BEING BAYESIAN, EVEN JUST A BIT, FIXES OVERCONFIDENCE IN RELU NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

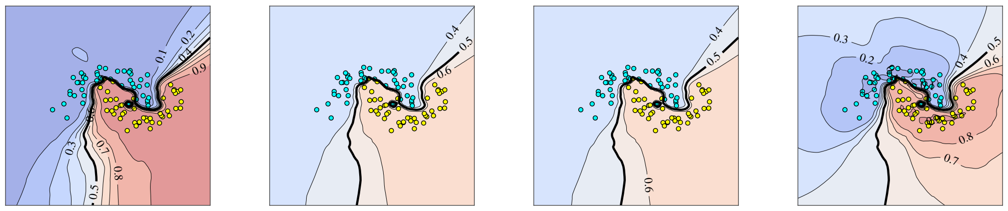
The point estimates of ReLU classification networks, arguably the most widely used neural network architecture, have recently been shown to have arbitrarily high confidence far away from the training data. This architecture is thus not robust, e.g., against out-of-distribution data. Approximate Bayesian posteriors on the weight space have been empirically demonstrated to improve predictive uncertainty in deep learning. The theoretical analysis of such Bayesian approximations is limited, including for ReLU classification networks. We present an analysis of approximate Gaussian posterior distributions on the weights of ReLU networks. We show that even a simplistic (thus cheap), non-Bayesian Gaussian distribution fixes the asymptotic overconfidence issue. Furthermore, when a Bayesian method, even if a simple one, is employed to obtain the Gaussian, the confidence becomes better calibrated. This theoretical result motivates a range of Laplace approximations along a fidelity-cost trade-off. We validate these findings empirically via experiments using common deep ReLU networks.

1 INTRODUCTION

As neural networks have been successfully applied in ever more domains, including safety-critical ones, the robustness of their predictions and the calibration of their predictive uncertainty have moved into focus, subsumed under the notion of AI safety (Amodei et al., 2016). A principal goal of uncertainty calibration is that learning machines (and neural networks in particular) should assign low confidence to test cases not explained well by the training data or prior information (Gal, 2016). The most obvious such instance are test points that lie "far away" from the training data. Many methods to achieve this goal have been proposed, both Bayesian (Gal & Ghahramani, 2016; Blundell et al., 2015; Louizos & Welling, 2017) and non-Bayesian (Lakshminarayanan et al., 2017; Liang et al., 2018; Hein et al., 2019).

ReLU networks are currently among the most widely used neural architectures. This class comprises any network that can be written as a composition of linear layers (including fully-connected, convolutional, and residual layers) and a ReLU activation function. But while ReLU networks often achieve high accuracy, the *uncertainty* of their predictions has been shown to be miscalibrated (Guo et al., 2017). Indeed, Hein et al. (2019) demonstrated that ReLU networks are *always* overconfident "far away from the data": scaling a training point $\mathbf{x}$ (a vector in a Euclidean input space) with a scalar δ yields predictions of arbitrarily high confidence in the limit $\delta \to \infty$. This means ReLU networks are susceptible to adversarial or out-of-distribution (OOD) examples. Bayesian methods have long been known empirically to improve predictive uncertainty calibration. MacKay (1992) demonstrated empirically that the predictive uncertainty of Bayesian neural networks will naturally be high in regions not covered by training data. Results like this raise the hope that the overconfidence problem of ReLU networks, too, might be mitigated by the use of Bayesian methods.

This paper offers a theoretical analysis of the binary classification case of ReLU networks with logistic output layer. We show that equipping such networks with *virtually any* Gaussian probability distribution (i.e. regardless of whether it is motivated in a Bayesian fashion or not) mitigates the aforementioned theoretical problem, so that predictive confidence far away from the training data approaches a known constant, bounded away from one, whose value is controlled by the covariance (cf. Figure 1). At the same time, this treatment does not change the decision boundary of the trained network, so it has no negative effect on the predictive performance.



(a) MAP (b) Isotropic $O(1)$ (c) Diagonal $O(d)$ (d) Laplace $O(d^3)$

Figure 1: Binary classification on a toy dataset using a MAP estimate (a) and various Gaussian approximations over the weights, sorted by their complexity of inverting the precision matrix. These approximations are carried out only at the last layer of the network and d denotes the number of hidden units at that layer. The shade of color represents the confidence of the prediction (darker shade means higher confidence). The decision boundary is in thick black. Even an arbitrary (i.e. non-Bayesian) isotropic (b) or diagonal (c) covariance makes the confidence bounded away from one. Using the data in a more Bayesian fashion (d) calibrates the uncertainty further, in particular in regions close to the data.

A central aspect of our result is that *asymptotic* overconfidence can be mitigated with an essentially arbitrary Gaussian distribution on the weight space, including one of simple diagonal or even scalar covariance, and one whose covariance need not even depend on the training data. Achieving calibration at *finite* distances from the training data requires increasing levels of fidelity towards full Bayesian inference, for which our results also give some quantification. Our results thus answer a question about "how Bayesian" one needs to be to achieve certain levels of calibration. This is valuable because even approximate Bayesian treatments of deep learning, such as through Laplace approximations, can have high computational cost.

We empirically validate our results through a simple Laplace approximation to *only* the last layer of deep ReLU architectures, and find that this cheap procedure is already competitive to recently proposed non-Bayesian methods specifically constructed to overcome the overconfidence problem of ReLU networks. We also show that this cheap Bayesian approach yields good performance in the multi-class classification setting, indicating that our analysis may carry over to this case. Section 2 begins with a rigorous problem statement and assumptions, then develops the main theoretical results. We discuss related work in Section 3, while empirical results are in Section 4.

2 Analysis

2.1 Preliminaries

Definitions We call a function $f : \mathbb{R}^n \to \mathbb{R}$ piecewise affine if there exists a finite set of polytopes $\{Q_r\}_{r=1}^R$, referred to as linear regions of f, such that $\cup_{r=1}^R Q_r = \mathbb{R}^n$ and $f|_{Q_r}$ is an affine function for every Q_r. ReLU networks are networks that result in piecewise affine classifier functions (Arora et al., 2018) which include networks with fully-connected, convolutional, and residual layers where just ReLU or leaky-ReLU are used as activation functions and max or average pooling are used as a convolution layer. Let $\mathcal{D} := \{\mathbf{x}_i \in \mathbb{R}^n, t_i\}_{i=1}^m$ be a dataset, where the targets $t_i \in \{0, 1\}$ or $t_i \in \{1, \ldots, k\}$ for the binary and multi-class case, respectively. We define the logistic (*sigmoid*) function as $\sigma(z) := 1/(1 + \exp(-z))$ for $z \in \mathbb{R}$ and the softmax function as $\text{softmax}(\mathbf{z}, i) := \exp(\mathbf{z}_i)/\sum_j \exp(\mathbf{z}_j)$ for $\mathbf{z} \in \mathbb{R}^k$. Given a linear classifier,[1] we will consider probability distributions $p(\mathbf{w}|\mathcal{D})$ or $p(\mathbf{W}|\mathcal{D})$ over the weight vector and matrix, respectively. We call these distributions *posterior* if they arose from Bayes' theorem or an approximation thereof.

[1]Unless stated otherwise, we assume the bias is absorbed into the weights.

The *predictive* distribution (also called the *marginalized* prediction) is

$$p(y=1|\mathbf{x},\mathcal{D}) := \int \sigma(\mathbf{w}^T\mathbf{x})\, p(\mathbf{w}|\mathcal{D})\, d\mathbf{w} \tag{1}$$

or

$$p(y=i|\mathbf{x},\mathcal{D}) := \int \mathrm{softmax}(\mathbf{W}\mathbf{x}, i)\, p(\mathbf{W}|\mathcal{D})\, d\mathbf{W}\,, \tag{2}$$

for the binary and multi-class cases, respectively. For Euclidean spaces we use the standard inner product and norm. Finally, $\lambda_i(\cdot)$, $\lambda_{\text{max}}(\cdot)$, and $\lambda_{\text{min}}(\cdot)$ return the ith, maximum, and minimum eigenvalue (which are assumed to exist) of their matrix argument, respectively.

Problem statement The following theorem from Hein et al. (2019) shows that ReLU networks exhibit arbitrarily high confidence far away from the training data: If a training point $\mathbf{x} \in \mathbb{R}^n$ is scaled by a sufficiently large scalar $\delta > 0$, the input $\delta\mathbf{x}$ attains arbitrarily high confidence.

Theorem 2.1 (Hein et al. (2019))**.** *Let* $\mathbb{R}^d = \cup_{r=1}^R Q_r$ *and* $f(\mathbf{x}) = \mathbf{V}_r\mathbf{x} + \mathbf{a}_r$ *be the piecewise affine representation of the output of a ReLU network on* Q_r*. Suppose that* $\mathbf{V}_r$ *does not contain identical rows for all* $r = 1, \dots, R$*, then for almost any* $\mathbf{x} \in \mathbb{R}^n$ *and* $\epsilon > 0$ *there exists an* $\delta > 0$ *and a class* $i \in \{1, \dots, K\}$ *such that it holds* $\mathrm{softmax}(f(\delta\mathbf{x}), i) \geq 1 - \epsilon$*. Moreover,* $\lim_{\delta\to\infty} \mathrm{softmax}(f(\delta\mathbf{x}), i) = 1$. $\square$

2.2 Assumptions

For binary classification tasks, it is standard to treat neural networks as probabilistic models of the conditional distribution $p(y|\mathbf{x}, \mathbf{w})$. Standard deep training involves assigning a maximum a posteriori (MAP) value $\mathbf{w}_{\text{MAP}}$ to the weights. Doing so ignores potential uncertainty on $\mathbf{w}$. We will show that this lack of uncertainty is the primary cause of the overconfidence discussed in Hein et al. (2019).

