Biasing Laplace Approximations towards Variational Bayes

Philipp von Bachmann
University of Tübingen

Abstract

Although Laplace Approximations (LA) are a Bayesian deep learning tool which can be applied post-hoc to any trained model with small computational overhead, they are still considered inferior to Variational Bayes (VB). However recent work has drawn a connection by between LA and VB at the *maximum-a-posteriori* (MAP) estimate, but relies on the assumption that MAP and VB converge to the same point which seems not to be the case. To close this gap, we theoretically compare their training objectives and propose that the main advantage of VB is retaining high entropy. We test this hypothesis on multiple algorithms biasing MAP towards high entropy and investigate the resulting changes in the predictive distribution when applying the LA on top. The experiments show that adding this bias reflects the training behaviour of VB better. The resulting changes in the LA are less definite which requires further experiments on real-world datasets. Since our theoretical analysis is not bound to specific methods, it additionally opens up the space for other new algorithms in between VB and MAP to be considered.

1 Introduction

Bayesian neural networks (BNN) Neal [2012] are essential to modeling uncertainty in deep learning by adding epistemic uncertainty - uncertainty over model parameters that depends on the training data. Especially for out-of-distribution (OOD) data, BNNs have the potential to improve upon traditional *maximum a posteriori* estimates in various aspects from reducing overconfidence Kristiadi et al. [2020] to improving outlier detection in applications like computer vision Kendall and Gal [2017].

One approach to estimate epistemic uncertainty is to treat the model from a probabilistic point of view, see for example Murphy [2022]. Instead of searching for the best possible parameter values, training BNNs aims at estimating a *posterior* probability distribution over the model parameters, given the training data. When evaluating the model on test data, the parameter distribution is incorporated into the predictive distribution. A common method with strong theoretical background is called Variational Bayes, see Blundell et al. [2015]. In practice however, VB is rarely applied to larger networks and datasets, mainly

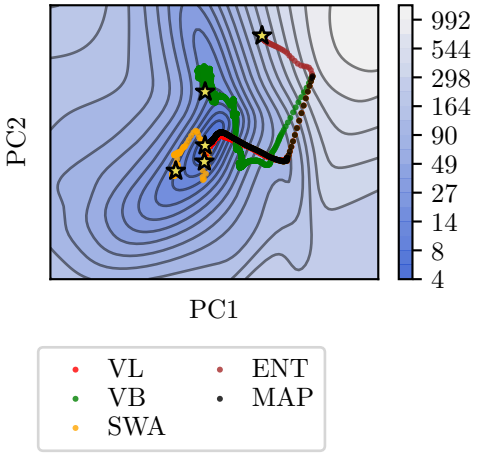

Figure 1: Training trajectories of different algorithms on 2D Toy dataset. X and y axis show first two principal components to visualize the high-dimensional parameter space. