Being Bayesian, Even Just a Bit, Fixes Overconfidence in ReLU Networks paper - review & extension

DELAVANDE Julien, VAKILI Anatole

March 2025

1 Introduction

Neural networks approximate $p(y \mid \mathbf{x}, \boldsymbol{\theta})$, but MAP estimates ignore uncertainty, causing overconfidence, especially on out of distribution (OOD) data, in regions far from training points [1]. Recent work [2] addresses this with temperature scaling and Laplace approximations, improving confidence calibration.

We implement and extend [2], applying these methods to binary and multiclass classification. Our contributions include (1) evaluating alternative activations like Tanh and (2) deriving a closed-form multiclass correction. This strengthens the analysis of Laplace approximations in handling uncertainty and reducing overconfidence on OOD inputs.

2 Bayesian Neural Networks and the Laplace Approximation

2.1 Bayesian Neural Networks

A standard neural network learns a single set of parameters $\boldsymbol{\theta}$ by maximizing the log-likelihood (or posterior in the case of MAP estimation) : $\theta_{\text{MAP}} = \arg\max_\theta \log p(\theta \mid D)$, resulting in deterministic predictions, D beeing the training dataset. This approach ignores uncertainty over $\boldsymbol{\theta}$, often leading to overconfident predictions, particularly on out-of-distribution data. In contrast, a **Bayesian neural network** (BNN) treats $\boldsymbol{\theta}$ as a distribution rather than a fixed point estimate, capturing model uncertainty by integrating over possible values of $\boldsymbol{\theta}$.

For **binary classification**, the posterior predictive distribution (assumed to be Bernoulli) is given by integrating over the parameter uncertainty, where $\sigma(\cdot)$ is the sigmoid function and $f_{\boldsymbol{\theta}}(\mathbf{x})$ is the network's logit output. For **multiclass classification**, we generalize using softmax(.).

$$p_{binary}(y = 1 \mid \mathbf{x}, \mathcal{D}) = \int \sigma(f_{\boldsymbol{\theta}}(\mathbf{x}))p(\boldsymbol{\theta} \mid \mathcal{D})d\boldsymbol{\theta}, \quad p_{multiclass}(y \mid \mathbf{x}, \mathcal{D}) = \int \text{softmax}(\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}))p(\boldsymbol{\theta} \mid \mathcal{D})d\boldsymbol{\theta}.$$

2.2 Laplace approximation

Since exact Bayesian inference is intractable, the **Laplace approximation** models the posterior $p(\boldsymbol{\theta} \mid \mathcal{D})$ as a Gaussian centered at θ_{MAP} with covariance $\boldsymbol{\Sigma} = (-\nabla^2 \log p(\boldsymbol{\theta} \mid \mathcal{D})|_{\theta_{\text{MAP}}})^{-1}$.

$$p(\boldsymbol{\theta} \mid \mathcal{D}) \approx \mathcal{N}(\boldsymbol{\theta} \mid \theta_{\text{MAP}}, \boldsymbol{\Sigma})$$

Laplace comes from approximating the log-posterior via a second-order Taylor expansion around θ_{MAP}, we get $\log p(\theta \mid D) \approx \log p(\theta_{\text{MAP}} \mid D) + \frac{1}{2}(\theta - \theta_{\text{MAP}})^T H(\theta - \theta_{\text{MAP}})$, where $H = \nabla^2 \log p(\theta \mid D)\big|_{\theta_{\text{MAP}}}$. By Bayes' rule, $\log p(\theta \mid D) \propto \log p(D \mid \theta) + \log p(\theta)$, where $\log p(D \mid \theta) = \sum_{i=1}^N \log p(y_i \mid x_i, \theta)$.