Unfortunately, there is generally no analytic solution for eq. (1). But for the logistic link function, good approximations exist when the distribution over the weights is Gaussian $p(\mathbf{w}|\mathcal{D}) = \mathcal{N}(\mathbf{w}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. One such approximation (MacKay, 1992) is constructed by scaling the input of the *probit* function[2] Φ by a constant $\lambda = \sqrt{\pi/8}$. Using this approximation and the Gaussian assumption, if we let $a := \mathbf{w}^T\mathbf{x}$, we get

$$p(y=1|\mathbf{x},\mathcal{D}) \approx \int \Phi(\sqrt{\pi/8}\, a)\, \mathcal{N}(a|\boldsymbol{\mu}^T\mathbf{x}, \mathbf{x}^T\boldsymbol{\Sigma}\mathbf{x})\, da \;= \Phi\left(\frac{\boldsymbol{\mu}^T\mathbf{x}}{\sqrt{8/\pi + \mathbf{x}^T\boldsymbol{\Sigma}\mathbf{x}}}\right) \approx \sigma\left(z(\mathbf{x})\right)\,, \tag{3}$$

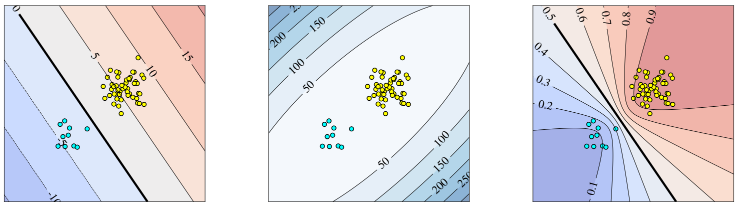
where the last step uses the approximation $\Phi(\sqrt{\pi/8}\, x) \approx \sigma(x)$ a second time, with

$$z(\mathbf{x}) := \frac{\boldsymbol{\mu}^T\mathbf{x}}{\sqrt{1 + \pi/8\, \mathbf{x}^T\boldsymbol{\Sigma}\mathbf{x}}}\,. \tag{4}$$

In the case of $\boldsymbol{\mu} = \mathbf{w}_{\text{MAP}}$, eq. (3) can be seen as the "softened" version of the MAP prediction of the classifier, using the covariance of the Gaussian. The principal aspect of interest of in this paper will be not so much any philosophical point about Bayesian inference, but that the approximate probabilistic Gaussian formalism as outlined in eqs. (1) and (3) introduces the second set of parameters in the form of $\boldsymbol{\Sigma}$. We will find that at least *asymptotic* overconfidence problems can be fixed by setting $\boldsymbol{\Sigma}$ to virtually any sensible value, regardless of whether they are motivated in a Bayesian fashion or not.

As a first notable property of this approximation, we show below that, in contrast to some other methods for uncertainty quantification (e.g. Monte Carlo dropout (Gal & Ghahramani, 2016)) it preserves the decision boundary induced by the MAP estimate. Moreover, this property still holds even if we use *any* feature map ϕ and define the linear classifier on the image of this map instead. The implication is important in practice, as this gives a guarantee that if we apply this approximation to the last layer of *any* MAP pre-trained neural networks, then the classification accuracy of the marginalized prediction is exactly the same as the MAP classification accuracy.

[2]The probit function Φ is another sigmoid, the distribution function (CDF) of the standard Gaussian.



(a) $\boldsymbol{\mu}^T\mathbf{x}$ (b) $\mathbf{x}^T\boldsymbol{\Sigma}\mathbf{x}$ (c) Confidence

Figure 2: An illustration of Proposition 2.3 for a linear classifier defined on $\mathbb{R}^2$. The confidence of the marginalized prediction of a linear classifier is the highest in the direction of the lowest curvature, as described by $\boldsymbol{\Sigma}$.

Proposition 2.2. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a binary linear classifier defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T\phi(\mathbf{x})$ where $\phi : \mathbb{R}^n \to \mathbb{R}^d$ is any feature map and let $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\mathbf{w}_{\text{MAP}}, \boldsymbol{\Sigma})$ be the distribution over $\mathbf{w}$. Then for any $\mathbf{x} \in \mathbb{R}^n$, we have $\sigma(z \circ \phi(\mathbf{x})) = 0.5$ if and only if $\sigma(\mathbf{w}_{\text{MAP}}^T\phi(\mathbf{x})) = 0.5$.*

Proof. See Proposition A.1 in Appendix A. □

Particularly in the Bayesian setting, the Gaussian approximate posterior required in the previous approximation can be obtained via various methods, such as a Laplace approximation. Let $p(\mathbf{w}|\mathcal{D}) \propto p(\mathbf{w}) \prod_{\mathbf{x},t\in\mathcal{D}} p(y = t|\mathbf{x}, \mathbf{w})$ be the posterior of a binary linear classifier.[3] Then we can obtain a Gaussian approximation $p(\mathbf{w}|\mathcal{D}) \approx \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ of the posterior by setting $\boldsymbol{\mu} = \mathbf{w}_{\text{MAP}}$ and $\boldsymbol{\Sigma} = (-\nabla^2|_{\mathbf{w}_{\text{MAP}}} \log p(\mathbf{w}|\mathcal{D}))^{-1}$, the inverse Hessian of the negative log-posterior. In our binary classification case, $p(y|\mathbf{x}, \mathbf{w})$ is assumed to be Bernoulli$(\sigma(\mathbf{w}^T\mathbf{x}))$ while $p(\mathbf{w})$ is assumed to be $\mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma_0^2\mathbf{I})$, leading to the standard ℓ^2-regularized binary cross-entropy loss.

2.3 Main results

As our central theoretical contribution, we show that, far away from the training points, $z(\mathbf{x})$ goes to a quantity that only depends on the mean and covariance of the Gaussian over the weights. This result implies that we can make $p(y = 1|\mathbf{x}, \mathcal{D})$ closer to one-half far away from the training points if we can make $z(\mathbf{x})$ closer to zero by controlling the Gaussian. Proposition 2.3 below shows this in the case of linear classifiers (also cf. Figure 2), while Theorem 2.4 shows that the analysis actually also holds in the case of ReLU networks.

Proposition 2.3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a binary linear classifier defined by $f(\mathbf{x}) := \mathbf{w}^T\mathbf{x}$ and $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the distribution over $\mathbf{w}$. Then for any $\mathbf{x} \in \mathbb{R}^n$,*

$$|z(\mathbf{x})| \leq \frac{|\boldsymbol{\mu}^T\mathbf{x}|}{\sqrt{1 + \pi/8\, \lambda_{\text{min}}(\boldsymbol{\Sigma})\|\mathbf{x}\|^2}} \,. \tag{5}$$

Furthermore, if $\mathbf{x} \in \mathbb{R}^n$ then as $\delta > 0$ goes to infinity

$$\lim_{\delta\to\infty} |z(\delta\mathbf{x})| \leq \frac{\|\boldsymbol{\mu}\|}{\sqrt{\pi/8\, \lambda_{\text{min}}(\boldsymbol{\Sigma})}} \,. \tag{6}$$

Proof. See Proposition A.3 in Appendix A. □

Let $\phi : \mathbb{R}^n \to \mathbb{R}^d$ be a ReLU network. Recall from the definition, ϕ is a piecewise affine function. Thus, we can write the input space as $\mathbb{R}^n = \cup_{r=1}^R Q_r$ and for every Q_r, the restriction $\phi|_{Q_r} : Q_r \to \mathbb{R}^d$ is an affine function $\phi|_{Q_r}(\mathbf{x}) := \mathbf{V}_r\mathbf{x} + \mathbf{a}_r$ for some $\mathbf{V}_r \in \mathbb{R}^{d\times n}$ and $\mathbf{a}_r \in \mathbb{R}^d$. Note that if $i, j \in \{1, \ldots, M\}$ with $i \neq j$ then in general $\mathbf{V}_i \neq \mathbf{V}_j$ and $\mathbf{a}_i \neq \mathbf{a}_j$. Using this definition, we can

[3]We use a linear model here for simplicity and not as a requirement.

also show a similar result to Proposition 2.3, in the case when $\mathbf{x}$ is replaced by any feature vector in the image of ϕ.

Theorem 2.4. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a binary linear classifier defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$ where $\phi : \mathbb{R}^n \to \mathbb{R}^d$ is a ReLU network and let $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the distribution over $\mathbf{w}$. Then for any $\mathbf{x} \in \mathbb{R}^n$,*

$$|z \circ \phi(\mathbf{x})| \leq \frac{|\boldsymbol{\mu}^T(\mathbf{V}\mathbf{x} + \mathbf{a})|}{\sqrt{1 + \pi/8\, \lambda_{\text{min}}(\boldsymbol{\Sigma}) \|\mathbf{V}\mathbf{x} + \mathbf{a}\|^2}}\,, \tag{7}$$

where $\mathbf{V} \in \mathbb{R}^{d \times n}$ and $\mathbf{a} \in \mathbb{R}^d$ are some matrix and vector that depend on $\mathbf{x}$. Furthermore, as $\delta > 0$ goes to infinity

$$\lim_{\delta \to \infty} |z \circ \phi(\delta \mathbf{x})| \leq \frac{\|\boldsymbol{\mu}\|}{\sqrt{\pi/8\, \lambda_{\text{min}}(\boldsymbol{\Sigma})}}\,. \tag{8}$$

Proof. See Theorem A.5 in Appendix A. □

Given a target upper bound on the logit and confidence values of a ReLU network, we can concretely pick the covariance $\boldsymbol{\Sigma}$ that respects the asymptotic bound of Theorem 2.4.

Corollary 2.5 ($\boldsymbol{\Sigma}$ from a desired upper confidence bound on ReLU networks)**.** *Let $f \circ \phi$, with $\phi : \mathbb{R}^n \to \mathbb{R}^d$ and $f : \mathbb{R}^d \to \mathbb{R}$, be a ReLU network defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$ and $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the distribution over $\mathbf{w}$ where the mean $\boldsymbol{\mu}$ is fixed and $\boldsymbol{\Sigma}$ is any SPD matrix. Then:*

(i) For any $\epsilon > 0$ there exists $\boldsymbol{\Sigma}$ such that for any $\mathbf{x} \in \mathbb{R}^n$ far away from the training data, we have that $|z \circ \phi(\mathbf{x})| \leq \epsilon$.

(ii) For any $0.5 < p < 1$ there exists $\boldsymbol{\Sigma}$ such that for any $\mathbf{x} \in \mathbb{R}^n$ far away from the training data, we have that $\sigma(|z \circ \phi(\mathbf{x})|) \leq p$.

Proof. See Corollary A.6 in Appendix A. □

2.4 Being Bayesian (not just probabilistic) with Laplace approximations

Proposition 2.3 and Theorem 2.4 imply that the confidence of a binary linear classifier with ReLU features can be bound closer to one-half by increasing the minimum eigenvalue of the posterior covariance. In this section, we will move towards the Bayesian setting (i.e. using an explicit prior and likelihood, not just an imposed probability measure on the weights). Specifically, we will present a way to control the posterior through the prior in a Laplace approximation. Concretely, the following proposition and its immediate corollary point out that the eigenvalues of the posterior covariance can be increased (bringing $|z(\delta\mathbf{x})|$ closer to zero) by increasing the prior variance.

Proposition 2.6. *Let $f \circ \phi$, with $\phi : \mathbb{R}^n \to \mathbb{R}^d$ and $f : \mathbb{R}^d \to \mathbb{R}$, be a ReLU network defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$, modeling a Bernoulli distribution with $p(y = 1|\mathbf{x}, \mathbf{w}) = \sigma(f \circ \phi(\mathbf{x}))$. Let $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the posterior over $\mathbf{w}$, obtained via a Laplace approximation with prior $\mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma_0^2 \mathbf{I})$. Suppose $\mathbf{H}$ is the Hessian w.r.t. $\mathbf{w}$ at $\boldsymbol{\mu}$ of the negative log-likelihood of the model. Then*

(i) $\mathbf{H} = \sum_{\mathbf{x} \in \mathcal{D}} \beta(\mathbf{x})\, \phi(\mathbf{x})\phi(\mathbf{x})^T$ where $\beta(\mathbf{x}) := \sigma(f \circ \phi(\mathbf{x}))(1 - \sigma(f \circ \phi(\mathbf{x})))$.

(ii) For each $i = 1, \ldots, d$, the ith eigenvalue $\lambda_i(\boldsymbol{\Sigma})$ of $\boldsymbol{\Sigma}$ is a non-decreasing function of σ_0^2 with limits $1/\lambda_i(\mathbf{H})$ as $\sigma_0^2 \to \infty$ and 0 as $\sigma_0^2 \to 0$.

Proof. See Proposition A.7 in Appendix A. □

We get an immediate corollary from Proposition 2.6 that relates its results to Theorem 2.4.

Corollary 2.7. *Let $f \circ \phi$, with $\phi : \mathbb{R}^n \to \mathbb{R}^d$ and $f : \mathbb{R}^d \to \mathbb{R}$, be a ReLU network defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$, modeling $p(y = 1|\mathbf{x}, \mathbf{w})$ with $\sigma(f \circ \phi(\mathbf{x}))$. Let $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the posterior over $\mathbf{w}$, obtained via a Laplace approximation with prior $\mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma_0^2 \mathbf{I})$. Then $z \circ \phi(\mathbf{x})$ is a non-increasing function of σ_0^2 with limits*

$$\lim_{\sigma_0^2 \to \infty} |z \circ \phi(\mathbf{x})| \leq \frac{|\boldsymbol{\mu}^T \phi(\mathbf{x})|}{\sqrt{1 + \pi/(8\, \lambda_{\max}(\mathbf{H})) \|\phi(\mathbf{x})\|^2}} \,,$$
$$\lim_{\sigma_0^2 \to 0} z \circ \phi(\mathbf{x}) = \boldsymbol{\mu}^T \phi(\mathbf{x}) \,,$$

where $\mathbf{H}$ is as defined in (i) of Proposition 2.6.

Proof. See Corollary A.8 in Appendix A. □

Lastly, the following corollary formalizes the intuition that the marginalized prediction with the inverse empirical features covariance $\mathbf{C}^{-1}$ as $\boldsymbol{\Sigma}$ will naturally have high uncertainty far away from the training data. Furthermore, this property can also be observed for Laplace approximation if the spectral properties of the Hessian ((i) of Proposition 2.6) are not too different to those of $\mathbf{C}$.

Corollary 2.8. *Let $f \circ \phi$, with $\phi : \mathbb{R}^n \to \mathbb{R}^d$ and $f : \mathbb{R}^d \to \mathbb{R}$, be a ReLU network defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$, modeling $p(y = 1|\mathbf{x}, \mathbf{w})$ with $\sigma(f \circ \phi(\mathbf{x}))$. Let $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the distribution over $\mathbf{w}$. If either*

(i) $\boldsymbol{\Sigma} = \mathbf{C}^{-1} := \left(\sum_{\mathbf{x} \in \mathcal{D}} \phi(\mathbf{x}) \phi(\mathbf{x})^T\right)^{-1}$*, or*

(ii) $\boldsymbol{\Sigma}$ *is obtained via a Laplace approximation w.r.t. a prior $\mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma_0^2 \mathbf{I})$ with $\sigma_0^2 \to \infty$ and suppose $\mathbf{H}$ defined in (i) of Proposition 2.6 is invertible and the ordering of its eigenvalues is the same as that of $\mathbf{C}$, while the eigenvectors are the same as those of $\mathbf{C}$,*

then on any level set of $\boldsymbol{\mu}^T \phi(\mathbf{x})$, the confidence decreases faster in the direction where the training data are sparser in the feature space $\mathbb{R}^d$.

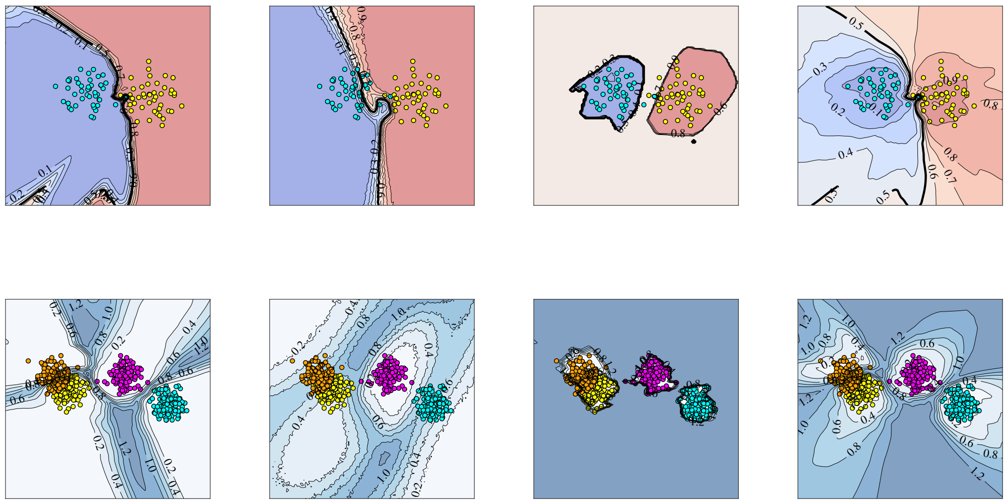
Proof. See Corollary A.9 in Appendix A. □

Similar statements for multi-class classifiers are not as straight-forward due to the lack of a good closed-form approximation of the integral of softmax under a Gaussian measure. However, as can be seen in Appendix C, at least the application of the above analysis can easily be generalized to the multi-class case. In fact, in the experiments (Section 4), we mainly use multi-class classifiers and show empirically that they are effective in mitigating issues that arise from the overconfidence problem.

3 RELATED WORK

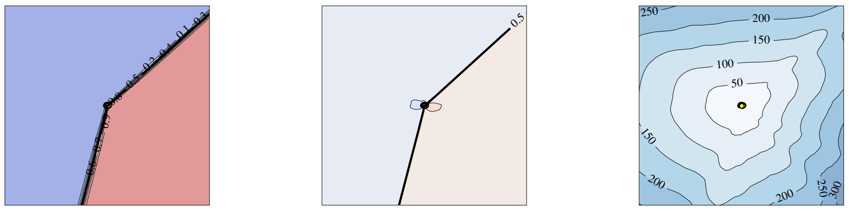
The overconfidence problem of deep neural networks, and thus ReLU networks, has long been known in the deep learning community (Nguyen et al., 2015). However, only recently this issue was demonstrated formally (Hein et al., 2019). Many methods have been proposed to combat or at least detect this issue. *Post-hoc* heuristics based on temperature or Platt scaling (Guo et al., 2017; Liang et al., 2018) are unable to detect inputs with arbitrarily high confidence far away from the training data (Hein et al., 2019). Hein et al. (2019) proposed enhanced training objectives based on robust optimization to mitigate this issue.

Bayesian methods have long been thought to mitigate the overconfidence problem on any neural network (MacKay, 1992). Empirical evidence supporting this intuition has also been presented (Liu et al., 2019; Wu et al., 2019, etc.). Our results complement these with a theoretical justification for the ReLU-logistic case. But while our work is theoretical in nature, we believe its application has practical value since it shows that a full Bayesian (expensive) treatment is not necessary if one is only worried about overconfidence. Indeed, fully Bayesian neural networks are often intractable and crude approximations have to be used, resulting in undesirable results (Foong et al., 2019).



(a) MAP (b) MC-Dropout (c) ACET (d) Last-layer Laplace

Figure 3: Binary (top) and multi-class (bottom) toy classification. The color represents either the confidence (top) or entropy (bottom) of the prediction, with darker shade implies higher value.



(a) MAP confidence (b) Laplace confidence (c) Denominator of z

Figure 4: The zoomed-out version of the MAP's and Laplace's confidence from Figure 1d, along with denominator of z.

4 Experiments

In this section we validate our theoretical results by applying a Laplace approximation *only* to the last layer of various widely used ReLU networks and call this method *last-layer Laplace approximation* (LLLA). We refer the reader to Appendix C for details. Note that since LLLA is the simplest Laplace approximation that we can apply to deep networks, our results should also hold for more general Laplace methods, e.g. Kronecker-factored Laplace (KFLA) (Ritter et al., 2018), where not only the linear classifier's posterior but also the posterior of the feature map is approximated. Note however, these fully-Bayesian methods are significantly more expensive and require a significant amount of implementation effort.

We will present our empirical results on (i) a 2D toy classification task and (ii) out-of-distribution (OOD) data detection experiments. For the OOD experiment, we find the optimal prior variance σ_0^2 via a heuristic that follows directly from Corollary 2.7. Concretely, we pick the largest positive integer that makes the drop on the mean maximum confidence (MMC) of the in-distribution dataset to be within around 0.03 of the MAP's MMC. Thus, we only set this once without seeing any of the OOD datasets.

