ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

Improving the Linearized Laplace Approximation via Quadratic Approximations

Pedro Jiménez[1], Luis A. Ortega[1,2],
Pablo Morales-Álvarez[3,4], Daniel Hernández-Lobato[1,5] *

1- Machine Learning Group - Computer Science Department
Escuela Politécnica Superior, Universidad Autónoma de Madrid, Spain

2- Department of Computer Science, Aalborg University, Copenhagen

3- Visual Information Processing Group
Department of Statistics and Operation Research, University of Granada, Spain

4- Information and Communication Technologies Research Centre (CITIC),
University of Granada, Spain

5- Centro de Investigación Avanzada en Física Fundamental,
Universidad Autónoma de Madrid, Spain

Abstract. Deep neural networks (DNNs) often produce overconfident out-of-distribution predictions, motivating Bayesian uncertainty quantification. The Linearized Laplace Approximation (LLA) achieves this by linearizing the DNN and applying Laplace inference to the resulting model. Importantly, the linear model is also used for prediction. We argue this linearization in the posterior may degrade fidelity to the true Laplace approximation. To alleviate this problem, without increasing significantly the computational cost, we propose the Quadratic Laplace Approximation (QLA). QLA approximates each second order factor in the approximate Laplace log-posterior using a rank-one factor obtained via efficient power iterations. QLA is expected to yield a posterior precision closer to that of the full Laplace without forming the full Hessian, which is typically intractable. For prediction, QLA also uses the linearized model. Empirically, QLA yields modest yet consistent uncertainty estimation improvements over LLA on five regression datasets.

1 Introduction

Deep neural networks (DNNs) are the state-of-the-art models for many supervised learning tasks, achieving remarkable performance across different domains [1]. Despite their predictive power, such models often produce over-confident outputs for inputs that lie far from the training distribution, which undermines their reliability in safety-critical applications [2]. A principled remedy is to adopt

*DHL and LAO acknowledge financial support from project PID2022-139856NB-I00 funded by MCIN, and from project IDEA-CM (TEC-2024/COM-89) and the ELLIS Unit Madrid, funded by the Autonomous Community of Madrid. We acknowledge support from Centro de Computación Científica-Universidad Autónoma de Madrid (CCC-UAM). PMA acknowledges project PID2022-140189OB-C22 funded by MCIN/AEI/10.13039/501100011033 (Spanish Ministry of Science) and grant C-EXP-153-UGR23 funded by Consejería de Universidad, Investigación e Innovación and by the European Union (EU) ERDF Andalusia Program 2021–2027.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

a Bayesian treatment of the model parameters, which yields a posterior distribution over the parameters and, via marginalization, a predictive distribution that quantifies uncertainty alongside point predictions [3].

Exact Bayesian inference is intractable for modern DNNs due to the high-dimensional integrals involved and the large number of parameters. Consequently, approximate inference methods are widely used. The Laplace approximation (LA) [4] is classical and conceptually simple but becomes computationally demanding in deep models because it requires handling very large Hessians. The Linearized Laplace Approximation (LLA) [5] linearizes the network around the parameters and applies LA in the resulting linear model. LLA substantially reduces cost and yields useful predictive distributions in practice. Importantly, using the linear model for prediction also alleviates some of the biases of LA [5].

Linearization in the posterior computation, however, may reduce expressivity and degrade uncertainty calibration. To better balance efficiency and fidelity, we propose the *Quadratic Laplace Approximation* (QLA). QLA builds a second-order expansion of the network around the maximum a posteriori estimation (MAP). However, it approximates the dominant per-datapoint precision terms by a rank-one factor obtained via power iterations with efficient Hessian-vector products. This yields *refined Jacobians* without forming the full Hessian. For prediction, we employ the linearized predictive model, which, as shown in [5], helps to mitigate the effect of the tendency to assign excessive probability mass to regions of parameter space unsupported by data. Empirically, QLA produces small but consistent improvements in uncertainty estimation over LLA across five regression datasets.

2 Linearized Laplace Approximation

We consider a regression problem with data $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where each input $\mathbf{x}_n \in \mathbb{R}^D$ and target $y_n \in \mathbb{R}$. A neural networks is a parametric function $f : \mathbb{R}^D \times \mathbb{R}^P \to \mathbb{R}$ with P learnable parameters, mapping an input $\mathbf{x}_n$ and a parameter vector $\theta \in \mathbb{R}^P$ to a predicted output $f(\mathbf{x}_n, \theta)$.