Assuming a Gaussian prior $p(\theta) = \mathcal{N}(0, \sigma_0^2 I)$ gives $\log p(\theta) = -\frac{1}{2\sigma_0^2}\|\theta\|^2$, leading to the Hessian $H = \nabla^2 \log p(D \mid \theta) - \frac{1}{\sigma_0^2} I$, where the prior regularizes the curvature of the log-likelihood.

The Laplace approximation thus enables uncertainty-aware predictions while remaining computationally efficient.

3 Closed-form Approximation and asymptotic results

3.1 Binary Laplace Approximation

3.1.1 Close form

Approximating via a **first-order Taylor expansion** around θ_{MAP}, we set $f_{\boldsymbol{\theta}}(x) \approx f_{\theta_{\text{MAP}}}(x) + \mathbf{d}^\top(\boldsymbol{\theta} - \theta_{\text{MAP}})$ with $\mathbf{d} = \nabla_\theta f_{\boldsymbol{\theta}}(x)|_{\theta_{\text{MAP}}}$. Using a **Laplace approximation**, where $\boldsymbol{\theta} \mid x, D \sim \mathcal{N}(\theta_{\text{MAP}}, \boldsymbol{\Sigma})$, the logits follow $p(f_{\boldsymbol{\theta}}(x) \mid x, D) \approx \mathcal{N}\big(f_{\theta_{\text{MAP}}}(x),\, \mathbf{d}^\top \boldsymbol{\Sigma} \mathbf{d}\big)$.

Applying a **probit approximation** [3], we get:

$$p(y = 1 \mid x, D) \approx \sigma\left(\frac{f_{\theta_{\text{MAP}}}(x)}{\sqrt{1 + \frac{\pi}{8}\mathbf{d}^\top \boldsymbol{\Sigma}\mathbf{d}}}\right).$$

This **tempers overconfidence** by incorporating parameter uncertainty.

To improve efficiency, a **last-layer Laplace approximation** applies Bayesian inference only to the final layer's weights $\boldsymbol{w}^L$, treating earlier layers as a deterministic feature extractor $\phi(x)$:

$$p(y = 1 \mid x, D) \approx \int \sigma({\boldsymbol{w}^L}^\top \phi(x)) \mathcal{N}(\boldsymbol{w}^L \mid \boldsymbol{w}^L_{\text{MAP}}, \boldsymbol{\Sigma}) d\boldsymbol{w}^L.$$

This balances complexity and uncertainty estimation, whereas a **full-layer Laplace approximation** models all parameters $\boldsymbol{\theta}$ with a Gaussian posterior, improving uncertainty estimates at the cost of inverting the full Hessian.

3.1.2 Asymptotic Behavior

Hein et al. [1] showed that such networks yield arbitrarily high confidence predictions for inputs far from the training data. Specifically, for an input x scaled by a large factor δ, the network's confidence approaches one as $\delta \to \infty$:

$$\lim_{\delta \to \infty} \sigma(f_{\text{MAP}}(\delta x)) = 1.$$

Now, as $\delta \to \infty$, $f_{\theta_{\text{MAP}}}(\delta x)$ grows unbounded, but the denominator scales accordingly due to the covariance structure. This ensures that the confidence remains bounded:

$$\lim_{\delta \to \infty} \sigma(|z(\delta x)|) \leq \sigma\left(\frac{\|\boldsymbol{u}\|}{\sqrt{s_{min}(\mathbf{J})\frac{\pi}{8}\lambda_{\min}(\boldsymbol{\Sigma})}}\right).$$

Where δ is a scaling factor for simulating out-of-distribution inputs, $u \in \mathbb{R}^n$ is a vector depending only on $\mu = \theta_{\text{MAP}}$, $J = \frac{\partial u}{\partial \theta} \in \mathbb{R}^{n \times p}$ is its Jacobian, $s_{\min}$ is the smallest singular value, and $\lambda_{\min}(\boldsymbol{\Sigma})$ is the smallest eigenvalue of the weight covariance matrix.

