Improving predictions of Bayesian neural nets via local linearization

Alexander Immer*
Department of Computer Science
ETH Zurich, Switzerland
Max Planck ETH Center
for Learning Systems

Maciej Korzepa
Technical University of Denmark
Copenhagen, Denmark

Matthias Bauer*
DeepMind
London, UK

Abstract

The generalized Gauss-Newton (GGN) approximation is often used to make practical Bayesian deep learning approaches scalable by replacing a second order derivative with a product of first order derivatives. In this paper we argue that the GGN approximation should be understood as a local linearization of the underlying Bayesian neural network (BNN), which turns the BNN into a generalized linear model (GLM). Because we use this linearized model for posterior inference, we should also predict using this modified model instead of the original one. We refer to this modified predictive as "GLM predictive" and show that it effectively resolves common underfitting problems of the Laplace approximation. It extends previous results in this vein to general likelihoods and has an equivalent Gaussian process formulation, which enables alternative inference schemes for BNNs in function space. We demonstrate the effectiveness of our approach on several standard classification datasets and on out-of-distribution detection. We provide an implementation at https://github.com/AlexImmer/BNN-predictions.

1 Introduction

Inference in Bayesian neural networks (BNNs) usually requires posterior approximations due to intractable integrals and high computational cost. Given such an approximate posterior of the parameters, we can make predictions at new locations by combining the posterior with the original Bayesian neural network likelihood.

*A.I. and M. B. contributed equally. Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

Copyright 2021 by the author(s).

BNN predictive (Eq. (9)) GLM predictive (Eq. (13))

$$p_{\text{BNN}}(\mathbf{y}|\mathbf{x},\mathcal{D}) = \int q(\boldsymbol{\theta}) p(\mathbf{y}|\mathbf{f}(\mathbf{x},\boldsymbol{\theta}))\, \mathrm{d}\boldsymbol{\theta} \xrightarrow{\text{GGN}} p_{\text{GLM}}(\mathbf{y}|\mathbf{x},\mathcal{D}) = \int q(\boldsymbol{\theta}) p(\mathbf{y}|\mathbf{f}_{\text{lin}}^{\boldsymbol{\theta}^*}(\mathbf{x},\boldsymbol{\theta}))\, \mathrm{d}\boldsymbol{\theta}$$

$$\mathbf{f}_{\text{lin}}^{\boldsymbol{\theta}^*}(\mathbf{x},\boldsymbol{\theta}) = \mathbf{f}(\mathbf{x},\boldsymbol{\theta}^*) + \nabla_{\boldsymbol{\theta}} \mathbf{f}(\mathbf{x},\boldsymbol{\theta})|_{\boldsymbol{\theta}=\boldsymbol{\theta}^*}^{\mathsf{T}} (\boldsymbol{\theta}-\boldsymbol{\theta}^*)$$

Figure 1: The generalized Gauss Newton approximation (GGN) turns a Bayesian neural network (BNN) into a generalized linear model (GLM) with same likelihood distribution, but network function $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$ linearized around $\boldsymbol{\theta}^*$. When using GGN, we should also use the GLM in the predictive. $q(\boldsymbol{\theta})$ is an approximate posterior and $\boldsymbol{\theta}^*$ is found by MAP estimation, Eq. (3).

One common posterior approximation is the Laplace approximation (MacKay, 1992a), which has recently seen a revival for modern neural networks (Khan et al., 2019; Ritter et al., 2018). It approximates the posterior by a Gaussian around its maximum and has become computationally feasible through further approximations, most of which build on the *generalized Gauss-Newton approximation* (GGN; Martens and Grosse (2015)). The GGN replaces an expensive second order derivative by a product of first order derivatives, and is often jointly applied with approximate inference in BNNs using the Laplace approximation (Ritter et al., 2018; Foresee and Hagan, 1997; Foong et al., 2019) or variational approximations (Khan et al., 2018; Zhang et al., 2018).

Recently, Foong et al. (2019) showed empirically that predictions using a "linearized Laplace" predictive distribution in this setting can match or outperform other approximate inference approaches, such as mean field variational inference (MFVI) in the *original* BNN model (Blundell et al., 2015) and provide better "in-between" uncertainties for regression. Here we explain that their approach relies on an implicit change in probabilistic model due to the GGN approximation.

More specifically, we argue that the GGN approximation should be considered separately from approximate posterior inference: (1) the GGN approximation locally linearizes the underlying probabilistic model in its parameters and gives rise to a generalized linear model (GLM); (2) approximate inference such as through the Laplace approximation enables posterior inference in this linearized GLM. Because we have done inference in a *modified* probabilistic model (the GLM), we should also predict with this modified model. We call the resulting predictive that uses locally linearized neural network features the "GLM predictive" in contrast to the normally used "BNN predictive" that uses the original BNN features in the likelihood, see Fig. 1.

Our approach generalizes previous results by Khan et al. (2019) and Foong et al. (2019) to non-Gaussian likelihoods. It explains why the GLM predictive works well compared to the BNN predictive, which can show underfitting for Laplace posteriors (Lawrence, 2001), especially when combined with the GGN approximation (Ritter et al., 2018). We demonstrate that our proposed GLM predictive resolves these underfitting problems and consistently outperforms the BNN predictive by a wide margin on several datasets; it is on par or better than the neural network MAP or MFVI. Further, the GLM in weight space can be viewed as an equivalent Gaussian process (GP) in function space, which enables complementary inference approximations. Finally, we show that the proposed GLM predictive can be successfully used for out-of-distribution detection.

2 Background

In this paper we consider supervised learning tasks with inputs $\mathbf{x}_n \in \mathbb{R}^D$ and outputs $\mathbf{y}_n \in \mathbb{R}^C$ (e.g. regression) or $\mathbf{y}_n \in \{0,1\}^C$ (e.g. classification), $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$. We introduce features $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$ with parameters $\boldsymbol{\theta} \in \mathbb{R}^P$ and use a likelihood function $p(\mathcal{D}|\boldsymbol{\theta})$ to map them to the outputs $\mathbf{y}$ using an inverse link function $\mathbf{g}^{-1}$, $\mathbb{E}[\mathbf{y}] = \mathbf{g}^{-1}(\mathbf{f}(\mathbf{x}, \boldsymbol{\theta}))$, such as the sigmoid or softmax:

$$p(\mathcal{D}|\boldsymbol{\theta}) = \textstyle\prod_{n=1}^N p(\mathbf{y}_n|\mathbf{f}(\mathbf{x}_n, \boldsymbol{\theta})), \tag{1}$$

In Bayesian deep learning (BDL) we impose a prior $p(\boldsymbol{\theta})$ on the likelihood parameters and aim to compute their posterior given the data, $p(\boldsymbol{\theta}|\mathcal{D})$; a typical choice is to assume a Gaussian prior $p(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{m}_0, \mathbf{S}_0)$. Given a parameter posterior $p(\boldsymbol{\theta}|\mathcal{D})$, we make probabilistic predictions for new inputs $\mathbf{x}^*$ using the posterior predictive

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathcal{D}) = \mathbb{E}_{p(\boldsymbol{\theta}|\mathcal{D})}[p(\mathbf{y}^*|\mathbf{f}(\mathbf{x}^*, \boldsymbol{\theta}))]. \tag{2}$$

Exact posterior inference requires computation of a high-dimensional integral, the *model evidence* or *marginal likelihood* $p(\mathbf{y}|\mathbf{x}) = \int p(\mathcal{D}|\boldsymbol{\theta})p(\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}$, and is often infeasible. We therefore have to resort to approximate posterior inference techniques, such as mean field variational inference or the Laplace approximation, that approximate $q(\boldsymbol{\theta}) \approx p(\boldsymbol{\theta}|\mathcal{D})$.