The target variable is modeled as the network output plus random noise, inducing the conditional observation model $p(y \mid f(\mathbf{x}, \theta))$. Assuming conditional independence across samples, the likelihood factorizes as

$$p(\mathcal{D} \mid \theta) = \prod_{n=1}^{N} p(y_n \mid f(\mathbf{x}_n, \theta)) \ . \tag{1}$$

Given a prior distribution $p(\theta)$, a Bayesian treatment aims to infer the posterior over parameters conditioned on the observed data. However, exact inference is intractable in modern DNNs due to the high dimensionality of θ and the non-linearity of f. We therefore resort to LA, which approximates the posterior by a Gaussian $q(\theta)$ centered at the MAP estimate θ^*:

$$q(\theta) = \mathcal{N}(\theta \mid \theta^*, \Sigma), \qquad \theta^* = \arg\max_\theta \ell(\theta, \mathcal{D}) \ , \tag{2}$$

where $\ell(\theta, \mathcal{D}) = \sum_{n=1}^{N} \log p(y_n \mid f(\mathbf{x}_n, \theta)) + \log p(\theta)$ is the log-joint density. The precision matrix is given by the negative Hessian at θ^*: $\Sigma^{-1} = -\nabla^2_\theta \ell(\theta, \mathcal{D}) \Big|_{\theta=\theta^*}$.

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

To compute this Hessian we use its per-datum decomposition,

$$\nabla^2_\theta \ell\left(\theta, \mathcal{D}\right) = \sum_{n=1}^{N} \nabla^2_\theta \log p\left(y_n \mid f\left(\mathbf{x}_n, \theta\right)\right) + \nabla^2_\theta \log p(\theta)\,. \tag{3}$$

The prior is typically tractable, so we focus on the data-dependent summands:

$$\nabla_\theta \log p\left(y \mid f\left(\mathbf{x}, \theta\right)\right) = \mathbf{r}\left(y; f\right) \mathcal{J}_\theta(\mathbf{x})\,, \tag{4}$$

$$\nabla^2_\theta \log p\left(y \mid f\left(\mathbf{x}, \theta\right)\right) = \mathbf{r}\left(y; f\right) \mathcal{H}_\theta(\mathbf{x}) - \mathcal{J}_\theta(\mathbf{x}) \Lambda(y; f) \mathcal{J}_\theta^\intercal(\mathbf{x})\,, \tag{5}$$

where $\mathcal{J}_\theta(\mathbf{x})$ and $\mathcal{H}_\theta(\mathbf{x})$ are the parameter Jacobian and parameter Hessian of the network, $\mathbf{r}\left(y; f\right)$ denotes the residual, and $\Lambda(y; f)$ denotes per-input noise. The generalized Gauss-Newton (GGN) approximation [6] discards the term $\mathcal{H}_\theta(\mathbf{x})^\intercal \mathbf{r}\left(y; f\right)$, yielding

$$\nabla^2_\theta \log p\left(y \mid f\left(\mathbf{x}, \theta\right)\right) \approx -\mathcal{J}_\theta(\mathbf{x}) \Lambda(y; f) \mathcal{J}_\theta^\intercal(\mathbf{x})\,. \tag{6}$$

This approximation can be seen as a linearization of the DNN around θ^* [5],

$$f\left(\mathbf{x}, \theta\right) \approx f^{\theta^*}_{\text{lin}}(\mathbf{x}, \theta) = f(\mathbf{x}, \theta^*) + \mathcal{J}_{\theta^*}^\intercal(\mathbf{x})(\theta - \theta^*)\,. \tag{7}$$

Applying LA to the linearized model, the posterior approximation becomes

$$q(\theta) = \mathcal{N}\left(\theta \mid \theta^*, \Sigma_{\mathbf{GGN}}\right)\,, \quad \Sigma^{-1}_{\mathbf{GGN}} = \sum_{n=1}^{N} \mathcal{J}_{\theta^*}(\mathbf{x}_n) \Lambda(y_n; f_n) \mathcal{J}_{\theta^*}^\intercal(\mathbf{x}_n) + \mathbf{S}_0^{-1}\,,$$

with $\mathbf{S}_0$ the prior covariance, typically diagonal. Marginalizing under $q(\theta)$ yields the standard Laplace predictive,

$$p\left(y \mid \mathbf{x}, \mathcal{D}\right) = \mathbb{E}_{q(\theta)}\left[p\left(y \mid f\left(\mathbf{x}, \theta\right)\right)\right]\,. \tag{8}$$

