IBP ($\mathcal{R}_{0.1}^{IBP}$) and LBP ($\mathcal{R}_{0.1}^{LBP}$). We use both IBP as well as the more computationally expensive but tighter LBP in order to study the effect of training with our robust likelihood without the bias of training and evaluating with the same certification method. Moreover, in Figure 2 we analyze the maximum certifiable radius (via the methodology in (Boopathy et al., 2019)) in order to further compare the effects of each likelihood.

In Figure 1, we find that, while all BNNs trained with the standard likelihood (left plot) perform comparably well in terms of accuracy, there is a marked difference in their robustness against PGD. This is in line with what was observed by Carbone et al. (2020), where the more fidelity an inference method has to the true Bayesian posterior, the greater is its robustness to gradient-based attacks. However, the certified robust accuracy obtained using standard likelihood is identically zero, that is, we obtain no certification in these settings. This implies, for example, that although HMC is resistant to PGD attacks, we cannot guarantee that a different, successful attack method does not exist. Similarly, for training with the PGD approximation of the robust likelihood (central plot), we obtain that, while the robustness against PGD of each model is now around 80%, the certified robustness is still at 0%. Similar behaviour has also been observed for adversarial training with gradient-based attacks for deterministic neural networks (Gowal et al., 2018). By using IBP during training to lower bound the robust likelihood (right plot in the figure), we find that, not only do we obtain similar levels of accuracy and PGD robustness as before, but we are also able to provide non-trivial certification on the robust accuracy, $\mathcal{R}_\epsilon$, of the networks, that is, against *any* possible adversarial perturbation of magnitude up to $\epsilon = 0.1$. For example, using SWAG we obtain $\mathcal{R}_{0.1} \approx 75\%$, that is, the BNN trained with SWAG with our robust likelihood is provably adversarially robust on 75% of the points included in the MNIST dataset.

Evaluation on FashionMNIST In the center row of Figure 1, we use the same networks, p_ϵ distribution (with $\eta = 0.1$ and $\lambda = 0.25$) and evaluation methods stated for MNIST, but applied to the FashionMNIST dataset (Xiao et al., 2017). Despite being a harder dataset than MNIST, as evidenced by the reduced accuracy of the approximate posteriors, we find the robustness trends to be qualitatively similar to those on MNIST and CIFAR10. We do note, however, that PGD training was much more effective at increasing the certified bound when the bound is computed with LBP. Our hypothesis about this difference is that because the networks are less accurate they are also potentially less robust to gradient-based attacks (following the logic of (Carbone et al., 2020)) and thus, PGD is able to find strong adversarial examples which can increase the robustness of the posterior.

EVALUATION ON CIFAR-10 We now evaluate the effect of the robust likelihood on BNNs trained on the CIFAR-10 dataset. The CIFAR-10 is more challenging compared to MNIST, and hence not all the training methods considered for MNIST can be used to train reasonably

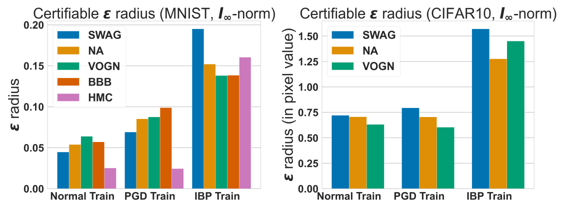

Figure 2: We plot the average certified radius for images from MNIST (right), and CIFAR-10 (left) using the methods of Boopathy et al. (2019). We observe that robust training with IBP roughly doubles the maximum verifiable radius compared with standard training and that obtained by training on PGD adversarial examples.

accurate BNNs on this dataset (Blier and Ollivier, 2018). Consequently, for CIFAR-10 we provide results only for SWAG, NA and VOGN. In particular, we train a Bayesian convolutional neural network (CNN) with 2 convolutional layers (each with 32, 4 by 4 filters) followed by a max pooling layer and 2 fully connected layers (one with 512 hidden neurons and the other with 10). For the robust likelihood, we consider Eqn (8) with $\eta = 1/255$ and $\lambda = 0.25$. Finally, we introduce the standard exponential decay on the learning rate to ensure stable convergence (results for additional parameter values are given in the Supplementary Material).