Mean-field VI. Popular in recent years, *mean-field variational inference* (MFVI) approximates the posterior $p(\boldsymbol{\theta}|\mathcal{D})$ by a factorized variational distribution $q(\boldsymbol{\theta})$ optimized using an evidence lower bound (ELBO) to the marginal likelihood (Blundell et al., 2015).

MAP. Many practical approaches compute the *maximum a posteriori* (MAP) solution $\boldsymbol{\theta}_{\text{MAP}} = \arg\max_{\boldsymbol{\theta}} \ell(\boldsymbol{\theta}, \mathcal{D})$ and return a point estimate $q(\boldsymbol{\theta}) = \delta(\boldsymbol{\theta} - \boldsymbol{\theta}_{\text{MAP}})$ or a distribution $q(\boldsymbol{\theta})$ around $\boldsymbol{\theta}_{\text{MAP}}$; here $\ell(\boldsymbol{\theta}, \mathcal{D})$ denotes the log joint distribution

$$\ell(\boldsymbol{\theta}, \mathcal{D}) = \textstyle\sum_{n=1}^N \log p(\mathbf{y}_n|\mathbf{f}(\mathbf{x}_n, \boldsymbol{\theta})) + \log p(\boldsymbol{\theta}), \tag{3}$$

Laplace. The *Laplace approximation* (MacKay, 1992a) approximates the posterior by a Gaussian $q(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}_{\text{MAP}}, \Sigma)$ around the mode $\boldsymbol{\theta}_{\text{MAP}}$ with covariance Σ given by the Hessian of the posterior

$$\Sigma = -\left[\nabla^2_{\boldsymbol{\theta}\boldsymbol{\theta}} \log p(\boldsymbol{\theta}|\mathcal{D})\big|_{\boldsymbol{\theta}=\boldsymbol{\theta}_{\text{MAP}}}\right]^{-1}. \tag{4}$$

To compute Σ, we need to compute the Hessian of Eq. (3); the prior terms are usually trivial, such that we focus on the log likelihood. We express the involved Jacobian and Hessian of the log likelihood *per data point* through the *Jacobian* $\mathcal{J} \in \mathbb{R}^{C\times P}$ and *Hessian* $\mathcal{H} \in \mathbb{R}^{C\times P\times P}$ of the feature extractor $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$, $[\mathcal{J}_{\boldsymbol{\theta}}(\mathbf{x})]_{ci} = \frac{\partial f_c(\mathbf{x},\boldsymbol{\theta})}{\partial \theta_i}$ and $[\mathcal{H}_{\boldsymbol{\theta}}(\mathbf{x})]_{cij} = \frac{\partial^2 f_c(\mathbf{x},\boldsymbol{\theta})}{\partial\theta_i\partial\theta_j}$, respectively:

$$\nabla_{\boldsymbol{\theta}} \log p(\mathbf{y}|\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})) = \mathcal{J}_{\boldsymbol{\theta}}(\mathbf{x})^{\mathsf{T}}\mathbf{r}(\mathbf{y}; \mathbf{f}) \tag{5}$$

$$\begin{aligned}\nabla^2_{\boldsymbol{\theta}\boldsymbol{\theta}} \log p(\mathbf{y}|\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})) = {} & \mathcal{H}_{\boldsymbol{\theta}}(\mathbf{x})^{\mathsf{T}}\mathbf{r}(\mathbf{y}; \mathbf{f}) \\ & - \mathcal{J}_{\boldsymbol{\theta}}(\mathbf{x})^{\mathsf{T}}\boldsymbol{\Lambda}(\mathbf{y}; \mathbf{f})\mathcal{J}_{\boldsymbol{\theta}}(\mathbf{x}).\end{aligned} \tag{6}$$

We can interpret $\mathbf{r}(\mathbf{y}; \mathbf{f}) = \nabla_{\mathbf{f}} \log p(\mathbf{y}|\mathbf{f})$ as a residual and $\boldsymbol{\Lambda}(\mathbf{y}; \mathbf{f}) = -\nabla^2_{\mathbf{ff}} \log p(\mathbf{y}|\mathbf{f})$ as per-input noise.

GGN. The network Hessian $\mathcal{H}_{\boldsymbol{\theta}}(\mathbf{x})$ in Eq. (6) is infeasible to compute in practice, such that many approaches employ the *generalized Gauss-Newton* (GGN) approximation, which drops this term (Schraudolph, 2002; Martens, 2020) and approximates Eq. (6) as:

$$\nabla^2_{\boldsymbol{\theta}\boldsymbol{\theta}} \log p(\mathbf{y}|\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})) \approx -\mathcal{J}_{\boldsymbol{\theta}}(\mathbf{x})^{\mathsf{T}}\boldsymbol{\Lambda}(\mathbf{y}; \mathbf{f})\mathcal{J}_{\boldsymbol{\theta}}(\mathbf{x}). \tag{7}$$

This GGN approximation to the Hessian is also guaranteed to be positive semi-definite, whereas the original Hessian Eq. (6) is not. The GGN is often further approximated, and in this paper, we consider the most common cases (Ritter et al., 2018; Zhang et al., 2018), diagonal and Kronecker-factored (KFAC) approximations (Martens and Grosse, 2015; Botev et al., 2017). KFAC approximations are block-diagonal to enable efficient storage and computation of inverses and decompositions while maintaining expressivity compared to a diagonal approximation. Each block corresponding to a parameter group, e.g., a neural network layer, is

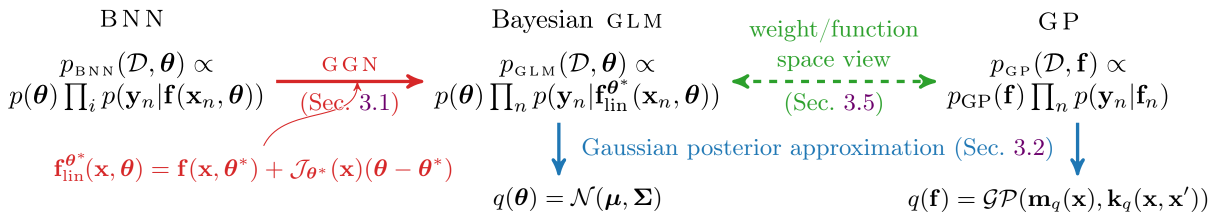

Figure 2: The generalized Gauss Newton approximation (GGN) turns a Bayesian neural network (BNN) into a generalized linear model (GLM) with same prior and likelihood distribution, but network function $\mathbf{f}(\mathbf{x}_n, \boldsymbol{\theta})$ linearized around $\boldsymbol{\theta}^*$ (→). The GLM is equivalent to a Gaussian process (GP) (←-→). Inference is made tractable with a Gaussian posterior approximation (↓), and we predict using the GLM predictive (Eq. (13)).

Kronecker factored; the GGN of the l-th parameter group is approximated as

$$\left[\sum_{n=1}^{N} \mathcal{J}_{\boldsymbol{\theta}}(\mathbf{x}_n)^\mathsf{T} \boldsymbol{\Lambda}(\mathbf{y}_n; \mathbf{f}_n) \mathcal{J}_{\boldsymbol{\theta}}(\mathbf{x}_n)\right]_l \approx \mathbf{Q}_l \otimes \mathbf{W}_l, \quad (8)$$

where $\mathbf{Q}_l$ is the uncentered covariance of the activations and $\mathbf{W}_l$ is computed recursively (Botev et al., 2017). Therefore, $\mathbf{Q}_l$ is quadratic in the size of the input and $\mathbf{W}_l$ in the output of the layer, and both are positive semidefinite. Inversion of the Kronecker approximation is cheap because we only need to invert its factors individually. The Kronecker approximation can be combined with the prior exactly (Grosse and Martens, 2016) or using *dampening* (Ritter et al., 2018). We use the exact version, see App. A.1 for a discussion.

Posterior predictive. Regardless of the posterior approximation, we usually obtain a predictive distribution by integrating the approximate posterior $q(\boldsymbol{\theta})$ against the model likelihood $p(\mathcal{D}|\boldsymbol{\theta})$:

BNN predictive

$$\begin{aligned} p_{\text{BNN}}(\mathbf{y}|\mathbf{x}, \mathcal{D}) &= \mathbb{E}_{q(\boldsymbol{\theta})}\left[p(\mathbf{y}|\mathbf{f}(\mathbf{x}, \boldsymbol{\theta}))\right] \\ &\approx \tfrac{1}{S} \textstyle\sum_s p(\mathbf{y}|\mathbf{f}(\mathbf{x}, \boldsymbol{\theta}_s)), \qquad \boldsymbol{\theta}_s \sim q(\boldsymbol{\theta}) \end{aligned} \quad (9)$$

where we have approximated the (intractable) expectation by Monte Carlo sampling. To distinguish this predictive from our proposed method, we refer to Eq. (9) as BNN predictive. Typically, the BNN predictive distribution is non-Gaussian, because the likelihood can be non-Gaussian and/or $\mathbf{f}$ depends non-linearly on $\boldsymbol{\theta}_s$.

3 Methods

Here, we discuss the effects of the GGN approximation in more detail (Sec. 3.1) and introduce our main contributions, the GLM predictive (Sec. 3.3) and its GP counterpart (Sec. 3.5); see Fig. 2 for an overview.

3.1 Generalized Gauss-Newton turns BNNs into generalized linear models

In Sec. 2 we introduced the GGN as a positive semi-definite approximation to the Hessian by simply dropping the term $\mathcal{H}_{\boldsymbol{\theta}}(\mathbf{x})^\mathsf{T}\mathbf{r}(\mathbf{y}; \mathbf{f})$ in Eq. (7); in other words, we assume that $\mathcal{H}_{\boldsymbol{\theta}}(\mathbf{x})^\mathsf{T}\mathbf{r}(\mathbf{y}; \mathbf{f}) = 0$. Two independently sufficient conditions are commonly used as justification (Bottou et al., 2018): (i) The residual vanishes for all data points, $\mathbf{r}(\mathbf{y}; \mathbf{f}(\mathbf{x}, \boldsymbol{\theta})) = 0\, \forall (\mathbf{x}, \mathbf{y})$, which is true if the network is a perfect predictor. However, this is neither desired, as it indicates overfitting, nor is it realistic. (ii) The Hessian vanishes, $\mathcal{H}_{\boldsymbol{\theta}}(\mathbf{x}) = 0\, \forall \mathbf{x}$, which is true for linear networks and can be enforced by linearizing the network. Hence, an alternative definition uses this second condition as a starting point and defines the GGN through the linearization of the network (Martens and Sutskever, 2011).

In this work, we follow this alternative definition and motivate the GGN approximation as a *local linearization* of the network function $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$,

$$\mathbf{f}_{\text{lin}}^{\boldsymbol{\theta}^*}(\mathbf{x}, \boldsymbol{\theta}) = \mathbf{f}(\mathbf{x}, \boldsymbol{\theta}^*) + \mathcal{J}_{\boldsymbol{\theta}^*}(\mathbf{x})(\boldsymbol{\theta} - \boldsymbol{\theta}^*), \quad (10)$$

at a parameter setting $\boldsymbol{\theta}^*$ (→ in Fig. 2). This linearization reduces the BNN to a Bayesian generalized linear model (GLM) with log joint distribution $\ell_{\text{GLM}}(\boldsymbol{\theta}, \mathcal{D})$

$$\ell_{\text{GLM}}(\boldsymbol{\theta}, \mathcal{D}) = \sum_{n=1}^{N} \log p(\mathbf{y}_n | \mathbf{f}_{\text{lin}}^{\boldsymbol{\theta}^*}(\mathbf{x}_n, \boldsymbol{\theta})) + \log p(\boldsymbol{\theta}), \quad (11)$$

where $\mathbf{f}_{\text{lin}}^{\boldsymbol{\theta}^*}(\mathbf{x}, \boldsymbol{\theta})$ is linear in the parameters $\boldsymbol{\theta}$ but not in the inputs $\mathbf{x}$. In practice, we often choose the linearization point $\boldsymbol{\theta}^*$ to be the MAP estimate found by optimization of Eq. (3). At $\boldsymbol{\theta}^*$ the GGN approximation to the Hessian of the linearized model, Eq. (11), is identical to that of the full model, Eq. (3).

Remark 1. *Applying the* GGN *approximation to the likelihood Hessian turns the underlying probabilistic model locally from a* BNN *into a* GLM.

3.2 Approximate inference in the GLM

Previous works, e.g. Ritter et al. (2018) and Khan et al. (2019), apply the Laplace and the GGN approximation jointly. We refer to the resulting posterior $q(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}_{\text{MAP}}, \boldsymbol{\Sigma}_{\text{GGN}})$ as the "Laplace-GGN posterior", where $\boldsymbol{\Sigma}_{\text{GGN}}$ denotes one of the GGN approxima-

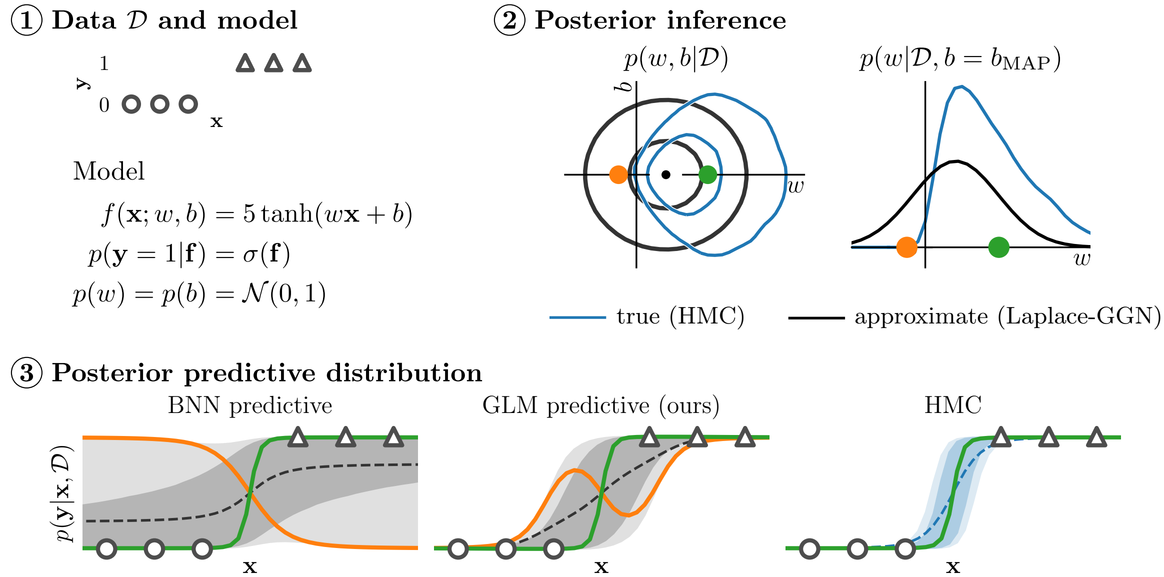

Figure 3: The BNN predictive underfits because some samples can give extremely wrong predictions (an example shown in orange,). The GLM predictive corrects this.
② Laplace-GGN posterior () vs. the true posterior () through 10^5 HMC samples: the Laplace-GGN is symmetric and extends beyond the true, skewed posterior with same MAP. We highlight two posterior samples, one where both distributions have mass () and another where only the Laplace-GGN has mass ().
③ Posterior predictives $p(\mathbf{y}|\mathbf{x}, \mathcal{D})$. The BNN and GLM predictive both use the same Laplace-GGN posterior; while the proposed GLM predictive closely resembles HMC (using the true posterior), the BNN predictive underfits. Underfitting is due to samples from the mismatched region of the posteriors (); while the GLM predictive reasonably extrapolates the behaviour around the MAP, the BNN predictive behaves qualitatively different.
predictive means; innermost 50%/66% of samples.

tions to the covariance introduced in Sec. 2 (full, diagonal, or KFAC). The full covariance case is given by:

$$\mathbf{\Sigma}_{\text{GGN}}^{-1} = \sum_{n=1}^{N} \mathcal{J}_{\boldsymbol{\theta}^*}(\mathbf{x}_n)^\intercal \mathbf{\Lambda}(\mathbf{y}_n; \mathbf{f}_n) \mathcal{J}_{\boldsymbol{\theta}^*}(\mathbf{x}_n) + \mathbf{S}_0^{-1} \tag{12}$$

with prior covariance $\mathbf{S}_0$. In our GLM setting, this corresponds to linearizing the original BNN around $\boldsymbol{\theta}^* = \boldsymbol{\theta}_{\text{MAP}}$ and using the same Laplace-GGN posterior. For large-scale experiments we use this posterior as it is simpler and computationally more feasible than the refinement we describe next. Note that our main contribution is to propose a different predictive (see Sec. 3.3), not a different posterior.

We can use the GLM perspective to *refine* the posterior, because in practise we are only ever approximately able to find $\boldsymbol{\theta}_{\text{MAP}}$ of Eq. (3). We linearize the network around its state after MAP training, $\boldsymbol{\theta}^* \approx \boldsymbol{\theta}_{\text{MAP}}$, and perform inference in the GLM, which typically results in a posterior with mode different from $\boldsymbol{\theta}^*$. The GLM objective Eq. (11) is convex and therefore easier to optimize and guarantees convergence. For general likelihoods, posterior inference is still intractable and we resort to Laplace and variational approximations (see Sec. 2). Both lead to Gaussian posterior approximations $q(\boldsymbol{\theta})$ to $p(\boldsymbol{\theta}|\mathcal{D})$ ($\rightarrow$ in Fig. 2) and are computed iteratively for general likelihoods, see e.g. Bishop (2006, Chapter 4); for Gaussian likelihoods they can be evaluated in a single step. On small-scale experiments (Sec. 4.2) we found that refinement can improve performance but at a higher computational cost; we discuss computational constraints in Sec. 3.6. Nonetheless, the refinement view allows us to consider the GGN approximation separately from the Laplace approximation: the GGN approximation linearizes the network around $\boldsymbol{\theta}^*$, whereas the Laplace approximation is only one of several possible posterior approximations.

Remark 2. *The GGN approximation should be treated as an approximation to the model. It locally linearizes the network features and is independent of posterior inference approximations such as the Laplace approximation or variational inference.*

3.3 The GLM predictive distribution

To make predictions, we combine the approximate posterior with the likelihood; the posterior is the Laplace-GGN posterior or a refinement thereof. Previous works have used the full network features in the likelihood resulting in the BNN predictive (Eq. (9)), which was shown to severely underfit (Ritter et al., 2018). Because we have effectively done inference in the GGN-linearized model, we should instead predict using these modified features:

GLM predictive

$$p_{\text{GLM}}(\mathbf{y}|\mathbf{x},\mathcal{D}) = \mathbb{E}_{q(\boldsymbol{\theta})}\left[p(\mathbf{y}|\mathbf{f}^{\boldsymbol{\theta}^*}_{\text{lin}}(\mathbf{x},\boldsymbol{\theta}))\right] \approx \tfrac{1}{S}\textstyle\sum_s p(\mathbf{y}|\mathbf{f}^{\boldsymbol{\theta}^*}_{\text{lin}}(\mathbf{x},\boldsymbol{\theta}_s)), \quad \boldsymbol{\theta}_s \sim q(\boldsymbol{\theta}). \tag{13}$$

We stress that the GLM predictive in Eq. (13) uses the same approximate posterior as the BNN predictive, Eq. (9), but locally linearized features in the likelihood.

Remark 3. *Because the Laplace-GGN posterior corresponds to the posterior of a linearized model, we should use this linearized model to make predictions. In this sense, the GLM predictive is consistent with Laplace-GGN inference, while the BNN predictive is not.*

3.4 Illustrative example

In Fig. 3 we illustrate the underfitting problem of the BNN predictive on a simple 1d binary classification problem and show how the GLM predictive resolves it.

We consider data sampled from a step function ($y = 0$ for $x < 0$ and $y = 1$ for $x \geq 0$) and use a 2-parameter feature function $\mathbf{f}(\mathbf{x};\boldsymbol{\theta}) = 5\texttt{tanh}(wx + b)$, $\boldsymbol{\theta} = (w, b)$, Bernoulli likelihood, and factorized Gaussian prior on the parameters. The data (● vs ▲ in Fig. 3 *left*) is ambiguous as to where the step from 0 to 1 occurs, such that both parameters w and b are uncertain.

We obtain the true parameter posterior through HMC sampling (Neal, 2010) and find that it is symmetric w.r.t the shift parameter b but *skewed* w.r.t to the slope w (see Fig. 3 *left*). The skewness makes sense as we expect only positive slopes w. The corresponding posterior predictive is certain where we observe data but uncertain around the step, and the predictive mean monotonically increases from 0 to 1 (see Fig. 3 *right*).

The Laplace-GGN posterior as a Gaussian approximation is *symmetric* w.r.t the slope parameter w. It also extends to regions of the parameter space with negative slopes, $w < 0$, which have no mass under the true posterior (see Fig. 3 *left*). Samples ● from this mismatched region result in a monotonically decreasing predictive when using the non-linear features of the BNN predictive (— in Fig. 3 *right*). In contrast, the linearized features of the GLM predictive extrapolate the behaviour around the MAP and result in a more sensible predictive in this case. Samples ● from the matched region behave sensibly for both predictives (— in Fig. 3 *right*). See App. B.1 for further details and an extended discussion.

We derive the following general intuition from this example: The Laplace-GGN approximate posterior may be overly broad compared to the true posterior. Because the feature function $\mathbf{f}(\mathbf{x};\boldsymbol{\theta})$ in the BNN predictive is highly non-linear in $\boldsymbol{\theta}$, samples $\boldsymbol{\theta}_s$ from this mismatched region of the posterior can ulimately result in underfitting. While the GLM predictive maintains non-linearity in the inputs $\mathbf{x}$, its features $\mathbf{f}^{\boldsymbol{\theta}^*}_{\text{lin}}(\mathbf{x};\boldsymbol{\theta})$ are *linear* in the parameters, allowing it to behave more gracefully for samples $\boldsymbol{\theta}_s$ from the mismatched region. In other words, the GLM predictive linearly extrapolates the behavior around the MAP, while the BNN predictive with its non-linear features can behave almost arbitrarily away from the MAP.

Remark 4. *The underfitting of the BNN predictive is not a failure of the Laplace-GGN posterior per se but is due to using a mismatched predictive model.*

3.5 Gaussian process formulation of the GLM

A Bayesian GLM in weight space is equivalent to a Gaussian process (GP) in function space with a particular kernel (◄-► in Fig. 2) (Rasmussen and Williams, 2006). The corresponding log joint is given by $\sum_{n=1}^N \log p(\mathbf{y}_n|\mathbf{f}_n) + \log p(\mathbf{f})$, where the GP prior $p(\mathbf{f})$ is specified by its mean and covariance function that can be computed based on the expectation and covariance of Eq. (10) under the parametric prior $p(\boldsymbol{\theta}) = \mathcal{N}(\mathbf{m}_0, \mathbf{S}_0)$:

$$\begin{aligned} \mathbf{m}(\mathbf{x}) &= \mathbb{E}_{p(\boldsymbol{\theta})}[\mathbf{f}^{\boldsymbol{\theta}^*}_{\text{lin}}(\mathbf{x};\boldsymbol{\theta})] = \mathbf{f}^{\boldsymbol{\theta}^*}_{\text{lin}}(\mathbf{x};\mathbf{m}_0) \\ \mathbf{k}(\mathbf{x},\mathbf{x}') &= \text{Cov}_{p(\boldsymbol{\theta})}[\mathbf{f}^{\boldsymbol{\theta}^*}_{\text{lin}}(\mathbf{x};\boldsymbol{\theta}), \mathbf{f}^{\boldsymbol{\theta}^*}_{\text{lin}}(\mathbf{x}';\boldsymbol{\theta})] \\ &= \mathcal{J}_{\boldsymbol{\theta}^*}(\mathbf{x})\mathbf{S}_0\mathcal{J}_{\boldsymbol{\theta}^*}(\mathbf{x}')^\intercal. \end{aligned} \tag{14}$$

As for the GLM, we now perform approximate inference in this GP model or solve it in closed-form for regression; we denote the GP posterior (approximation) by $q(\mathbf{f})$. For a single output and at $\boldsymbol{\theta}^* = \boldsymbol{\theta}_{\text{MAP}}$ the Laplace-GGN approximation to GP posterior $q(\mathbf{f}^*)$ at a new location $\mathbf{x}^*$ is given by (Rasmussen and Williams, 2006):

$$\begin{aligned} &\mathbf{f}^*|\mathbf{x}^*,\mathcal{D} \sim \mathcal{N}\left(\mathbf{f}(\mathbf{x}^*;\boldsymbol{\theta}^*), \boldsymbol{\sigma}^2_*\right) \\ &\qquad \boldsymbol{\sigma}^2_* = \boldsymbol{K}_{**} - \boldsymbol{K}_{*N}(\boldsymbol{K}_{NN} + \boldsymbol{\Lambda}^{-1}_{NN})^{-1}\boldsymbol{K}_{N*}, \end{aligned} \tag{15}$$

where $\boldsymbol{K}_{*N}$ denotes the kernel $\mathbf{k}(\cdot,\cdot)$ evaluated between $\mathbf{x}^*$ and the N training points, and $\boldsymbol{\Lambda}_{NN}$ is a diagonal matrix with entries $\boldsymbol{\Lambda}(\mathbf{y}_n;\mathbf{f}_n)$ (Eq. (6)). See App. A.2 for the derivation and an extension to multiple outputs. Further, we can perform posterior refinement in function space by optimizing w.r.t. $\mathbf{f}(X) = \mathcal{J}_{\boldsymbol{\theta}^*}(X)\boldsymbol{\theta}$ on a set of data points X, which follows from the linearized formulation in Eq. (10). Analogous to the GLM predictive, we define the GP predictive:

GP predictive

$$p_{\text{GP}}(\mathbf{y}|\mathbf{x},\mathcal{D}) = \mathbb{E}_{q(\mathbf{f})}\left[p(\mathbf{y}|\mathbf{f})\right] \approx \tfrac{1}{S}\textstyle\sum_s p(\mathbf{y}|\mathbf{f}_s), \quad \mathbf{f}_s \sim q(\mathbf{f}). \tag{16}$$

Functional approximations of a GP model are orthogonal to parametric approximations in weight space: While parametric posterior approximations sparsify the covariances of the parameters (e.g. KFAC), functional posterior approximations consider sparsity in data space (e.g. subset of data); also see Sec. 3.6.

Remark 5. *The* GLM *in weight space is equivalent to a* GP *in function space that enables complementary approximations.*

3.6 Computational considerations

Scalability is a major concern for inference in BNNs for large-scale problems. Here, we briefly discuss practical aspects of the Laplace-GGN computations and highlight the influence of approximations as well as implementation details; see App. A.3 for further details.

Jacobians. A key component of the Laplace-GGN approximation and our GLM are the neural network Jacobians $\mathcal{J}_{\boldsymbol{\theta}}(\mathbf{x})$. For common architectures, the complexity of computing and storing a Jacobian is $\mathcal{O}(PC)$ per datapoint for a network with C class outputs and P parameters. Therefore, ad-hoc computation of Jacobians is possible while storage of all Jacobians for an entire data set of size N is often prohibitive ($\mathcal{O}(NPC)$).

Laplace-GGN. Inversion of the full covariance Laplace-GGN approximation (Eq. (12)) scales cubically in the number of parameters ($\mathcal{O}\left(P^3\right)$) and is prohibitive for large neural networks; we only consider it for small problems. The diagonal approximation is a cheap alternative for storage and inversion ($\mathcal{O}(P)$) but misses important posterior correlations and performs worse (see MacKay 1995, Sec. 4.2, and App. B.4). KFAC approximations trade off between feasible computation/storage and the ability to model important dependencies within blocks, e.g., layers (Martens and Grosse, 2015; Botev et al., 2017). Storage and computation only depend on the size of the Kronecker factors and the blocks can be inverted individually. For scalable computation of GGN approximations we use `backpack` for `pytorch` which makes use of additional performance improvements and does not require explicit computation of Jacobians (Dangel et al., 2019).

Parametric predictives. We use S Monte Carlo samples to evaluate the predictives; naively, computation of the BNN predictive ($\mathcal{O}(SP)$) is cheaper than of the GLM predictive ($\mathcal{O}(SPC)$) due to the Jacobians. However, in both cases we can use local reparameterization (Kingma et al., 2015) to sample either the activations per layer (BNN predictive) or the final preactivations directly (GLM predictive) instead.

Functional inference. GP inference replaces inversion of the Hessian in parameter space ($\mathcal{O}\left(P^3\right)$) with inversion of the kernel matrix ($\mathcal{O}\left(N^3\right)$ in computation and $\mathcal{O}\left(N^2\right)$ in memory). Additionally, we need to compute the inner products of N Jacobians to evaluate the kernel. For scalability, we consider a subset of $M \ll N$ training points to construct the kernel (App. A.2) and obtain the GP posterior in $\mathcal{O}(M^3 + M^2P)$ and predictives per new location in $\mathcal{O}(MP + M^2)$. We found that $M \geq 50$ already improves performance over the MAP (see ablations in App. B.4) even when N was orders of magnitude larger; increasing M strictly improved performance. Instead of a naive subset approximation we could also use sparse approximations (Titsias, 2009; Hensman et al., 2015) to scale the kernel computations.

GP and GLM refinement. To perform posterior refinement (cf. Secs. 3.2 and 3.5) efficiently, we have to compute and store the Jacobians on all data, as we require them in every iterative update step. For large networks and datasets we are memory bound and, thus, only consider refinement for small problems in Sec. 4.2.

4 Experiments

We empirically evaluate the proposed GLM predictive for the Laplace-GGN approximated posterior in weight space (Eq. (13)) and the corresponding GP predictive in function space (Eq. (16)). We compare them to the BNN predictive (Eq. (9)) with same posterior for several sparsity structures of the Laplace and variational approximation as well as mean-field VI (BBB, Blundell et al. (2015)) and a dampened KFAC Laplace-GGN approximation with BNN predictive (Ritter et al., 2018).

We consider a second example on $2d$ binary classification (Sec. 4.1), several small-scale classification problems (Sec. 4.2), for which posterior refinement is possible, as well as larger image classification tasks (Sec. 4.3). We close with an application of the GLM predictive to out-of-distribution (OOD) detection (Sec. 4.4). Because the GLM predictive for $\boldsymbol{\theta}^* = \boldsymbol{\theta}_{\text{MAP}}$ is identical to Foong et al. (2019) and Khan et al. (2019), we focus on classification and refer to their works for regression.

In all experiments, we use a diagonal prior, $p(\boldsymbol{\theta}) = \mathcal{N}(0, \delta^{-1}\boldsymbol{I}_P)$, and choose its precision δ based on the negative log likelihood on a validation set for each dataset, architecture, and method. The prior precision δ corresponds to *weight-decay* with factor $\frac{\delta}{N}$. For each task, we first train the network to find a MAP estimate using the objective Eq. (3) and the Adam optimizer (Kingma and Ba, 2015). We then compute the different posteriors and predictives using the values of the parameters after training, $\boldsymbol{\theta}^*$ (details in App. B).

The proposed GLM and GP predictives consistently resolve underfitting problems of the BNN predictive, and are on par or better than other methods considered.

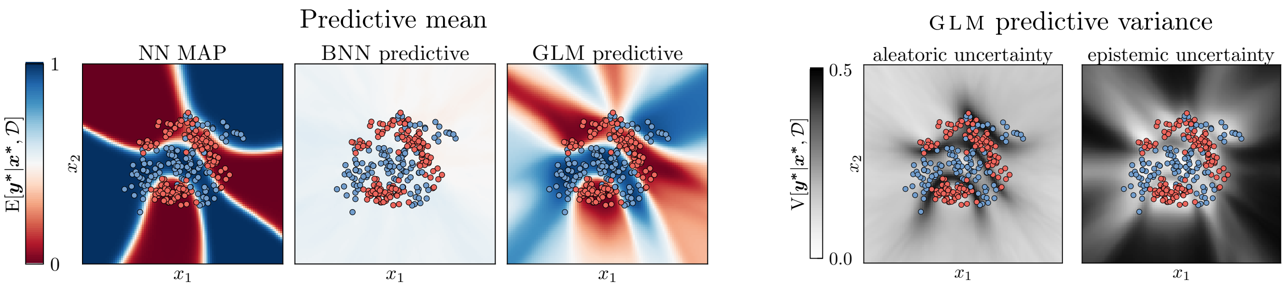

Figure 4: Binary classification on the banana dataset. *left:* Predictive means; the BNN predictive severely underfits; like the MAP, our proposed GLM predictive makes meaningful predictions but also becomes less certain away from the data. *right:* The GLM predictive variance (see App. B.2) decomposes into meaningful aleatoric (data-inherent) uncertainty at class boundaries and epistemic (model-specific) uncertainty away from data.

Dataset	NN MAP	MFVI	BNN	GLM	GLM diag	GLM refine	GLM refine d
australian	**0.31**$\pm$**0.01**	0.34$\pm$0.01	0.42$\pm$0.00	**0.32**$\pm$**0.02**	0.33$\pm$0.01	**0.32**$\pm$**0.02**	**0.31**$\pm$**0.01**
cancer	**0.11**$\pm$**0.02**	**0.11**$\pm$**0.01**	0.19$\pm$0.00	**0.10**$\pm$**0.01**	**0.11**$\pm$**0.01**	**0.11**$\pm$**0.01**	0.12$\pm$0.02
ionosphere	0.35$\pm$0.02	0.41$\pm$0.01	0.50$\pm$0.00	**0.29**$\pm$**0.01**	0.35$\pm$0.01	0.35$\pm$0.05	0.32$\pm$0.03
glass	0.95$\pm$0.03	1.06$\pm$0.01	1.41$\pm$0.00	0.86$\pm$0.01	0.99$\pm$0.01	0.98$\pm$0.07	**0.83**$\pm$**0.02**
vehicle	0.420$\pm$0.007	0.504$\pm$0.006	0.885$\pm$0.002	0.428$\pm$0.005	0.618$\pm$0.003	**0.402**$\pm$**0.007**	0.432$\pm$0.005
waveform	**0.335**$\pm$**0.004**	0.393$\pm$0.003	0.516$\pm$0.002	**0.339**$\pm$**0.004**	0.388$\pm$0.003	**0.335**$\pm$**0.004**	0.364$\pm$0.008
digits	**0.094**$\pm$**0.003**	0.219$\pm$0.004	0.875$\pm$0.002	0.250$\pm$0.002	0.409$\pm$0.002	0.150$\pm$0.002	0.149$\pm$0.008
satellite	0.230$\pm$0.002	0.307$\pm$0.002	0.482$\pm$0.001	0.241$\pm$0.001	0.327$\pm$0.002	**0.227**$\pm$**0.002**	0.248$\pm$0.002

Table 1: Negative test log likelihood (lower is better) on UCI classification tasks (2 hidden layers, 50 `tanh`). The GLM predictive clearly outperforms the BNN predictive; the GLM posterior refined with variational inference is overall the best method. This also holds for accuracy and calibration and on other architectures, see App. B.3.

4.1 Second illustrative example

First, we consider $2d$ binary classification on the banana dataset in Fig. 4. We use a neural network with 2 hidden layers of 50 `tanh` units each and compare the BNN and the GLM predictive for the same full Laplace-GGN posterior (experimental details and additional results for MFVI and diagonal posteriors in App. B.2).

Like in the $1d$ example (Fig. 3), the BNN predictive severely underfits compared to the MAP; its predictive mean is completely washed out and its variance is very large everywhere (see App. B.2). Using the same posterior but the proposed GLM predictive instead resolves this problem. In contrast to the MAP point-estimate, our GLM predictive with Laplace-GGN posterior leads to growing predictive variances away from the data in line with previous observations for regression (Foong et al., 2019; Khan et al., 2019). Moreover, the GLM predictive variance decomposes into meaningful aleatoric (data-inherent) uncertainty at the boundaries between classes and epistemic (model-specific) uncertainty away from the data (Kwon et al., 2020) (Fig. 4 *(right)*). In App. B.2 we show that the GLM predictive easily adapts to deeper and shallower architectures and yields qualitatively similar results in all cases, whereas the BNN predictive performs even worse for deeper (more non-linear) architectures. MFVI requires extensive tuning and yields lower quality results.

4.2 UCI classification

We now compare the different methods on a set of UCI classification tasks on a network with 2 hidden layers of 50 `tanh` units. On this scale, posterior refinement in the GLM using variational inference is feasible as discussed in Sec. 3.2. In Tab. 1, we report the test log predictive probabilities over 10 splits (70% train/15% valid/15% test). See App. B.3 for details and results for accuracy and calibration as well as on other architectures.

Using the same Laplace-GGN posterior, the GLM predictive ("GLM" in Tab. 1) clearly outperforms the BNN predictive ("BNN") on almost all datasets and metrics considered. Moreover, the proposed posterior refinement using variational inference in the GLM ("GLM refine") can further boost performance. The proposed methods also perform consistently better than MFVI on most datasets, even when considering only a diagonal posterior approximation ("... d(iag)"); and they easily adapt to deeper architectures, unlike MFVI, which is often hard to tune (see App. B.3). In Fig. 5 we highlight that the GLM predictive consistently outperforms the BNN predictive for *any* setting of the prior precision hyperparameter δ and that posterior refinement consistently improves over the MAP estimate.

Dataset	Method	Accuracy ↑	NLL ↓	ECE ↓	OOD-AUC ↑
FMNIST	MAP	91.39±0.11	0.258±0.004	0.017±0.001	0.864±0.014
	BNN predictive	84.42±0.12	0.942±0.016	0.411±0.008	0.945±0.002
	BNN predictive (Ritter et al.)	91.20±0.07	0.265±0.004	0.024±0.002	0.947±0.006
	GLM predictive (*ours*)	**92.25±0.10**	**0.244±0.003**	0.012±0.003	**0.955±0.006**
	GP predictive (*ours*)	91.36±0.11	0.250±0.004	**0.007±0.001**	0.918±0.010
CIFAR10	MAP	80.92±0.32	0.605±0.007	0.066±0.004	0.792±0.008
	BNN predictive	21.74±0.80	2.114±0.021	0.095±0.012	0.689±0.020
	BNN predictive (Ritter et al.)	80.78±0.36	0.588±0.005	0.052±0.005	0.783±0.007
	GLM predictive (*ours*)	**81.37±0.15**	0.601±0.008	0.084±0.010	**0.843±0.016**
	GP predictive (*ours*)	81.01±0.32	**0.555±0.008**	**0.017±0.003**	0.820±0.013

Table 2: Accuracy, negative test log likelihood (NLL), expected calibration error (ECE) on the test set, and area under the curve for out-of-distribution detection (OOD-AUC). The proposed methods (GLM and GP predictive) outperform the BNN predictive with same posterior and with dampened (concentrated) posterior (Ritter et al., 2018) as well as the MAP (point-)estimate posterior on most tasks and metrics. See App. B.4 for further results.

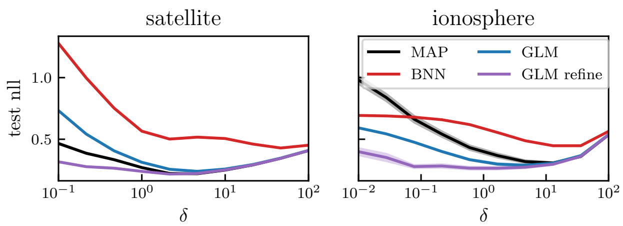

Figure 5: The GLM predictive (—/—) outperforms the BNN predictive (—) *for all* settings of the prior precision hyperparameter δ.

4.3 Image classification

As larger scale problems, we consider image classification on MNIST (LeCun and Cortes, 2010), FashionMNIST (Xiao et al., 2017), and CIFAR10 (Krizhevsky, 2009). We use a KFAC Laplace-GGN approximation for parametric models and a subset posterior approximation with $M = 3200$ data points for the GP. We compare to the MAP estimate and to the BNN predictive with same posterior as well as with dampened posterior (Ritter et al., 2018) and present results for several performance metrics on CNNs in Tab. 2; see App. B.4 for details, additional results on MNIST, other network architectures, and diagonal approximation.

As for the other problems, the GLM predictive consistently outperforms the BNN predictive by a wide margin using the same posterior. It also typically outperforms the BNN predictive with dampened (concentrated) posterior (Ritter et al., 2018), in particular for fully connected networks, see App. B.4. While the GLM predictive performs best on most tasks in terms of accuracy and negative test log likelihood, the GP predictive interestingly achieves better expected calibration error (ECE) (Naeini et al., 2015). We attribute the improved calibration to the GP implicitly using a full-covariance Laplace-GGN, while the parametric approaches are limited to a KFAC approximation of the posterior covariance. However, the GP is limited to a subset of the training data to make predictions; we hypothesize that better sparse approximations could further improve its performance on accuracy and negative log likelihood.

4.4 Out-of-distribution detection

We further evaluate the predictives on out-of-distribution (OOD) detection on the following in-distribution (ID)/OOD pairs: MNIST/FMNIST, FMNIST/MNIST, and CIFAR10/SVHN. Following Osawa et al. (2019) and Ritter et al. (2018), we compare the entropies of the predictive distributions on ID vs OOD data and the associated OOD detection performance measured in terms of the area under the curve (OOD-AUC). We use the same KFAC posterior approximations as in Sec. 4.3; see Apps. B.4 and B.5 for details and additional results on other ID/OOD pairs.

We provide OOD detection performance (OOD-AUC) in Tab. 2 for FMNIST/MNIST and CIFAR10/SVHN and compare the predictive entropy histograms for CIFAR10/SVHN in Fig. 6. Across all tasks considered, we find that the GLM predictive achieves the best OOD detection performance, while the BNN predictive consistently performs worst. The BNN predictive with concentrated (dampened) Laplace-GGN posterior (Ritter et al., 2018) improves over the undampened posterior, but performs worse than the GLM predictive.

5 Related Work

The Laplace approximation for BNNs was first introduced by MacKay (1992a) who applied it to small networks using the full Hessian but also suggested an approximation similar to the generalized Gauss Newton (MacKay, 1992b). Foresee and Hagan (1997) later used the Gauss-Newton for Bayesian regression neural networks with Gaussian likelihoods. The generalized Gauss-Newton (Martens, 2020) in conjunction with scalable factorizations or diagonal Hessian approximations

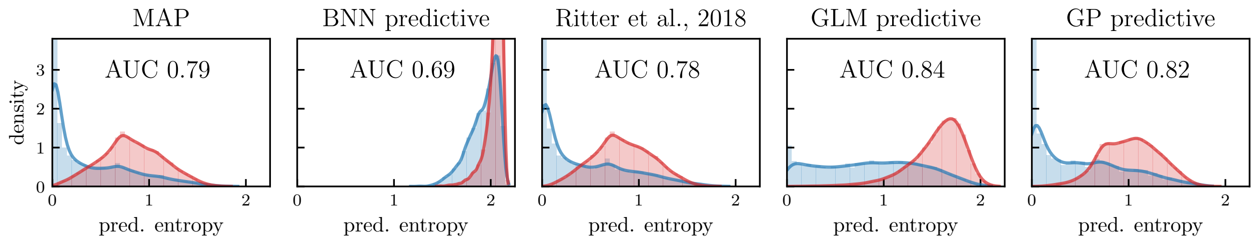

Figure 6: In-distribution (, CIFAR10) vs out-of-distribution (, SVHN) detection using a fully convolutional architecture. The MAP is overconfident while the BNN predictive is underconfident. GP and GLM predictives show best out-of-distribution detection (area under the curve, AUC), also see Tab. 2.

(Martens and Grosse, 2015; Botev et al., 2017) enabled a revival of the Laplace approximation for modern neural networks (Ritter et al., 2018; Khan et al., 2019). Bottou et al. (2018) discuss the linearizing effect of the GGN approximation for MAP or maximum likelihood optimization; here we use this interpretation to obtain a consistent Bayesian predictive.

To address underfitting problems of the Laplace (Lawrence, 2001) that are particularly egregious when combined with the GGN, Ritter et al. (2018) introduced a Kronecker factored Laplace-GGN approximation, which does not seem to suffer in the same way despite using the same BNN predictive. Our analysis and experiments suggest that this is because of an additional ad-hoc approximation they introduce, dampening, which can reduce the posterior covariance (see App. A.1). Dampening is typically used in optimization procedures using Kronecker-factored Hessian approximations (Martens and Grosse, 2015) but can lead to significant distortions when applied to a posterior approximation. In contrast, we use an undampened Laplace-GGN posterior in combination with the GLM predictive to resolve underfitting.

For Gaussian likelihoods our GLM predictive recovers the analytically tractable "linearized Laplace" model (Foong et al., 2019) as well as DNN2GP (Khan et al., 2019). Both apply the Laplace and GGN approximations jointly at the posterior mode $\boldsymbol{\theta}^* = \boldsymbol{\theta}_{\text{MAP}}$ and are limited to regression. We separate the GGN from approximate inference to derive an explicit GLM model for general likelihoods and to justify the GLM predictive. Our experiments generalize their observations to general likelihoods. Khan et al. (2019) introduce DNN2GP to relate inference in (linearized) BNNs to GPs but are limited to Gaussian likelihoods. Our approach builds on their work but considers general likelihoods; therefore, we obtain a similar GP covariance function that is related to the neural tangent kernel (NTK) (Jacot et al., 2018). Our proposed *refinement* is related to training an empirical NTK (Lee et al., 2019). In contrast to the empirical NTK, the GGN corresponds to a local linearization at the MAP and not at a random initialization. Therefore, we expect that these learned feature maps represent the data better.

Out-of-distribution detection has become a benchmark for predictive uncertainties (Nalisnick et al., 2019), on which many recent BNN approaches are evaluated, e.g., Ritter et al. (2018), Osawa et al. (2019), and Wenzel et al. (2020). Our simple change in the predictive also leads to improved OOD detection.

6 Conclusion

In this paper we argued that in Bayesian deep learning, the frequently utilized generalized Gauss-Newton (GGN) approximation should be understood as a modification of the underlying probabilistic model and should be considered separately from approximate posterior inference. Applying the GGN approximation turns a Bayesian neural network (BNN) locally into a generalized linear model or, equivalently, a Gaussian process. Because we then use this linearized model for inference, we should also predict using these modified features in the likelihood rather than the original BNN features. The proposed GLM predictive extends previous results by Khan et al. (2019) and Foong et al. (2019) to general likelihoods and resolves underfitting problems observed e.g. by Ritter et al. (2018). We conclude that underfitting is not due to the Laplace-GGN posterior but is caused by using a mismatched model in the predictive distribution. We illustrated our approach on several simple examples, demonstrated its effectiveness on UCI and image classification tasks, and showed that it can be used for out-of-distribution detection. In future work, we aim to scale our approach further.

Acknowledgments

We thank Emtiyaz Khan for the many fruitful discussions that lead to this work as well as Michalis Titsias and Andrew Foong for feedback on the manuscript. We are also thankful for the RAIDEN computing system and its support team at the RIKEN AIP.

References

Amari, Shun-Ichi (1998). "Natural gradient works efficiently in learning". In: *Neural computation* 2.

Bishop, Christopher M (2006). *Pattern recognition and machine learning*. Springer.

Blundell, Charles, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra (2015). "Weight Uncertainty in Neural Networks". In: *Proceedings of the 32nd International Conference on Machine Learning*.

Botev, Aleksandar, Hippolyt Ritter, and David Barber (2017). "Practical Gauss-Newton Optimisation for Deep Learning". In: *International Conference on Machine Learning*.

Bottou, Léon, Frank E Curtis, and Jorge Nocedal (2018). "Optimization methods for large-scale machine learning". In: *Siam Review* 2.

Dangel, Felix, Frederik Kunstner, and Philipp Hennig (2019). "BackPACK: Packing more into Backprop". In: *Proceedings of 7th International Conference on Learning Representations*.

Foong, Andrew YK, Yingzhen Li, José Miguel Hernández-Lobato, and Richard E Turner (2019). "'In-Between'Uncertainty in Bayesian Neural Networks". In: *arXiv preprint arXiv:1906.11537*.

Foresee, F Dan and Martin T Hagan (1997). "Gauss-Newton approximation to Bayesian learning". In: *International Conference on Neural Networks (ICNN'97)*. IEEE.

Grosse, Roger and James Martens (2016). "A kronecker-factored approximate fisher matrix for convolution layers". In: *International Conference on Machine Learning*. PMLR.

Gupta, Arjun K and Daya K Nagar (1999). *Matrix variate distributions*. CRC Press.

Hensman, James, Alexander Matthews, and Zoubin Ghahramani (2015). "Scalable Variational Gaussian Process Classification". In: *Proceedings of 38th International Conference on Artificial Intelligence and Statistics*.

Jacot, Arthur, Franck Gabriel, and Clément Hongler (2018). "Neural tangent kernel: Convergence and generalization in neural networks". In: *Advances in neural information processing systems*.

Khan, Mohammad Emtiyaz E, Alexander Immer, Ehsan Abedi, and Maciej Korzepa (2019). "Approximate Inference Turns Deep Networks into Gaussian Processes". In: *Advances in Neural Information Processing Systems*.

Khan, Mohammad, Didrik Nielsen, Voot Tangkaratt, Wu Lin, Yarin Gal, and Akash Srivastava (2018). "Fast and Scalable Bayesian Deep Learning by Weight-Perturbation in Adam". In: *International Conference on Machine Learning*.

Kingma, Diederik P and Jimmy Ba (2015). "Adam: A method for stochastic optimization". In: *International Conference on Learning Representations*.

Kingma, Durk P, Tim Salimans, and Max Welling (2015). "Variational dropout and the local reparameterization trick". In: *Advances in neural information processing systems*.

Krizhevsky, Alex (2009). *Learning multiple layers of features from tiny images*. Tech. rep.

Kwon, Yongchan, Joong-Ho Won, Beom Joon Kim, and Myunghee Cho Paik (2020). "Uncertainty quantification using Bayesian neural networks in classification: Application to biomedical image segmentation". In: *Computational Statistics & Data Analysis*.

Lawrence, Neil D. (2001). *Variational Inference in Probabilistic Models*. University of Cambridge.

LeCun, Yann and Corinna Cortes (2010). *MNIST handwritten digit database*. URL: http://yann.lecun.com/exdb/mnist/.

Lee, Jaehoon, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington (2019). "Wide neural networks of any depth evolve as linear models under gradient descent". In: *Advances in neural information processing systems*.

MacKay, David JC (1992a). "Bayesian model comparison and backprop nets". In: *Advances in neural information processing systems*.

MacKay, David JC (1992b). "The evidence framework applied to classification networks". In: *Neural computation* 5.

MacKay, David JC (1995). "Probable networks and plausible predictions—a review of practical Bayesian methods for supervised neural networks". In: *Network: computation in neural systems* 3.

Martens, James (2020). "New Insights and Perspectives on the Natural Gradient Method". In: *Journal of Machine Learning Research* 146.

Martens, James and Roger Grosse (2015). "Optimizing neural networks with kronecker-factored approximate curvature". In: *International conference on machine learning*.

Martens, James and Ilya Sutskever (2011). "Learning Recurrent Neural Networks with Hessian-Free Optimization". In: *Proceedings of the 28th International Conference on Machine Learning (ICML-11).*

Naeini, Mahdi Pakdaman, Gregory F. Cooper, and Milos Hauskrecht (2015). "Obtaining Well Calibrated Probabilities Using Bayesian Binning". In: *AAAI Conference on Artificial Intelligence.*

Nalisnick, Eric, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan (2019). "Do deep generative models know what they don't know?" In: *International Conference on Learning Representations.*

Neal, Radford M. (2010). "MCMC Using Hamiltonian Dynamics". In: *Handbook of Markov Chain Monte Carlo.*

Osawa, Kazuki, Siddharth Swaroop, Mohammad Emtiyaz E Khan, Anirudh Jain, Runa Eschenhagen, Richard E Turner, and Rio Yokota (2019). "Practical deep learning with bayesian principles". In: *Advances in Neural Information Processing Systems.*

Pan, Pingbo, Siddharth Swaroop, Alexander Immer, Runa Eschenhagen, Richard E Turner, and Mohammad Emtiyaz Khan (2020). "Continual Deep Learning by Functional Regularisation of Memorable Past". In: *arXiv preprint arXiv:2004.14070.*

Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer (2017). "Automatic differentiation in pytorch". In:

Rasmussen, Carl Edward and Christopher K. I. Williams (2006). *Gaussian Processes for Machine Learning.* MIT Press.

Ritter, Hippolyt, Aleksandar Botev, and David Barber (2018). "A scalable laplace approximation for neural networks". In: *International Conference on Learning Representations.*

Schneider, Frank, Lukas Balles, and Philipp Hennig (2018). "DeepOBS: A Deep Learning Optimizer Benchmark Suite". In: *International Conference on Learning Representations.*

Schraudolph, Nicol N (2002). "Fast curvature matrix-vector products for second-order gradient descent". In: *Neural computation* 7.

Titsias, Michalis (2009). "Variational Learning of Inducing Variables in Sparse Gaussian Processes". In: *International Conference on Artificial Intelligence and Statistics.*

Wenzel, Florian, Kevin Roth, Bastiaan S Veeling, Jakub Świątkowski, Linh Tran, Stephan Mandt, Jasper Snoek, Tim Salimans, Rodolphe Jenatton, and Sebastian Nowozin (2020). "How good is the bayes posterior in deep neural networks really?" In: *International Conference on Machine Learning.*

Xiao, Han, Kashif Rasul, and Roland Vollgraf (28, 2017). *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms.*

Zhang, Guodong, Shengyang Sun, David Duvenaud, and Roger Grosse (2018). "Noisy Natural Gradient as Variational Inference". In: *International Conference on Machine Learning.*