Following [5], one may instead use the linearized model for prediction to get better results, giving the generalized linear model (GLM) predictive

$$p_{\text{GLM}}\left(y \mid \mathbf{x}, \mathcal{D}\right) = \mathbb{E}_{q(\theta)}\left[p\left(y \mid f^{\theta^*}_{\text{lin}}\left(\mathbf{x}, \theta\right)\right)\right]\,. \tag{9}$$

For Gaussian observations, this reduces to a Gaussian predictive distribution with mean $f\left(\mathbf{x}, \theta^*\right)$ and variance $\mathcal{J}_{\theta^*}^\intercal(\mathbf{x}) \Sigma_{\mathbf{GGN}} \mathcal{J}_{\theta^*}(\mathbf{x})$.

3 Quadratic Laplace Approximation

LLA provides an efficient framework that substantially simplifies the original LA computation. By locally approximating the DNN with a linear model, LLA avoids the explicit computation of Hessians and, when using the linearized model for predictive inference, yields more stable uncertainty estimates [5].

The linearity assumption in the posterior, however, may bias the posterior variance estimation, which can limit the fidelity of prediction uncertainty estimation. To address this limitation, we propose the *Quadratic Laplace Approximation* (QLA), which extends LLA by adopting a quadratic local approximation of the DNN in the posterior computation:

$$f\left(\mathbf{x}, \theta\right) \approx f^{\theta^*}_{\text{quad}}\left(\mathbf{x}, \theta\right) = f^{\theta^*}_{\text{lin}}\left(\mathbf{x}, \theta\right) + 0.5\left(\theta - \theta^*\right)^\intercal \mathcal{H}_{\theta^*}(\mathbf{x})\left(\theta - \theta^*\right)\,, \tag{10}$$

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

where $\mathcal{J}_{\theta^*}(\mathbf{x})$ and $\mathcal{H}_{\theta^*}(\mathbf{x})$ denote the Jacobian and Hessian of the network evaluated at the MAP estimate θ^*. This corresponds to a *Quadratic Taylor Expansion* (QTE) around θ^*. This is the approach followed in LA.

As in the classical LA, we approximate the posterior by a Gaussian centered at the MAP solution:

$$q(\theta) = \mathcal{N}\left(\theta \mid \theta^*, \Sigma_{\mathbf{QTE}}\right) , \tag{11}$$

with precision

$$\Sigma^{-1}_{\mathbf{QTE}} = \sum_{n=1}^{N} \nabla^2_\theta \log p\left(y_n \mid f^{\theta^*}_{\text{quad}}\left(\mathbf{x}_n, \theta\right)\right) + \mathbf{S}_0^{-1} . \tag{12}$$

Focusing on the data-dependent term, we obtain

$$\nabla^2_\theta \log p\Big(y \mid f^{\theta^*}_{\text{quad}}\left(\mathbf{x}, \theta\right)\Big) \Big|_{\theta=\theta^*} = \mathbf{r}(y; f^{\theta^*}_{\text{quad}})\mathcal{H}_{\theta^*}(\mathbf{x}) - \mathcal{J}_{\theta^*}(\mathbf{x})\Lambda(y; f^{\theta^*}_{\text{quad}})\mathcal{J}^\intercal_{\theta^*}(\mathbf{x}) .$$

At this stage, we face the challenge of explicitly obtaining the network Hessian. However, in frameworks such as *PyTorch*, Hessian-vector products can be efficiently computed without forming the full Hessian matrix. Exploiting this property, we employ the *power iteration* method to estimate the dominant eigenvector of

$$\mathbf{A} := \nabla^2_\theta \log p\Big(y \mid f^{\theta^*}_{\text{quad}}\left(\mathbf{x}, \theta\right)\Big) \Big|_{\theta=\theta^*} . \tag{13}$$

We initialize the procedure with the network Jacobian at the reference point, $\mathbf{z}^{(0)} = \mathcal{J}_{\theta^*}(\mathbf{x})$, and recursively update

$$\mathbf{z}^{(k+1)} = \frac{\mathbf{A}\mathbf{z}^{(k)}}{||\mathbf{A}\mathbf{z}^{(k)}||} . \tag{14}$$