We perform a similar evaluation to that discussed for MNIST, the results of which are plotted in the bottom row of Figure 1. Consistently with what we observed for MNIST, we obtain that BNNs trained by using the standard likelihood (left plot) and PGD attacks (central plot) do not allow for the computation of certified guarantees (the lower bound of the bars is close to zero for all the inference methods). In contrast, for the BNN trained with our robust likelihood and IBP we find that, even for CIFAR-10, we are able to compute non-trivial lower bounds on $\mathcal{R}_\epsilon$. For instance, on SWAG we obtain $\mathcal{R}_\epsilon^{IBP} \approx 50\%$, which is comparable to state-of-the-art results with adversarial training of deterministic NNs on CIFAR-10 (Boopathy et al., 2019). We analyse the effect that robust training at a specific ϵ has on the robustness of the network at other values of ϵ. In order to do so, we employ the method developed by Boopathy et al. (2019), which, for each test image, computes the maximal adversarial perturbation, ϵ, such that the image is provably safe in the related $\epsilon-$ball. The results of this analysis are given in Figure 2, where we report the average maximum adversarially safe radius over 100 test CIFAR-10 images. We find that, while PGD training does not increase the robustness of the model compared to standard training, our training procedure is able to roughly double the robustness for the three training methods explored here (further results on MNIST are in the Supplementary Material).

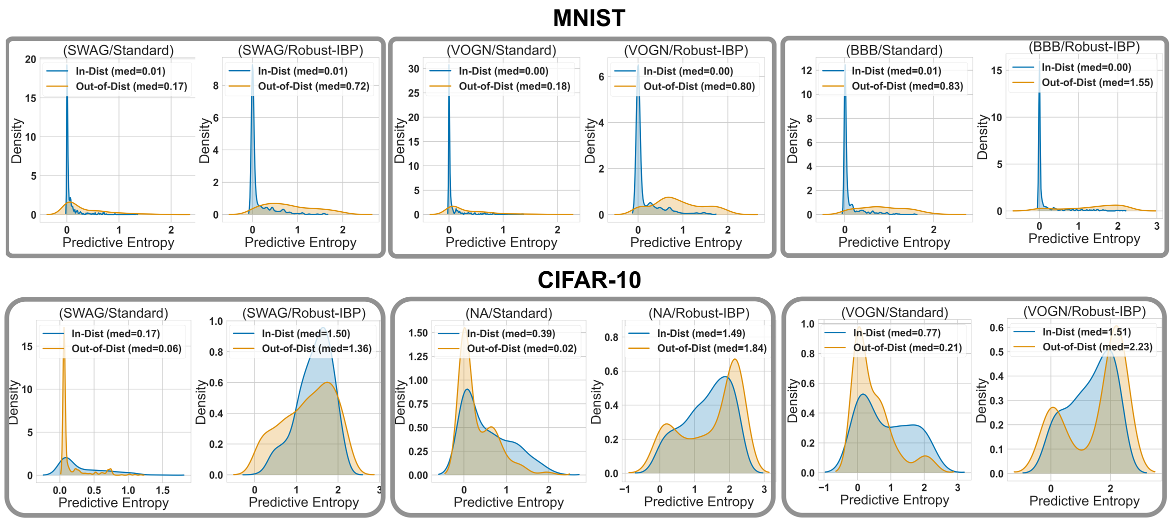

Figure 3: We plot the in-distribution (blue) and out-of-distribution (orange) predictive uncertainty. Each pair of plots corresponds to an inference method where the left plot represents the entropy distribution for standard training and the right plot represents robust IBP training. For both MNIST (top row) and CIFAR-10 (bottom row) we find that robust training improves the uncertainty calibration of the network w.r.t. out-of-distribution samples.