Table 1: OOD detection results. For MMC, the lower the better in the OOD case, but the higher the better for the in-distribution case; for AUROC, the higher the better.

	MAP		**CEDA**		**ACET**		**LLLA**	
Train - Test	MMC	AUROC	MMC	AUROC	MMC	AUROC	MMC	AUROC
MNIST - MNIST	**0.993**	-	**0.993**	-	**0.993**	-	0.962	-
MNIST - notMNIST	0.715	0.976	0.676	0.979	0.649	**0.983**	**0.511**	0.978
MNIST - EMNIST	0.803	0.944	0.802	0.943	0.790	**0.948**	**0.608**	0.947
MNIST - FMNIST	0.570	0.993	0.458	**0.995**	0.471	**0.995**	**0.383**	**0.995**
CIFAR-10 - CIFAR-10	**0.910**	-	0.906	-	0.883	-	0.880	-
CIFAR-10 - SVHN	0.716	0.813	0.692	**0.824**	0.678	0.797	**0.606**	0.823
CIFAR-10 - LSUN-CR	0.732	0.810	0.699	0.824	0.649	**0.829**	**0.623**	0.818

4.1 Toy dataset

The dataset is constructed by sampling the input points from k Gaussians. The corresponding targets indicate from which Gaussian the point was sampled. We use a 5-layer ReLU network with 100 hidden units at each layer as the feature map ϕ. The classifier, along with this feature map is trained jointly. We show the results for the binary and multi-class ($k = 4$) case in Figure 3. As we can see, the MAP predictions have high confidence (low entropy) everywhere except at the region close to the decision boundary. The widely used MC-dropout does not remedy this issue. While ACET remedies the overconfidence issue, it is expensive and in general does not preserve the decision boundary. In contrast, LLLA yields better calibrated predictions: high confidence close to the training points and high uncertainty otherwise, while maintaining the MAP's decision boundary. We furthermore show the zoomed-out version of LLLA prediction we have presented in Figure 1d, along with the contour of the denominator of z (eq. (4)) in Figure 4. We see that the covariance acts as a "moderator" for the MAP predictions: As a test point moves away from the training data, the denominator of z becomes larger and the marginalized prediction goes to a constant close to one-half.

4.2 Out-of-distribution experiments

Following the experiment of Hein et al. (2019), we compare LLLA with CEDA and ACET (Hein et al., 2019), two recently-proposed non-Bayesian methods for mitigating the overconfidence problem of ReLU networks. We use a variant of LeNet for MNIST experiments and a pre-activation ResNet-20 architecture (He et al., 2016b) for the CIFAR-10 experiment. The CEDA's and ACET's out-distribution of choice is the uniform distribution. For ACET specifically, we solve the inner optimization via a projected gradient descent with 10 iterations, radius 0.3, and step size 0.0075. More detail about the training setup is presented in Appendix D. We use standard metrics for measuring robustness to OOD data (Hendrycks & Gimpel, 2017), namely: (i) mean maximum confidence (MMC) and (ii) area under the ROC curve (AUROC). As presented in Table 1, LLLA yields competitive performance compared to both CEDA and ACET.

5 Conclusion

We have shown that even an extremely approximate and virtually non-Bayesian probabilistic Gaussian treatment mitigates the most extreme aspects of overconfidence in ReLU networks. Our analytical results bound the confidence of the Bayesian prediction of linear classifiers and ReLU networks far away from the training data away from one. This motivates a spectrum of approximations, from ad-hoc isotropic to "full Bayesian" Laplace approximations. In the Laplace approximation case, the bound asymptotically converges to a constant whose value can be controlled via the prior. We validated our results experimentally by constructing a simple Laplace method that can still capture the properties we have shown, specifically by only approximating the last-layer's posterior distribution. In contrast to other approximations, this method is cheap and simple to implement, yet already yields competitive performance compared to the more expensive, recently proposed non-Bayesian method for combating the overconfidence problem. While more elaborate Laplace approximations can improve fidelity the further, our results provide virtually any ReLU network with a simple and computationally lightweight way to mitigate overconfidence.

REFERENCES

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

Raman Arora, Amitabh Basu, Poorya Mianjy, and Anirbit Mukherjee. Understanding deep neural networks with rectified linear units. In *ICLR*, 2018.

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. In *ICML*, 2015.

Andrew YK Foong, Yingzhen Li, José Miguel Hernández-Lobato, and Richard E Turner. 'in-between'uncertainty in bayesian neural networks. *arXiv preprint arXiv:1906.11537*, 2019.

Yarin Gal. Uncertainty in deep learning. *University of Cambridge*, 2016.

Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *ICML*, 2017.

Arjun K Gupta and Daya K Nagar. *Matrix variate distributions*. Chapman and Hall/CRC, 1999.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *ECCV*, 2016b.

Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *CVPR*, 2019.

Dan Hendrycks and Kevin Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*, 2017.

Shiyu Liang, Yixuan Li, and R. Srikant. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.

Xuanqing Liu, Yao Li, Chongruo Wu, and Cho-Jui Hsieh. Adv-BNN: Improved adversarial defense through robust bayesian neural network. In *ICLR*, 2019.

Christos Louizos and Max Welling. Multiplicative normalizing flows for variational Bayesian neural networks. In *ICML*, 2017.

David JC MacKay. The evidence framework applied to classification networks. *Neural computation*, 1992.

Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *CVPR*, 2015.

Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *ICLR*, 2018.

Anqi Wu, Sebastian Nowozin, Edward Meeds, Richard E. Turner, Jose Miguel Hernandez-Lobato, and Alexander L. Gaunt. Deterministic variational inference for robust bayesian neural networks. In *ICLR*, 2019.

APPENDIX A PROOFS

Proposition A.1. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a binary linear classifier defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$ where $\phi : \mathbb{R}^n \to \mathbb{R}^d$ is any feature map and let $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\mathbf{w}_{\text{MAP}}, \mathbf{\Sigma})$ be the distribution over $\mathbf{w}$. Then for any $\mathbf{x} \in \mathbb{R}^n$, we have $\sigma(z \circ \phi(\mathbf{x})) = 0.5$ if and only if $\sigma(\mathbf{w}_{\text{MAP}}^T \phi(\mathbf{x})) = 0.5$.*

Proof. Denote $\mu_f := \mathbf{w}_{\text{MAP}}^T \phi(\mathbf{x})$ and $\sigma_f^2 := \phi(\mathbf{x})^T \mathbf{\Sigma} \phi(\mathbf{x})$ and let $\mathbf{x} \in \mathbb{R}^n$ be arbitrary. For the forward direction, suppose that $\sigma(\mu_f) = 0.5$. This implies $\mu_f = 0$, and we have $\sigma(0/(1 + \pi/8\, \sigma_f^2)^{1/2}) = \sigma(0) = 0.5$. For the reverse direction, suppose that $\sigma(\mu_f/(1 + \pi/8\, \sigma_f^2)^{1/2}) = 0.5$. This implies $\mu_f/(1 + \pi/8\, \sigma_f^2)^{1/2} = 0$. Notice, the denominator of the l.h.s. is positive. Thus, it follows that μ_f must be 0, implying that $\sigma(\mu_f) = 0.5$. □

Lemma A.2. *Let $\mathbf{x} \in \mathbb{R}^n$ be a vector and $\mathbf{A} \in \mathbb{R}^{n \times n}$ be an SPD matrix. If $\lambda_{\text{min}}(\mathbf{A})$ is the minimum eigenvalue of $\mathbf{A}$, then $\mathbf{x}^T \mathbf{A} \mathbf{x} \geq \lambda_{\text{min}} \|\mathbf{x}\|^2$.*

Proof. Since $\mathbf{A}$ is SPD, it admits an eigendecomposition $\mathbf{A} = \mathbf{Q} \mathbf{\Lambda} \mathbf{Q}^T$ and $\mathbf{\Lambda} = \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Lambda}^{\frac{1}{2}}$ makes sense. Therefore, by keeping in mind that $\mathbf{Q}^T \mathbf{x}$ is a vector in $\mathbb{R}^n$, we have

$$
\begin{aligned}
\mathbf{x}^T \mathbf{A} \mathbf{x} = \mathbf{x}^T \mathbf{Q} \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{\Lambda}^{\frac{1}{2}} \mathbf{Q}^T \mathbf{x} = \|\mathbf{\Lambda}^{\frac{1}{2}} \mathbf{Q}^T \mathbf{x}\|^2 &= \sum_{i=1}^{n} \lambda_i (\mathbf{Q}^T \mathbf{x})_i^2 \\
&\geq \lambda_{\text{min}}(\mathbf{A}) \sum_{i=1}^{n} (\mathbf{Q}^T \mathbf{x})_i^2 = \lambda_{\text{min}}(\mathbf{A}) \|\mathbf{Q}^T \mathbf{x}\|^2 \\
&= \lambda_{\text{min}}(\mathbf{A}) \|\mathbf{x}\|^2 \,,
\end{aligned}
$$

where the last equality is obtained as $\|\mathbf{Q}^T \mathbf{x}\|^2 = \mathbf{x}^T \mathbf{Q}^T \mathbf{Q} \mathbf{x}$ and noting that $\mathbf{Q}$ is an orthogonal matrix. □

Proposition A.3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a binary linear classifier defined by $f(\mathbf{x}) := \mathbf{w}^T \mathbf{x}$ and $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \mathbf{\Sigma})$ be the distribution over $\mathbf{w}$. Then for any $\mathbf{x} \in \mathbb{R}^n$,*

$$
|z(\mathbf{x})| \leq \frac{|\boldsymbol{\mu}^T \mathbf{x}|}{\sqrt{1 + \pi/8\, \lambda_{\text{min}}(\mathbf{\Sigma}) \|\mathbf{x}\|^2}} \,. \tag{9}
$$

Furthermore, if $\mathbf{x} \in \mathbb{R}^n$ then as $\delta > 0$ goes to infinity

$$
\lim_{\delta \to \infty} |z(\delta \mathbf{x})| \leq \frac{\|\boldsymbol{\mu}\|}{\sqrt{\pi/8\, \lambda_{\text{min}}(\mathbf{\Sigma})}} \,. \tag{10}
$$