By distributivity, in each multiplication by $\mathbf{A}$, $\mathbf{A}\mathbf{z}^{(k)}$ decomposes as

$$\mathbf{A}\mathbf{z}^{(k)} = \mathbf{r}(y; f^{\theta^*}_{\text{quad}})\mathcal{H}_{\theta^*}(\mathbf{x})\mathbf{z}^{(k)} - \mathcal{J}_{\theta^*}(\mathbf{x})\Lambda(y; f^{\theta^*}_{\text{quad}})\mathcal{J}^\intercal_{\theta^*}(\mathbf{x})\mathbf{z}^{(k)} . \tag{15}$$

Thus, the full Hessian never needs to be constructed, only Hessian-vector products $\mathcal{H}_{\theta^*}(\mathbf{x})\mathbf{z}^{(k)}$ need to be computed, which can be efficiently obtained using automatic differentiation.

After a few iterations, the algorithm converges to the dominant eigenvector, denoted $\hat{\mathbf{z}}$, which can be interpreted as a *refined Jacobian* capturing both first-order information and the most significant quadratic contribution. Under this perspective, we approximate $\mathbf{A}$ by a rank-one factorization $\mathbf{A} \approx \hat{\mathbf{z}}\hat{\mathbf{z}}^\intercal$, leading to the following expression for the Gaussian precision matrix:

$$\Sigma^{-1}_{\mathbf{QTE}} \approx \sum_{n=1}^{N} \hat{\mathbf{z}}_n \hat{\mathbf{z}}_n^\intercal + \mathbf{S}_0^{-1} . \tag{16}$$

Marginalizing under $q(\theta)$ and using the quadratic network approximation yields the approximate predictive distribution

$$p_{\text{QTE}}\left(y \mid \mathbf{x}, \mathcal{D}\right) = \mathbb{E}_{q(\theta)}\left[p\left(y \mid f^{\theta^*}_{\text{quad}}\left(\mathbf{x}, \theta\right)\right)\right] . \tag{17}$$

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

For Gaussian observations, the predictive mean is

$$f(\mathbf{x}, \theta^*) + \frac{1}{2}\mathrm{tr}\left(\Sigma_{\mathbf{QTE}}\mathcal{H}_{\theta^*}(\mathbf{x})\right) , \tag{18}$$

and the predictive variance is

$$\mathcal{J}_{\theta^*}^{\intercal}(\mathbf{x})\Sigma_{\mathbf{QTE}}\mathcal{J}_{\theta^*}(\mathbf{x}) + \frac{1}{2}\mathrm{tr}\left[\left(\mathcal{H}_{\theta^*}(\mathbf{x}_i)\Sigma_{\mathbf{QTE}}\right)^2\right] . \tag{19}$$

Two drawbacks arise at this stage. First, the predictive mean no longer coincides with the network output evaluated at the MAP parameters. Second, the resulting variance, being closer in form to that of the standard LA, does not sufficiently mitigate the approximation's tendency to assign excessive probability mass to unsupported parameter regions.

To address this, we instead use the linearized model for predictions, as in LLA, which represents the best first-order approximation to our quadratic expansion. The resulting predictive distribution is then given by

$$p_{\mathrm{QTE}}\left(y \mid \mathbf{x}, \mathcal{D}\right) = \mathcal{N}\left(y \mid f\left(\mathbf{x}, \theta^*\right), \mathcal{J}_{\theta^*}^{\intercal}(\mathbf{x})\Sigma_{\mathbf{QTE}}\mathcal{J}_{\theta^*}(\mathbf{x})\right) . \tag{20}$$

4 Experiments

We compare LLA and QLA on standard UCI regression datasets: Boston Housing, Energy Efficiency, Yacht Hydrodynamics, Concrete Compressive Strength and Wine Quality. Inputs and targets were standardized prior to training. To assess uncertainty estimation in out-of-distribution samples, we use the *in-between* splits proposed in [7], see also [8]. For each input dimension, the data are sorted, the middle third is held out as test set and the outer two thirds form the training set, yielding D gap splits per dataset.

For each dataset and each gap split we train one DNN using back-propagation (thus D networks per dataset). The DNN hyper-parameters (weight decay $\{0.0, 10^{-4}, 10^{-3}\}$, number of units $\{20, 30, 50\}$, and number of layers $\{1, 2, 3\}$) are found using an inner-cv method on the training set. Then, we obtain the predictive distributions using both LLA and QLA, and compute the chosen metrics on the corresponding test split. In QLA, we use the prior variance estimated by LLA. The estimation is performed by maximizing the marginal likelihood as in [5]. We use 10 iterations of the power method in QLA. Finally, we report metrics averaged across data splits.

The quality of the predictive distribution is measured using the Negative Log Likelihood (NLL), and the Continuous Ranked Probability Score (CRPS) [9]. These are popular metrics used in the literature involving LLA and uncertainty estimation [2]. Because LLA and QLA share the same predictive mean, differences in these metrics reflect differences in predictive variance and therefore directly assess each method's ability to produce realistic uncertainty estimates.

Full obtained results are reported in Table 1. The table shows average test scores (NLL and CRPS, the lower the better) for LLA and QLA, aggregated over the in-between splits, for each dataset. Best results are highlighted in

ESANN 2026 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium) and online event, 22-24 April 2026, i6doc.com publ., ISBN 9782875870964. Available from http://www.i6doc.com/en/.

bold. Overall, QLA yields comparable performance to LLA and produces small, but consistent improvements in the uncertainty metrics across most datasets, although the differences are generally small. These results indicate that QLA refines the posterior precision and modestly improves uncertainty estimation while conserving the predictive performance of the initial DNN.

Table 1: Average metric results across each dataset for LLA and QLA.

Dataset	NLL (↓)		CRPS (↓)	
	LLA	QLA	LLA	QLA
Boston	2.7290 ± 0.0402	$\mathbf{2.7279 \pm 0.0401}$	2.0581 ± 0.0941	$\mathbf{2.0579 \pm 0.0941}$
Energy	2.8704 ± 0.4524	$\mathbf{2.8387 \pm 0.4455}$	2.5091 ± 0.8006	$\mathbf{2.5009 \pm 0.7975}$
Yacht	2.5153 ± 0.0899	$\mathbf{2.5143 \pm 0.0887}$	$\mathbf{1.6166 \pm 0.2008}$	1.6169 ± 0.1999
Concrete	3.3332 ± 0.0227	$\mathbf{3.3331 \pm 0.0228}$	3.8750 ± 0.1003	$\mathbf{3.8742 \pm 0.1003}$
Wine	1.0022 ± 0.0089	$\mathbf{1.0015 \pm 0.0089}$	0.3663 ± 0.0040	$\mathbf{0.3661 \pm 0.0040}$

5 Conclusions

In this paper we have introduced the *Quadratic Laplace Approximation* (QLA), an extension of the Linearized Laplace Approximation that incorporates second-order terms in the log-posterior through efficient rank-one updates obtained via the power iteration method. This yields a refined posterior precision without explicit Hessian computation. For prediction, it retains stable predictive behavior of LLA through the linearized predictive model, which mitigates the tendency to allocate excessive probability mass to unsupported parameter regions [5]. Experiments on five regression datasets show that QLA provides small but consistent improvements in uncertainty metrics over LLA at negligible additional cost. Future work will address extending QLA to classification and multivariate regression, and improving scalability through low-rank or inducing-point based strategies for large neural networks [2].

References

[1] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[2] L. A. Ortega, S. Rodríguez-Santana, and D. Hernández-Lobato. Variational linearized laplace approximation for Bayesian deep learning. In *ICML*, 2024.

[3] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer New York, NY, 2006.

[4] D. J. C. MacKay. Bayesian model comparison and backprop nets. In *Proceedings of the 5th International Conference on Neural Information Processing Systems*, 1991.

[5] A. Immer, M. Korzepa, and M. Bauer. Improving predictions of Bayesian neural nets via local linearization. In *AISTATS*, 2021.

[6] J. Martens and R. Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *ICML*, 2015.

[7] A. Y. K. Foong, Y. Li, J. M. Hernández-Lobato, and R.E. Turner. 'In-between' uncertainty in bayesian neural networks. *Uncertainty and Robustness in Deep Learning*, 2019.

[8] P. Morales-Alvarez, D. Hernández-Lobato, R. Molina, and J. M. Hernández-Lobato. Activation-level uncertainty in deep neural networks. In *ICLR*, 2021.

[9] T. Gneiting and A. E. Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102:359–378, 2007.