Thus, even a simple last-layer Laplace approximation prevents extreme confidence on out-of-distribution inputs, making the network more robust while preserving accuracy on in-distribution data.

3.2 Multiclass Laplace Approximation

3.2.1 Monte-carlo approximation

For **multiclass classification** the network outputs a vector $\mathbf{f}_{\boldsymbol{\theta}}(x) \in \mathbb{R}^k$, which is converted into probabilities using the softmax function and integrating over $p(\boldsymbol{\theta} \mid D)$ gives the **Bayesian predictive distribution**. Unlike the binary case, no closed-form solution exists for this integral. Previous works typically rely on **Monte Carlo (MC) sampling**, drawing parameter samples from the Gaussian posterior and averaging softmax outputs:

$$p(y = i \mid x, D) \approx \frac{1}{S}\sum_{s=1}^{S} \text{softmax}(\mathbf{f}_{\boldsymbol{\theta}^{(s)}}(x))_i, \quad \boldsymbol{\theta}^{(s)} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}).$$

3.2.2 Our Contribution: A Closed-Form Approximation for the Multiclass Case

Inspired by the **probit correction** in the binary case, we extend this approach to the softmax function. Using a **first-order Taylor expansion**:

$$f_{\boldsymbol{\theta}}(x) \approx f_{\theta_{\text{MAP}}}(x) + J(x)(\boldsymbol{\theta} - \theta_{\text{MAP}}),$$

where $J(x)$ is the Jacobian of $f_{\boldsymbol{\theta}}(x)$ at θ_{MAP}, we obtain:

$$f_{\boldsymbol{\theta}}(x) \sim \mathcal{N}\big(f_{\theta_{\text{MAP}}}(x),\, V(x)\big),$$

with covariance $V(x) = J(x)\boldsymbol{\Sigma}J(x)^T$. By analogy with the binary case, we propose the following **logit correction**:

$$z_i(x) := \frac{[f_{\theta_{\text{MAP}}}(x)]_i}{\sqrt{1 + \frac{\pi}{8}[V(x)]_{ii}}},$$

which accounts for uncertainty by **scaling down logits** in proportion to their variance. This leads to the following **closed-form predictive approximation**:

$$p(y = i \mid x, D) \approx \frac{\exp(z_i(x))}{\sum_{j=1}^{k} \exp(z_j(x))}.$$

This formulation provides a computationally efficient alternative to MC sampling while maintaining well-calibrated uncertainty estimates.

3.2.3 Asymptotic Behavior and Robustness

To verify its effectiveness, we analyze the approximation for large inputs $x_\delta = \delta x$. For ReLU networks, logits and covariance scale as:

$$f_{\theta_{\text{MAP}}}(x_\delta) \approx \delta u(x), \quad V(x_\delta) \approx \delta^2 J(x) \mathbf{\Sigma} J(x)^T.$$

Thus, the corrected logits behave as:

$$z_i(x_\delta) \approx \frac{\delta u_i(x)}{\delta \sqrt{\frac{\pi}{8}[V(x)]_{ii}}} = \frac{u_i(x)}{\sqrt{\frac{\pi}{8}[V(x)]_{ii}}}.$$

Since $z_i(x_\delta)$ remains finite, the predictive probabilities converge to a well-calibrated distribution rather than extreme confidence. This confirms that our proposed correction effectively prevents overconfident predictions in high-uncertainty regions.

4 Experiments

We evaluate our uncertainty estimation methods on two synthetic datasets: the binary classification task on the Two Moons dataset and the multiclass classification task on synthetic Gaussian clusters. For both settings, we compare the standard MAP estimator with several uncertainty-aware techniques: temperature scaling, last-layer Laplace approximation (LLLA), and full Laplace approximation (FLA). For multiclass classification, we further compare predictions obtained via Monte Carlo (MC) sampling and our proposed closed-form approximation. We also experiment replacing `ReLU` using `Tanh`, to see if the results still hold.