Proof. The first result follows directly from Lemma A.2 and by noting that the denominator of eq. (4) is positive since $\mathbf{\Sigma}$ is symmetric positive-definite (SPD) by definition. For the second result, let $\mathbf{x} \in \mathbb{R}^n$ be arbitrary. By computation and again since the denominator of eq. (4) is positive, we have

$$
|z(\delta \mathbf{x})| := \frac{|\boldsymbol{\mu}^T (\delta \mathbf{x})|}{\sqrt{1 + \pi/8\, (\delta \mathbf{x})^T \mathbf{\Sigma} (\delta \mathbf{x})}} = \frac{|\boldsymbol{\mu}^T \mathbf{x}|}{\sqrt{1/\delta^2 + \pi/8\, \mathbf{x}^T \mathbf{\Sigma} \mathbf{x}}} \,.
$$

We would like to inspect the asymptotic behavior of $z(\delta \mathbf{x})$ with respect to δ. First, for the sake of completeness, we can compute that $\lim_{\delta \to 0} |z(\delta \mathbf{x})| = 0$. This reflects the case when $\delta \mathbf{x}$ goes to the decision boundary. Now, for the case when $\delta \to \infty$, we can see that

$$
\lim_{\delta \to \infty} |z(\delta)| = \frac{|\boldsymbol{\mu}^T \mathbf{x}|}{\sqrt{\pi/8\, \mathbf{x}^T \mathbf{\Sigma} \mathbf{x}}} \,,
$$

since $1/\delta^2 \to 0$ as $\delta \to \infty$. Therefore, using Lemma A.2 and Cauchy-Schwarz inequality, we have

$$
\lim_{\delta \to \infty} |z(\delta \mathbf{x})| \leq \frac{|\boldsymbol{\mu}^T \mathbf{x}|}{\sqrt{\pi/8\, \lambda_{\text{min}}(\mathbf{\Sigma}) \|\mathbf{x}\|^2}} \leq \frac{\|\boldsymbol{\mu}\| \|\mathbf{x}\|}{\sqrt{\pi/8\, \lambda_{\text{min}}(\mathbf{\Sigma})} \|\mathbf{x}\|} = \frac{\|\boldsymbol{\mu}\|}{\sqrt{\pi/8\, \lambda_{\text{min}}(\mathbf{\Sigma})}} \,,
$$

thus the proof is complete. □

Lemma A.4 (Hein et al. (2019))**.** *Let $\{Q_i\}_{l=1}^R$ be the set of linear regions associated to the ReLU network $\phi : \mathbb{R}^n \to \mathbb{R}^n$. For any $\mathbf{x} \in \mathbb{R}^n$ there exists $\alpha \in \mathbb{R}$ with $\alpha > 0$ and $t \in \{1, \dots, R\}$ such that $\delta\mathbf{x} \in Q_t$ for all $\beta \geq \alpha$. Furthermore, the restriction of ϕ to Q_t can be written as an affine function.* □

Theorem A.5. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a binary linear classifier defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$ where $\phi : \mathbb{R}^n \to \mathbb{R}^d$ is a ReLU network and let $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the distribution over $\mathbf{w}$. Then for any $\mathbf{x} \in \mathbb{R}^n$,*

$$|z \circ \phi(\mathbf{x})| \leq \frac{|\boldsymbol{\mu}^T(\mathbf{V}\mathbf{x} + \mathbf{a})|}{\sqrt{1 + \pi/8\, \lambda_{\text{min}}(\boldsymbol{\Sigma}) \|\mathbf{V}\mathbf{x} + \mathbf{a}\|^2}} \,, \tag{11}$$

where $\mathbf{V} \in \mathbb{R}^{d \times n}$ and $\mathbf{a} \in \mathbb{R}^d$ are some matrix and vector that depend on $\mathbf{x}$. Furthermore, as $\delta > 0$ goes to infinity

$$\lim_{\delta \to \infty} |z \circ \phi(\delta\mathbf{x})| \leq \frac{\|\boldsymbol{\mu}\|}{\sqrt{\pi/8\, \lambda_{\text{min}}(\boldsymbol{\Sigma})}} \,. \tag{12}$$

Proof. Let $\mathbf{x} \in \mathbb{R}^n$ be arbitrary. By definition of ReLU network, there exists a linear region $Q \subset \mathbb{R}^n$ along with $\mathbf{V} \in \mathbb{R}^{d \times n}$ and $\mathbf{b} \in \mathbb{R}^d$ such that $\mathbf{x} \in Q$ and $\phi|_Q(\mathbf{x}) := \mathbf{V}\mathbf{x} + \mathbf{a}$. Applying eq. (4) to $\phi|_Q(\mathbf{x})$ and following the proof of Proposition 2.3 yield

$$|z \circ \phi|_Q(\mathbf{x})| = \frac{|\boldsymbol{\mu}^T(\mathbf{V}\mathbf{x} + \mathbf{a})|}{\sqrt{1 + \pi/8\, (\mathbf{V}\mathbf{x} + \mathbf{a})^T \boldsymbol{\Sigma} (\mathbf{V}\mathbf{x} + \mathbf{a})}} \leq \frac{|\boldsymbol{\mu}^T(\mathbf{V}\mathbf{x} + \mathbf{a})|}{\sqrt{1 + \pi/8\, \lambda_{\text{min}}(\boldsymbol{\Sigma}) \|\mathbf{V}\mathbf{x} + \mathbf{a}\|^2}} \,, \tag{13}$$

thus the first result is obtained.

For the second result, by Lemma 3.1 of Hein et al. (2019) (also presented in Lemma A.4) there exists $\alpha > 0$ and a linear region R, along with $\mathbf{U} \in \mathbb{R}^{d \times n}$ and $\mathbf{c} \in \mathbb{R}^d$, such that for any $\delta \geq \alpha$, we have that $\delta\mathbf{x} \in R$ and the restriction $\phi|_R$ can be written as $\mathbf{U}\mathbf{x} + \mathbf{c}$. Therefore, for any such δ,

$$\begin{aligned} |z \circ \phi|_R(\delta\mathbf{x})| &= \frac{|\boldsymbol{\mu}^T(\delta\mathbf{U}\mathbf{x} + \mathbf{c})|}{\sqrt{1 + \pi/8\, (\delta\mathbf{U}\mathbf{x} + \mathbf{b})^T \boldsymbol{\Sigma} (\delta\mathbf{U}\mathbf{x} + \mathbf{c})}} \\ &= \frac{|\boldsymbol{\mu}^T(\mathbf{U}\mathbf{x} + \frac{1}{\delta}\mathbf{c})|}{\sqrt{\frac{1}{\delta^2} + \pi/8\, (\mathbf{U}\mathbf{x} + \frac{1}{\delta}\mathbf{c})^T \boldsymbol{\Sigma} (\mathbf{U}\mathbf{x} + \frac{1}{\delta}\mathbf{c})}} \end{aligned}$$

Now, notice that as $\delta \to \infty$, $1/\delta^2$ and $1/\delta$ goes to zero. So, in the limit, we have that

$$\lim_{\delta \to \infty} |z \circ \phi|_R(\delta\mathbf{x})| = \frac{|\boldsymbol{\mu}^T(\mathbf{U}\mathbf{x})|}{\sqrt{\pi/8\, (\mathbf{U}\mathbf{x})^T \boldsymbol{\Sigma} (\mathbf{U}\mathbf{x})}} \,.$$

Again, following the proof of Proposition 2.3 (i.e. using Cauchy-Schwarz and Lemma A.2), we can upper-bound this limit with

$$\lim_{\delta \to \infty} |z \circ \phi|_R(\delta\mathbf{x})| \leq \frac{\|\boldsymbol{\mu}\| \|\mathbf{U}\mathbf{x}\|}{\sqrt{\pi/8\, \lambda_{\text{min}}(\boldsymbol{\Sigma}) \|\mathbf{U}\mathbf{x}\|^2}} = \frac{\|\boldsymbol{\mu}\|}{\sqrt{\pi/8\, \lambda_{\text{min}}(\boldsymbol{\Sigma})}} \,,$$

which concludes the proof. □

Corollary A.6 ($\lambda_{\text{min}}(\boldsymbol{\Sigma})$ from a desired upper confidence bound on ReLU networks)**.** *Let $f \circ \phi$, with $\phi : \mathbb{R}^n \to \mathbb{R}^d$ and $f : \mathbb{R}^d \to \mathbb{R}$, be a ReLU network defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$ and $\mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the distribution over $\mathbf{w}$ where the mean $\boldsymbol{\mu}$ is fixed and $\boldsymbol{\Sigma}$ is any SPD matrix. Then:*

(i) For any $\epsilon > 0$ there exists $\boldsymbol{\Sigma}$ such that for any $\mathbf{x} \in \mathbb{R}^n$ far away from the training data, we have that $|z \circ \phi(\mathbf{x})| \leq \epsilon$.

(ii) For any $0.5 < p < 1$ there exists $\boldsymbol{\Sigma}$ such that for any $\mathbf{x} \in \mathbb{R}^n$ far away from the training data, we have that $\sigma(|z \circ \phi(\mathbf{x})|) \leq p$.

Proof. We begin with (i). Let $\epsilon > 0$ and $\delta = \frac{8}{\pi}\left(\frac{\|\boldsymbol{\mu}\|}{\epsilon}\right)^2$. Pick any $\boldsymbol{\Sigma}$ SPD with $\lambda_{\text{min}}(\boldsymbol{\Sigma}) = \delta$. Then, by eq. (12) of Theorem 2.4 and our choice of $\lambda_{\text{min}}(\boldsymbol{\Sigma})$, for any $\mathbf{z} \in \mathbb{R}^n$, asymptotically we have that

$$\lim_{\delta \to \infty} |z \circ \phi(\delta \mathbf{z})| \leq \frac{\|\boldsymbol{\mu}\|}{\sqrt{\pi/8\, \lambda_{\text{min}}(\boldsymbol{\Sigma})}} = \frac{\|\boldsymbol{\mu}\|}{\sqrt{\pi/8\, \delta}} = \epsilon \,,$$

which is the desired result.

For (ii), let $0.5 < p < 1$ be arbitrary. Observe that the inverse logistic function is given by $\sigma^{-1}(x) := \log x/(1-x)$ for $0 < x < 1$ and it is positive for $0.5 < x < 1$. Therefore by setting ϵ in (i) with $\sigma^{-1}(p)$, we can pick $\boldsymbol{\Sigma}$ SPD with $\lambda_{\text{min}}(\boldsymbol{\Sigma}) = \frac{8}{\pi}\left(\frac{\|\boldsymbol{\mu}\|}{\sigma^{-1}(p)}\right)^2$ and verify that for any $\mathbf{x} \in \mathbb{R}^n$ this gives $|z(\mathbf{x})| \leq \sigma^{-1}(p)$. Thus, for any $\mathbf{x} \in \mathbb{R}^n$ far away from the training data, since σ is monotonic, we have that

$$\sigma(|z(\mathbf{x})|) \leq \sigma(\sigma^{-1}(p)) = p \,,$$

and the proof is complete. □

Proposition A.7. *Let $f \circ \phi$, with $\phi : \mathbb{R}^n \to \mathbb{R}^d$ and $f : \mathbb{R}^d \to \mathbb{R}$, be a ReLU network defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$, modeling a Bernoulli distribution with $p(y = 1|\mathbf{x}, \mathbf{w}) = \sigma(f \circ \phi(\mathbf{x}))$. Let $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the posterior over $\mathbf{w}$, obtained via a Laplace approximation with prior $\mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma_0^2 \mathbf{I})$. Suppose $\mathbf{H}$ is the Hessian w.r.t. $\mathbf{w}$ at $\boldsymbol{\mu}$ of the negative log-likelihood of the model. Then*

(i) $\mathbf{H} = \sum_{\mathbf{x} \in \mathcal{D}} \beta(\mathbf{x})\, \phi(\mathbf{x})\phi(\mathbf{x})^T$ *where* $\beta(\mathbf{x}) := \sigma(f \circ \phi(\mathbf{x}))(1 - \sigma(f \circ \phi(\mathbf{x})))$.

(ii) For each $i = 1, \ldots, d$, the ith eigenvalue $\lambda_i(\boldsymbol{\Sigma})$ of $\boldsymbol{\Sigma}$ is a non-decreasing function of σ_0^2 with limits $1/\lambda_i(\mathbf{H})$ as $\sigma_0^2 \to \infty$ and 0 as $\sigma_0^2 \to 0$.

Proof. The negative log-likelihood of Bernoulli distribution is given by

$$-\log \prod_{\mathbf{x}, t \in \mathcal{D}} p(y|\mathbf{x}, \mathbf{w}) = -\sum_{\mathbf{x}, t \in \mathcal{D}} t \, \log \sigma(f \circ \phi(\mathbf{x})) + (1-t) \, \log(1 - \sigma(f \circ \phi(\mathbf{x}))) \,.$$

Now, observing that $\sigma'(x) = \sigma(x)(1 - \sigma(x))$ for all $x \in \mathbb{R}$, we can compute

$$\nabla^2_{\mathbf{w}} \log \sigma(f \circ \phi(\mathbf{x})) = \nabla^2_{\mathbf{w}} \log(1 - \sigma(f \circ \phi(\mathbf{x}))) = \sigma(f \circ \phi(\mathbf{x}))(1 - \sigma(f \circ \phi(\mathbf{x})))\, \phi(\mathbf{x})\phi(\mathbf{x})^T \,.$$

This implies $\nabla^2_{\mathbf{w}} \log p(y|\mathbf{x}, \mathbf{w}) = \sigma(f \circ \phi(\mathbf{x}))(1 - \sigma(f \circ \phi(\mathbf{x})))\, \phi(\mathbf{x})\phi(\mathbf{x})^T$, since $t \in \{0, 1\}$ by assumption. By considering all $\mathbf{x}, t \in \mathcal{D}$, we get (i).

For (ii), first we assume that all Hessians mentioned below are w.r.t. $\mathbf{w}$. We note that the assumption on the prior implies $-\log p(\mathbf{w}) = 1/2\, \mathbf{w}^T (1/\sigma_0^2 \mathbf{I}) \mathbf{w} + \text{const}$, which has Hessian $1/\sigma_0^2 \mathbf{I}$. Thus, the Hessian of the negative log posterior $-\log p(\mathbf{w}|\mathcal{D}) = -\log p(\mathbf{w}) - \log \prod_{\mathbf{x}, t \in \mathcal{D}} p(y|\mathbf{x}, \mathbf{w})$ is $1/\sigma_0^2 \mathbf{I} + \mathbf{H}$. This implies that the posterior covariance $\boldsymbol{\Sigma}$ of the Laplace approximation is given by

$$\boldsymbol{\Sigma} = \left(\frac{1}{\sigma_0^2}\mathbf{I} + \mathbf{H}\right)^{-1} \,. \tag{14}$$

Therefore, the ith eigenvalue of $\boldsymbol{\Sigma}$ for any $i = 1, \ldots, n$ is

$$\lambda_i(\boldsymbol{\Sigma}) = \frac{1}{1/\sigma_0^2 + \lambda_i(\mathbf{H})} = \frac{\sigma_0^2}{1 + \sigma_0^2 \lambda_i(\mathbf{H})} \,.$$

For all $i = 1, \ldots, n$, the derivative of $\lambda_i(\boldsymbol{\Sigma})$ w.r.t. σ_0^2 is $1/(1 + \sigma_0^2 \lambda_i(\mathbf{H}))^2$ which is non-negative. This tells us that $\lambda_i(\boldsymbol{\Sigma})$ is a non-decreasing function of σ_0^2. Furthermore, it is also clear that $\sigma_0^2/(1 + \sigma_0^2 \lambda_i(\mathbf{H}))$ goes to $1/\lambda_i(\mathbf{H})$ as σ_0^2 goes to infinity, while it goes to 0 as σ_0^2 goes to zero. □

Corollary A.8. *Let $f \circ \phi$, with $\phi : \mathbb{R}^n \to \mathbb{R}^d$ and $f : \mathbb{R}^d \to \mathbb{R}$, be a ReLU network defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$, modeling $p(y = 1|\mathbf{x}, \mathbf{w})$ with $\sigma(f \circ \phi(\mathbf{x}))$. Let $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the posterior over $\mathbf{w}$, obtained via a Laplace approximation with prior $\mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma_0^2 \mathbf{I})$. Then $z \circ \phi(\mathbf{x})$ is a non-increasing function of σ_0^2 with limits*

$$\lim_{\sigma_0^2 \to \infty} |z \circ \phi(\mathbf{x})| \leq \frac{|\boldsymbol{\mu}^T \phi(\mathbf{x})|}{\sqrt{1 + \pi/(8\,\lambda_{\text{max}}(\mathbf{H}))\|\phi(\mathbf{x})\|^2}}\,,$$
$$\lim_{\sigma_0^2 \to 0} z \circ \phi(\mathbf{x}) = \boldsymbol{\mu}^T \phi(\mathbf{x})\,,$$

where $\mathbf{H}$ is as defined in (i) of Proposition 2.6.

Proof. We can rewrite eq. (13) as follows.

$$|z \circ \phi|_Q(\mathbf{x})| = \frac{|\boldsymbol{\mu}^T \phi(\mathbf{x})|}{\sqrt{1 + \pi/8 \sum_{i=1}^d \lambda_i(\boldsymbol{\Sigma})(\mathbf{Q}^T \phi(\mathbf{x}))_i^2}}\,, \tag{15}$$

where $\boldsymbol{\Sigma} = \mathbf{Q}\,\text{diag}(\lambda_i(\boldsymbol{\Sigma}), \ldots, \lambda_d(\boldsymbol{\Sigma}))\,\mathbf{Q}^T$ is the eigendecomposition of $\boldsymbol{\Sigma}$ (cf. the proof of Lemma A.2). It is therefore clear that the denominator of the r.h.s. is a non-decreasing function of σ_0^2 by virtue of Proposition 2.6. This implies $|z \circ \phi|_Q(\mathbf{x})|$ is a non-increasing function of σ_0^2. For the limits, Proposition 2.6 directly implies $\lambda_{\text{min}}(\boldsymbol{\Sigma})$ has limits $1/\lambda_{\text{max}}(\mathbf{H})$ and 0 whenever $\sigma_0^2 \to \infty$ and $\sigma^2 \to 0$, respectively. From these facts, the right limit is immediate from Lemma A.2 while the left limit is directly obtained by noticing that the denominator of eq. (15) goes to 1 as $\sigma_0^2 \to 0$ and hence also the denominator of $z \circ \phi|_Q(\mathbf{x})$. □

Corollary A.9. *Let $f \circ \phi$, with $\phi : \mathbb{R}^n \to \mathbb{R}^d$ and $f : \mathbb{R}^d \to \mathbb{R}$, be a ReLU network defined by $f \circ \phi(\mathbf{x}) := \mathbf{w}^T \phi(\mathbf{x})$, modeling $p(y = 1|\mathbf{x}, \mathbf{w})$ with $\sigma(f \circ \phi(\mathbf{x}))$. Let $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ be the distribution over $\mathbf{w}$. If either*

(i) $\boldsymbol{\Sigma} = \mathbf{C}^{-1} := \left(\sum_{\mathbf{x} \in \mathcal{D}} \phi(\mathbf{x})\phi(\mathbf{x})^T\right)^{-1}$, *or*

(ii) $\boldsymbol{\Sigma}$ *is obtained via a Laplace approximation w.r.t. a prior $\mathcal{N}(\mathbf{w}|\mathbf{0}, \sigma_0^2 \mathbf{I})$ with $\sigma_0^2 \to \infty$ and suppose $\mathbf{H}$ defined in (i) of Proposition 2.6 is invertible and the ordering of its eigenvalues is the same as that of $\mathbf{C}$, while the eigenvectors are the same as those of $\mathbf{C}$,*

then on any level set of $\boldsymbol{\mu}^T \phi(\mathbf{x})$, the confidence decreases faster in the direction where the training data are sparser in the feature space $\mathbb{R}^d$.