EVALUATION ON OUT-OF-DISTRIBUTION UNCERTAINTY One of the main features of BNNs, which distinguishes them from DNNs, is their ability to model their prediction's uncertainty (Neal, 2012). We investigate how training with our robust likelihood influences uncertainty calibration when testing on data that are outside of the training data distribution. As is common in the literature (Maddox et al., 2019; Lakshminarayanan et al., 2017), we evaluate uncertainty on in-distribution and out-of-distribution images using the entropy of the posterior predictive distribution. For $x \in \mathbb{R}^n$ the latter is defined as $\mathbb{E}_{w\sim p(w|\mathcal{D})}[-\sum_{c=1}^{C} \sigma_c(f^w(x)) \log(\sigma_c(f^w(x)))]$. In particular, we expect that the maximal class probabilities on out-of-distribution images will have high entropy, reflecting the model's uncertainty in its predictions, and considerably lower entropy on images that are similar to those on which the network was trained (in-distribution). In the top row of Figure 3, we plot the entropy of the posterior predictive distributions on in-distribution test samples (in blue) and on FashionMNIST samples (in orange), which are out-of-distribution data for networks trained on the MNIST dataset. For CIFAR-10 (bottom row in the figure) we use the SVHN (Goodfellow et al., 2013) dataset as the source of meaningful, but out-of-distribution, data. We hypothesize that if the robust likelihood introduces meaningful information about invariances in the data, then the uncertainty of the resulting posterior would be improved. On MNIST, we find that, in each case, robust training with IBP significantly improves uncertainty by making more uncertain predictions on out-of-distributions data, while still being confident on in-distribution data. For CIFAR-10, surprisingly, we find that BNNs trained with the standard likelihood are more confident on the out-of-distribution data than on in-distributions data, a behaviour that, with the exception of SWAG, is always reversed by performing robust training with IBP. In all the cases, training with IBP on CIFAR-10 increases the entropy of the predictions, which may be indicative of better calibrated uncertainty compared to the normally inferred models (note that on CIFAR-10 accuracy is of the order of 60%, and hence we do not expect the model to be very confident even on in-distribution images). In the Supplementary Materials, to further confirm our empirical results, we evaluate the entropy on adversarial examples and also analyze the likelihood ratio between in-distribution and out-distribution data.

6 CONCLUSION

We presented a framework for robust training of certifiably robust deep neural networks based on a principled Bayesian foundation. We developed an algorithmic implementation of our framework that employs constraint relaxation and can be integrated with existing approximate inference methods for BNNs. On the MNIST, FashionMNIST and CIFAR-10 datasets we showed that, not only does our framework allow us to train BNNs that are guaranteed to be robust to adversarial examples, it can also have a positive effect on uncertainty calibration.

Acknowledgements This project was funded by the ERC under the European Union's Horizon 2020 research and innovation programme (FUN2MODEL, grant agreement No. 834115). ZC and ZZ are supported by DOE grant No. DE-SC0021323.

References

Athalye, A., Carlini, N., and Wagner, D. (2018). Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*.

Bekasov, A. and Murray, I. (2018). Bayesian adversarial spheres: Bayesian inference and adversarial examples in a noiseless setting. *arXiv preprint arXiv:1811.12335*.

Biggio, B. and Roli, F. (2018). Wild patterns: Ten years after the rise of adversarial machine learning. *Pattern Recognition*, 84:317–331.

Bishop, C. M. et al. (1995). *Neural networks for pattern recognition*. Oxford university press.

Blaas, A., Laurenti, L., Patane, A., Cardelli, L., Kwiatkowska, M., and Roberts, S. (2020). Robustness quantification for classification with Gaussian processes. *AISTATS*.

Blier, L. and Ollivier, Y. (2018). The description length of deep learning models. In *Advances in Neural Information Processing Systems*, pages 2216–2226.

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

Boopathy, A., Weng, T.-W., Chen, P.-Y., Liu, S., and Daniel, L. (2019). Cnn-cert: An efficient framework for certifying robustness of convolutional neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3240–3247.

Carbone, G., Wicker, M., Laurenti, L., Patane, A., Bortolussi, L., and Sanguinetti, G. (2020). Robustness of bayesian neural networks to gradient-based attacks. *arXiv preprint arXiv:2002.04359*.

Cardelli, L., Kwiatkowska, M., Laurenti, L., Paoletti, N., Patane, A., and Wicker, M. (2019a). Statistical guarantees for the robustness of bayesian neural networks. *IJCAI*.

Cardelli, L., Kwiatkowska, M., Laurenti, L., and Patane, A. (2019b). Robustness guarantees for Bayesian inference with Gaussian processes. In *AAAI*, volume 33, pages 7759–7768.

Carlini, N. and Wagner, D. (2016). Towards Evaluating the Robustness of Neural Networks. *arXiv e-prints*, page arXiv:1608.04644.