4.1 Binary Classification Experiments on Two Moons

For binary classification, we use a simple neural network architecture with a two-layer feature extractor and a linear classifier. The network is trained using stochastic gradient descent for 15000 epochs. We experiment with different activation functions (e.g., `ReLU` vs. `Tanh`) to assess their impact on uncertainty calibration.

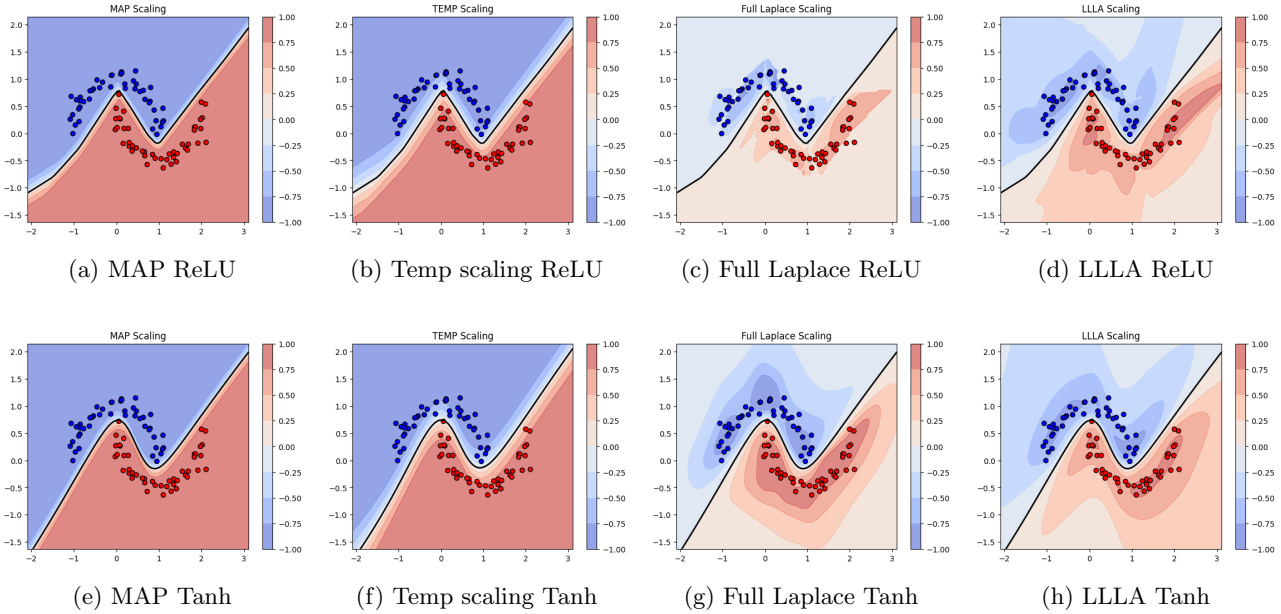
Optimization of Prior Variance. We optimize the standard deviation of the Gaussian prior on the weights (denoted by σ_0) via a grid search over log-variance values. In each iteration, the negative log-likelihood is computed both on in-distribution validation data and on a set of out-of-distribution (OOD) inputs. This combined loss drives the selection of the optimal σ_0, ensuring that both in-domain accuracy and OOD uncertainty are balanced. Let $\hat{D} := \{(\hat{x}_i, \hat{t}_i)\}_{i=1}^m$ be a validation set and $\tilde{D} := \{\tilde{x}_i\}_{i=1}^m$ be an out-of-distribution dataset. We then pick the optimal σ_0^2 by solving the following one-parameter optimization problem:

$$\arg\min_{\sigma_0^2} \left[-\frac{1}{m} \sum_{i=1}^{m} \log p\Big(y = \hat{t}_i \mid \hat{x}_i, D\Big) + \lambda \log p\Big(y = 0.5 \mid \tilde{x}_i, D\Big) \right],$$

where $\lambda \in [0, 1]$ controls the trade-off between both terms. We used $\lambda = 1$.

Temperature Scaling. In addition, temperature scaling is applied post-training to calibrate the logits. The temperature parameter T is optimized using an LBFGS optimizer, minimizing the binary cross-entropy loss on a held-out validation set.

Results. Figure 1 presents the predictive probability maps and calibration plots for the Two Moons dataset. The temperature scaling just temperate the confidence around the decision boundaries. Both LLLA and FLA significantly reduce overconfidence compared to the MAP estimator. The choice of activation function also influences the final calibration: networks with `Tanh` activations tend to yield smoother confidence estimates than those with `ReLU`. Hence, the results of reducing overconfidence seems to hold in this special binary simple case for smother activation function like `Tanh`, even improving the smoothness of the confidence regions.



(a) MAP ReLU (b) Temp scaling ReLU (c) Full Laplace ReLU (d) LLLA ReLU

(e) MAP Tanh (f) Temp scaling Tanh (g) Full Laplace Tanh (h) LLLA Tanh

Figure 1: Comparison of different approaches on a binary classification problem using ReLU (top) and Tanh (bottom). (a) MAP estimate without Bayesian inference, (b) Temperature scaling without Bayesian inference, (c) Full-layer Laplace approximation, and (d) Last-layer Laplace approximation.

4.2 Multiclass Classification Experiments on Gaussian Clusters

For multiclass classification, we design a network that includes a two-layer feature extractor followed by a linear classifier that outputs logits for four classes. The network is optimized over 15000 epochs using a similar stochastic gradient descent scheme.

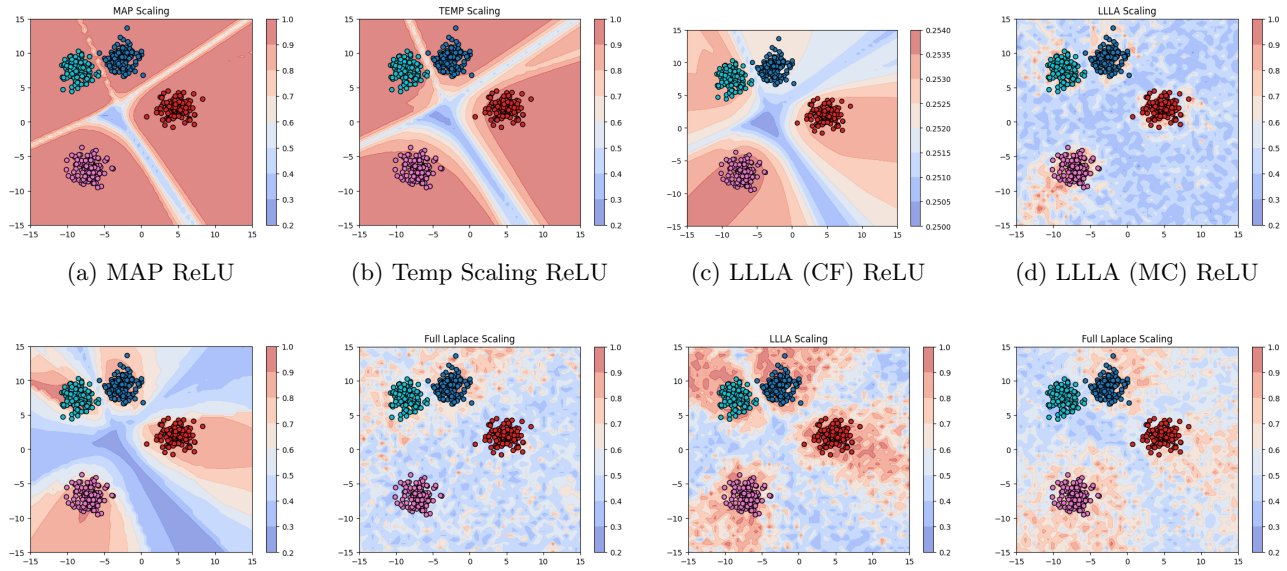
Methods and Optimization. We compare MAP (deterministic baseline), Temperature Scaling (logit calibration), and Laplace approximations (LLLA, FLA) to capture parameter uncertainty. Predictions for LLLA and FLA use Monte Carlo sampling or a closed-form logit correction. For multiclass models, we optimize the prior variance σ_0 and tune the temperature parameter on the validation set to minimize cross-entropy loss.