Proof. We start with (i). Let $\mathbf{q}_i, \mathbf{q}_j$ be an arbitrary pair of eigenvectors of $\mathbf{C}$ where $\mathbf{q}_i$ points in the direction where the training data become sparser faster than in the direction $\mathbf{q}_j$. By the property of covariance matrices, therefore $\phi(\mathbf{x})^T \mathbf{C}^{-1} \phi(\mathbf{x})$ increases faster along $\mathbf{q}_i$ than along $\mathbf{q}_j$. Thus the denominator of $z \circ \phi$ decrease faster in the direction of $\mathbf{q}_i$ compared to those in the direction of $\mathbf{q}_j$. This implies that on $\{\mathbf{x} : \boldsymbol{\mu}^T \mathbf{x} = c\}$ for any constant c, the confidence $z \circ \phi$ decreases faster in the direction $\mathbf{q}_i$ than in the direction $\mathbf{q}_j$. Thus, (i) is proven.

For (ii), let $\mathbf{q}_i, \mathbf{q}_j$ be an arbitrary pair of eigenvectors of $\mathbf{H}$ where $\mathbf{q}_i$ points in the direction where the training data become sparser faster than in the direction $\mathbf{q}_j$. The assumption about the eigendecomposition of $\mathbf{H}$ means that the corresponding eigenvalues of $\mathbf{q}_i$ and $\mathbf{q}_j$ are $\lambda_i(\mathbf{H}) < \lambda_j(\mathbf{H})$, implying $\lambda_i(\mathbf{C}) < \lambda_j(\mathbf{C})$. Therefore similar to $\phi(\mathbf{x})^T \mathbf{C}^{-1} \phi(\mathbf{x})$ (cf. the proof of (i)), $\phi(\mathbf{x})^T \mathbf{H}^{-1} \phi(\mathbf{x})$ increases faster in the direction $\mathbf{q}_i$ than in the direction $\mathbf{q}_j$. Now, since we assume that $\sigma_0^2 \to \infty$, eq. (14) in Proposition 2.6 tells us that, $\boldsymbol{\Sigma} = \mathbf{H}^{-1}$. Thus, it holds that in the direction $\mathbf{q}_i$, the denominator of $z \circ \phi$ increases faster than in the direction $\mathbf{q}_j$. By following the same argument as the proof of (i), we get the desired result. □

APPENDIX B SUPPLEMENTARY THEORETICAL ANALYSIS

For completeness, we show that in the case of two-layer networks (i.e. networks with one hidden layer) with saturating activation functions (e.g. sigmoid and tanh), the confidence goes to an exact constant for any input point far away from the training points.

Proposition B.1. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a linear classifier with a parameter $\mathbf{w} \in \mathbb{R}^d$ and let $g : \mathbb{R}^d \to \mathbb{R}^d$ be a component-wise function defined by $g(\mathbf{z}) := (h(\mathbf{z}_1), \dots, h(\mathbf{z}_d))^T$ for some saturating $h : \mathbb{R} \to \mathbb{R}$, which has $\lim_{x\to\infty} h(x) = l$. Let also $\phi : \mathbb{R}^n \to \mathbb{R}^d$ defined as $\phi(\mathbf{x}) := g(\mathbf{V}\mathbf{x} + \mathbf{a})$ for some $\mathbf{V} \in \mathbb{R}^{d\times n}$ and $\mathbf{a} \in \mathbb{R}^d$ be a feature map. Suppose $p(\mathbf{w}|\mathcal{D}) := \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ is the distribution over $\mathbf{w}$. Then for any $\mathbf{x} \in \mathcal{D}$, as $\delta > 0$ goes to infinity*

$$|z \circ \phi(\delta\mathbf{x})| = \frac{|l \sum_{i=1}^{d} \boldsymbol{\mu}_i|}{\sqrt{1 + \pi/8\, l^2 \sum_{i,j=1}^{d} \boldsymbol{\Sigma}_{ij}}} .$$

Proof. By definition,

$$|z \circ \phi(\delta\mathbf{x})| = \frac{|\boldsymbol{\mu}^T g(\delta\mathbf{V}\mathbf{x} + \mathbf{a})|}{\sqrt{1 + \pi/8\, g(\delta\mathbf{V}\mathbf{x} + \mathbf{a})^T \boldsymbol{\Sigma} g(\delta\mathbf{V}\mathbf{x} + \mathbf{a})}} .$$

By definition of g, $\lim_{\delta\to\infty} g(\delta\mathbf{V}\mathbf{x} + \mathbf{a}) = (l, \dots, l)^T =: \mathbf{l}$, which implies

$$\lim_{\delta\to\infty} |z \circ \phi(\delta\mathbf{x})| = \frac{|\boldsymbol{\mu}^T \mathbf{l}|}{\sqrt{1 + \pi/8\, \mathbf{l}^T \boldsymbol{\Sigma} \mathbf{l}}} = \frac{|l \sum_{i=1}^{d} \boldsymbol{\mu}_i|}{\sqrt{1 + \pi/8\, l^2 \sum_{i,j=1}^{d} \boldsymbol{\Sigma}_{ij}}} .$$

□

APPENDIX C LAST-LAYER LAPLACE APPROXIMATION

The theoretical results in the main text essentially tell us that if we have a Gaussian approximate posterior that comes from a Laplace approximation, then using eq. (1) (and eq. (2)) when making a prediction can remedy the overconfidence problem on any ReLU network. In this section we describe a simple Laplace method that can still capture the properties that we have presented in Section 2. Concretely, we apply the Laplace approximation only to the linear last layer of ReLU networks, that have been trained via MAP estimation. For the sake of clarity, we omit the bias in the following and revisit the case where the bias is included at the end of this section.

For the binary classification case, let $g : \mathbb{R}^n \to \mathbb{R}$ be a MAP-trained deep ReLU neural network with a linear last-layer. We can decompose g into a feature map $\phi : \mathbb{R}^n \to \mathbb{R}^d$ and a linear classifier $f : \mathbb{R}^d \to \mathbb{R}^n$ which is defined by $\phi(\mathbf{x}) \mapsto \mathbf{w}_{\text{MAP}}^T \phi(\mathbf{x})$. Based on Proposition 2.6, we can simply perform a Laplace approximation to get the posterior of the weight of the linear classifier f, i.e. $p(\mathbf{w}|\mathcal{D}) = \mathcal{N}(\mathbf{w}|\mathbf{w}_{\text{MAP}}, \mathbf{H}^{-1})$ where $\mathbf{H}$ is the Hessian of the negative log-posterior w.r.t. $\mathbf{w}$ at $\mathbf{w}_{\text{MAP}}$. This Hessian could be obtained via automatic differentiation or via the explicit formula stated in (i) of Proposition 2.6. We emphasize that we only deal with the weight at the last layer of g, i.e. the weight of f, and not the weight of the whole network, thus the inversion of $\mathbf{H}$ is rarely a problem. For instance, large models such as DenseNet-201 (Huang et al., 2017) and ResNet-152 (He et al., 2016a) have $d = 1920$ and $d = 2048$ respectively, implying that we only need to do the inversion of a single 1920×1920 or 2048×2048 matrix once.[4]

In the case of multi-class classification, we now have $f : \mathbb{R}^d \to \mathbb{R}^k : \phi(\mathbf{x}) \to \mathbf{W}_{\text{MAP}}\phi(\mathbf{x})$. We obtain the posterior over a random matrix $\mathbf{W} \in \mathbb{R}^{k\times d}$ in the form $\mathcal{N}(\text{vec}(\mathbf{W})|\text{vec}(\mathbf{W}_{\text{MAP}}), \boldsymbol{\Sigma})$ for some $\boldsymbol{\Sigma} \in \mathbb{R}^{dk\times dk}$ SPD. The procedure is still similar to the one described above, since the exact Hessian of the linear multi-class classifier can still be easily and efficiently obtained via automatic differentiation. Note that in this case we need to invert a $dk \times dk$ matrix, which, depending on the size of k, can be quite large.[5]

For a more efficient procedure, we can make a further approximation to the posterior in the multi-class case by assuming the posterior is a matrix Gaussian distribution. We can use the Kronecker-factored Laplace approximation (KFLA) Ritter et al. (2018), but only for the last layer of the network. That is, we find the Kronecker factorization of the Hessian $\mathbf{H}^{-1} \approx \mathbf{V}^{-1} \otimes \mathbf{U}^{-1}$ via automatic differentiation.[6] Then by definition of a matrix Gaussian (Gupta & Nagar, 1999), we immediately

[4] Based on the implementations available in the TorchVision package.

[5] For example, the ImageNet dataset has $k = 1000$.

[6] In practice, we take the running average of the Kronecker factors of the Hessian over the mini-batches.

obtain the posterior $\mathcal{MN}(\mathbf{W}|\mathbf{W}_{\text{MAP}}, \mathbf{U}, \mathbf{V})$. The distribution of the latent functions is Gaussian, since $\mathbf{f} := \mathbf{W}\phi(\mathbf{x})$ and $p(\mathbf{W}|\mathcal{D}) = \mathcal{MN}(\mathbf{W}|\mathbf{W}_{\text{MAP}}, \mathbf{U}, \mathbf{V})$ imply

$$\begin{aligned} p(\mathbf{f}|\mathcal{D}) &= \mathcal{MN}(\mathbf{f}|\mathbf{W}_{\text{MAP}}\phi(\mathbf{x}), \mathbf{U}, \phi(\mathbf{x})^T\mathbf{V}\phi(\mathbf{x})) \\ &= \mathcal{N}(\mathbf{f}|\mathbf{W}_{\text{MAP}}\phi(\mathbf{x}), (\phi(\mathbf{x})^T\mathbf{V}\phi(\mathbf{x})) \otimes \mathbf{U}) = \mathcal{N}(\mathbf{f}|\mathbf{W}_{\text{MAP}}\phi(\mathbf{x}), (\phi(\mathbf{x})^T\mathbf{V}\phi(\mathbf{x}))\mathbf{U})\,, \end{aligned} \tag{16}$$

where the last equality follows since $(\phi(\mathbf{x})^T\mathbf{V}\phi(\mathbf{x}))$ is a scalar. We then have the following integral

$$p(y = i|\phi(\mathbf{x}), \mathcal{D}) = \int \text{softmax}(\mathbf{f}, i)\,\mathcal{N}(\mathbf{f}|\mathbf{W}_{\text{MAP}}\phi(\mathbf{x}), (\phi(\mathbf{x})^T\mathbf{V}\phi(\mathbf{x}))\mathbf{U})\,d\mathbf{f}\,,$$

which can be approximated via MC-integration.