Ehlers, R. (2017). Formal verification of piece-wise linear feed-forward neural networks. In *International Symposium on Automated Technology for Verification and Analysis*, pages 269–286. Springer.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Goodfellow, I. J., Bulatov, Y., Ibarz, J., Arnoud, S., and Shet, V. (2013). Multi-digit number recognition from street view imagery using deep convolutional neural networks. *arXiv preprint arXiv:1312.6082*.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*.

Gowal, S., Dvijotham, K., Stanforth, R., Bunel, R., Qin, C., Uesato, J., Arandjelovic, R., Mann, T., and Kohli, P. (2018). On the effectiveness of interval bound propagation for training verifiably robust models. *arXiv preprint arXiv:1810.12715*.

Graves, A. (2011). Practical variational inference for neural networks. In *Advances in neural information processing systems*, pages 2348–2356.

Grosse, K., Pfaff, D., Smith, M. T., and Backes, M. (2018). The limitations of model uncertainty in adversarial settings. *arXiv:1812.02606*.

Kahn, G., Villaflor, A., Pong, V., Abbeel, P., and Levine, S. (2017). Uncertainty-aware reinforcement learning for collision avoidance. abs/1702.01182.

Katz, G., Barrett, C., Dill, D. L., Julian, K., and Kochenderfer, M. J. (2017). Reluplex: An efficient SMT solver for verifying deep neural networks. In *CAV*, pages 97–117. Springer.

Khan, M. E., Nielsen, D., Tangkaratt, V., Lin, W., Gal, Y., and Srivastava, A. (2018). Fast and scalable bayesian deep learning by weight-perturbation in adam. *arXiv preprint arXiv:1806.04854*.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. In *NeurIPS*, pages 6402–6413.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lin, W., Schmidt, M., and Khan, M. E. (2020). Handling the positive-definite constraint in the bayesian learning rule. *arXiv preprint arXiv:2002.10060*.

Liu, X., Li, Y., Wu, C., and Hsieh, C.-J. (2018). Adv-bnn: Improved adversarial defense through robust bayesian neural network. *arXiv preprint arXiv:1810.01279*.

Maddox, W. J., Izmailov, P., Garipov, T., Vetrov, D. P., and Wilson, A. G. (2019). A simple baseline for bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems*, pages 13153–13164.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. (2017). Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv e-prints*.

McAllister, R., Gal, Y., Kendall, A., Van Der Wilk, M., Shah, A., Cipolla, R., and Weller, A. (2017). Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning. International Joint Conferences on Artificial Intelligence, Inc.

Mirman, M., Gehr, T., and Vechev, M. (2018). Differentiable abstract interpretation for provably robust neural networks. In *International Conference on Machine Learning*, pages 3578–3586. PMLR.

Neal, R. M. (2012). *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media.

Neal, R. M. et al. (2011). Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2.

Osawa, K., Swaroop, S., Khan, M. E. E., Jain, A., Eschenhagen, R., Turner, R. E., and Yokota, R. (2019). Practical deep learning with bayesian principles. In *Advances in neural information processing systems*, pages 4287–4299.

Sutskever, I., Martens, J., Dahl, G., and Hinton, G. (2013). On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv e-prints*, page arXiv:1312.6199.

Vapnik, V. (2013). *The nature of statistical learning theory*. Springer science & business media.

Wicker, M., Laurenti, L., Patane, A., and Kwiatkowska, M. (2020). Probabilistic safety for bayesian neural networks. *UAI 2020*.

Xiao, H., Rasul, K., and Vollgraf, R. (2017). Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*.

Ye, N. and Zhu, Z. (2018). Bayesian adversarial learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 6892–6901.

Yuan, M., Wicker, M., and Laurenti, L. (2021). Gradient-free adversarial attacks for bayesian neural networks. *Advances in Approximate Bayesian Inference 2021*.

Zhang, G., Sun, S., Duvenaud, D., and Grosse, R. (2018a). Noisy natural gradient as variational inference. In *International Conference on Machine Learning*, pages 5852–5861.

Zhang, H., Weng, T.-W., Chen, P.-Y., Hsieh, C.-J., and Daniel, L. (2018b). Efficient neural network robustness certification with general activation functions. In *NeurIPS*, pages 4939–4948.