Results and Analysis. Figure 2 displays the confidence curves for the multiclass experiments. Overall, the MAP model tends to produce overconfident predictions, especially for inputs far from the training data. Temperature scaling increases a bit performance compared to MAP. Both LLLA and FLA considerably mitigate this issue, especially in the LLA case where we can really see islands of confidence around the training data. FLA does not perform as well having confidence more centered around certain classes. Notably, the predictions obtained via the closed-form approximation improves over temps scaling, producing smother confidence regions but do not performing as well as LLLA, probably due to eavy approximations. Finally, `Tanh` in the Monte Carlo setting show promising results but does not perform as good as LLLA (MC) using `ReLU` falling to really create true islands of confidence around the training data.

Computation Times. Table 1 summarizes the approximate computation times for each approach depicted in Figure 2. We see that the MAP baseline is fastest, while the full-layer Laplace scales poorly for larger networks.

Method	Computation Time
MAP / Temp	0.1s
Last-Layer Laplace (MC)	6s
Full-Layer Laplace (MC)	1m30s

Table 1: Computation times for each multiclass method depicted in Figure 2.



(a) MAP ReLU (b) Temp Scaling ReLU (c) LLLA (CF) ReLU (d) LLLA (MC) ReLU

(e) FLA (CF) ReLU (f) FLA (MC) ReLU (g) LLLA (MC) Tanh (h) FLA (MC) Tanh

Figure 2: Comparison of different Bayesian approaches for a 4-class classification problem. (a) MAP estimate without Bayesian inference ReLU, (b) Temperature scaling ReLU, (c-d) Last-Layer Laplace (Closed-Form vs. Monte Carlo) ReLU, (e-f) Full-Layer Laplace (Closed-Form vs. Monte Carlo) ReLU, (g-h) Last-Layer Laplace vs. Full-Layer Laplace Tanh.

5 Limitations

The Laplace approximation faces key limitations, particularly due to the Hessian H not always being positive definite, stemming from the non-convexity of the loss function. This makes the posterior covariance $\mathbf{\Sigma} = (-H)^{-1}$ ill-defined, especially in the full Laplace approximation (FLA).

A common fix is adding a small regularization term, λI, but choosing λ is non-trivial, and diagonal approximations degrade confidence estimates. In the closed-form methods, $\mathbf{d}^T\mathbf{\Sigma}\mathbf{d}$ can become negative in some part of the input space, preventing computation. The prior variance σ_0^2 acts as implicit regularization but is difficult to optimize effectively. These issues worsen for complex tasks, where loss landscapes are more intricate, making Hessian-based uncertainty estimation challenging.

6 Conclusion

We mitigated overconfidence in ReLU networks using Laplace approximations, extending prior work with Tanh activations and a closed-form multiclass correction. Experiments confirmed Last-Layer Laplace effectively calibrates uncertainty with minimal overhead. However, Hessian instabilities remain. Future work could use Cholesky-based regularization, refine priors and scalable approximations for improved robustness.

References

[1] Matthias Hein and Maksym Andriushchenko. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[2] Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being bayesian, even just a bit, fixes overconfidence in relu networks. In *International Conference on Machine Learning (ICML)*, 2020.

[3] David JC MacKay. The evidence framework applied to classification networks. *Neural Computation*, 4(5):720–736, 1992.