While one can always assume that the bias trick is already used, i.e. it is absorbed in the weight matrix/vector, in practice when dealing with pre-trained networks, one does not have such liberty. In this case, one can simply assume that the bias b or $\mathbf{b}$ is independent of the weight $\mathbf{w}$ or $\mathbf{W}$, respectively in the two- and multi-class cases. By using the same Laplace approximation procedure, one can easily get $p(b|\mathcal{D}) := \mathcal{N}(b, \mu_b, \sigma_b^2)$ or $p(\mathbf{b}|\mathcal{D}) := \mathcal{N}(\mathbf{b}|\boldsymbol{\mu}_b, \boldsymbol{\Sigma}_b)$. This implies $\mathbf{w}^T\phi(\mathbf{x})+b =: f$ and $\mathbf{W}\phi(\mathbf{x}) + \mathbf{b} =: \mathbf{f}$ are also Gaussians given by

$$\mathcal{N}(f|\boldsymbol{\mu}^T\phi(\mathbf{x})+\mu_b, \phi(\mathbf{x})^T\mathbf{H}^{-1}\phi(\mathbf{x})+\sigma_b^2) \quad \text{and} \quad \mathcal{N}(\mathbf{f}|\mathbf{M}\phi(\mathbf{x})+\mathbf{b}, (\phi(\mathbf{x})^T\otimes\mathbf{I})\boldsymbol{\Sigma}(\phi(\mathbf{x})\otimes\mathbf{I})+\boldsymbol{\Sigma}_b)\,, \tag{17}$$

respectively, with $\mathbf{I} \in \mathbb{R}^{k\times k}$ if $\mathbf{W} \in \mathbb{R}^{k\times d}$ and $\phi(\mathbf{x}) \in \mathbb{R}^d$. Similarly, in the case when the Kronecker-factored approximation is used, we have

$$p(\mathbf{f}|\mathcal{D}) = \mathcal{N}(\mathbf{f}|\mathbf{W}_{\text{MAP}}\phi(\mathbf{x}) + \boldsymbol{\mu}_b, (\phi(\mathbf{x})^T\mathbf{V}\phi(\mathbf{x}))\mathbf{U} + \boldsymbol{\Sigma}_b)\,. \tag{18}$$

Because of the construction above, which is simply done by applying Laplace approximation on the last layer of a ReLU network, we call this method *last layer Laplace approximation* or LLLA for short. We present the pseudocodes of LLLA in Algorithms 1 and 2.

Algorithm 1 LLLA with exact Hessian for binary classification.

```
Input:
    A pre-trained network $f \circ \phi$ with $\mathbf{w}_{\text{MAP}}$ as the weight of $f$, (averaged) cross-entropy loss $\mathcal{L}$,
    training set $\mathcal{D}_{\text{train}}$, test set $\mathcal{D}_{\text{test}}$, mini-batch size $m$, running average weighting $\rho$, and prior
    precision $\tau_0 = 1/\sigma_0^2$.
Output:
    Predictions $\mathcal{P}$ containing $p(y = 1|\mathbf{x}, \mathcal{D}_{\text{train}})\ \forall \mathbf{x} \in \mathcal{D}_{\text{test}}$.
 1: $\boldsymbol{\Lambda} = 0 \in \mathbb{R}^{d\times d}$
 2: for $i = 1, \dots, |\mathcal{D}_{\text{train}}|/m$ do                    ▷ Compute the Hessian over mini-batches
 3:     $\mathbf{X}_i, \mathbf{y}_i = \text{sampleMinibatch}(\mathcal{D}_{\text{train}}, m)$    ▷ Sample a mini-batch from the training set
 4:     $\mathbf{A}_i, \mathbf{B}_i = \text{getHessian}(\mathcal{L}(f \circ \phi(\mathbf{X}_i), \mathbf{y}_i), \mathbf{w}_{\text{MAP}})$    ▷ Via autodiff
 5:     $\boldsymbol{\Lambda}_i = m\,\boldsymbol{\Lambda}_i + \tau_0\,\mathbf{I}$    ▷ I is the identity matrix with the appropriate size
 6:     $\boldsymbol{\Lambda} = \rho\boldsymbol{\Lambda} + (1-\rho)\boldsymbol{\Lambda}_i$
 7: end for
 8: $\boldsymbol{\Sigma} = \boldsymbol{\Lambda}^{-1}$
 9: $p(\mathbf{w}|\mathcal{D}) = \mathcal{N}(\mathbf{w}|\mathbf{w}_{\text{MAP}}, \boldsymbol{\Sigma})$    ▷ Laplace approximation of the posterior
10: $\mathcal{P} = \varnothing$
11: for all $\mathbf{x} \in \mathcal{D}_{\text{test}}$ do    ▷ Predictions
12:     $p = \sigma(\mathbf{w}_{\text{MAP}}^T\phi(\mathbf{x})/(1 + \pi/8\,\phi(\mathbf{x})^T\boldsymbol{\Sigma}\phi(\mathbf{x}))^{1/2})$    ▷ Equation (3)
13:     $\mathcal{P} = \mathcal{P} \cup \{p\}$    ▷ Collect the prediction
14: end for
```

Appendix D Training detail

We train all networks we use in Table 1 for 100 epochs with batch size of 128. The initial learning rates are 0.001 and 0.1 for MNIST and CIFAR-10 experiments, respectively, and we divide them by 10 at epoch 50, 75, and 95. We use ADAM and SGD with 0.9 momentum, respectively. Standard data augmentations, i.e. random crop and standardization are also used for training the network on CIFAR-10. Meanwhile, for LLLA, we use the Kronecker-factored Hessian.

Algorithm 2 LLLA with Kronecker-factored Hessian for multi-class classification.

```
Input:
    A pre-trained network $f \circ \phi$ with $\mathbf{W}_{\text{MAP}}$ as the weight of $f$, (averaged) cross-entropy loss $\mathcal{L}$, training set $\mathcal{D}_{\text{train}}$, test set $\mathcal{D}_{\text{test}}$, mini-batch size $m$, number of samples $s$, running average weighting $\rho$, and prior precision $\tau_0 = 1/\sigma_0^2$.
Output:
    Predictions $\mathcal{P}$ containing $p(y = i|\mathbf{x}, \mathcal{D}_{\text{train}})\ \forall \mathbf{x} \in \mathcal{D}_{\text{test}}\ \forall i \in \{1, \dots, k\}$.
1:  $\mathbf{A} = 0 \in \mathbb{R}^{k \times k}, \mathbf{B} = 0 \in \mathbb{R}^{d \times d}$
2:  for $i = 1, \dots, |\mathcal{D}_{\text{train}}|/m$ do        ▷ Compute the Hessian over mini-batches
3:      $\mathbf{X}_i, \mathbf{y}_i = \text{sampleMinibatch}(\mathcal{D}_{\text{train}}, m)$        ▷ Sample a mini-batch from the training set
4:      $\mathbf{A}_i, \mathbf{B}_i = \text{getKroneckerFactors}(\mathcal{L}(f \circ \phi(\mathbf{X}_i), \mathbf{y}_i), \mathbf{W}_{\text{MAP}})$        ▷ E.g. via KFAC or KFRA
5:      $\mathbf{A}_i = \sqrt{m}\,\mathbf{A}_i + \sqrt{\tau_0}\,\mathbf{I}$        ▷ $\mathbf{I}$ is the identity matrix with the appropriate size
6:      $\mathbf{B}_i = \sqrt{m}\,\mathbf{B}_i + \sqrt{\tau_0}\,\mathbf{I}$
7:      $\mathbf{A} = \rho\mathbf{A} + (1 - \rho)\mathbf{A}_i$
8:      $\mathbf{B} = \rho\mathbf{B} + (1 - \rho)\mathbf{B}_i$
9:  end for
10: $\mathbf{U} = \mathbf{A}^{-1}, \mathbf{V} = \mathbf{B}^{-1}$
11: $p(\mathbf{W}|\mathcal{D}) = \mathcal{MN}(\mathbf{W}|\mathbf{W}_{\text{MAP}}, \mathbf{U}, \mathbf{V})$        ▷ Laplace approximation of the posterior
12: $\mathcal{P} = \varnothing$
13: for all $\mathbf{x} \in \mathcal{D}_{\text{test}}$ do        ▷ Predictions
14:     $p(\mathbf{f}|\mathcal{D}) = \mathcal{N}(\mathbf{f}|\mathbf{W}_{\text{MAP}}\phi(\mathbf{x}), (\phi(\mathbf{x})^T \mathbf{V} \phi(\mathbf{x}))\mathbf{U})$        ▷ Distribution over the outputs of $f$
15:     $\mathbf{p} = \mathbf{0}$
16:     for $j = 1, \dots, s$ do        ▷ Monte-Carlo integration
17:         $\mathbf{f}_j \sim p(\mathbf{f}|\mathcal{D})$
18:         $\mathbf{p} = \mathbf{p} + \text{softmax}(\mathbf{f}_j)$
19:     end for
20:     $\mathbf{p} = \mathbf{p}/m$
21:     $\mathcal{P} = \mathcal{P} \cup \{\mathbf{p}\}$        ▷ Collect the prediction
22: end for
```

Appendix E Further experiments

We further compare the OOD detection performance of LLLA to the temperature scaling method. To find the optimal temperature, we follow the method of Guo et al. (2017). In particular, we use the implementation provided by https://github.com/JonathanWenger/pycalib.

Table 2: Additional OOD detection results. For MMC, the lower the better in the OOD case, but the higher the better for the in-distribution case; for AUROC, the higher the better.

	MAP		**Temp. scaling**		**LLLA**	
Train - Test	MMC	AUROC	MMC	AUROC	MMC	AUROC
MNIST - MNIST	**0.993**	-	0.990	-	0.962	-
MNIST - notMNIST	0.715	0.976	0.715	0.961	**0.511**	**0.978**
MNIST - EMNIST	0.803	0.944	0.803	0.914	**0.608**	**0.947**
MNIST - FMNIST	0.570	0.993	0.570	0.989	**0.383**	**0.995**
CIFAR-10 - CIFAR-10	**0.910**	-	0.875	-	0.880	-
CIFAR-10 - SVHN	0.716	0.813	0.716	0.748	**0.606**	**0.823**
CIFAR-10 - LSUN-CR	0.732	0.810	0.732	0.824	**0.623**	**0.818**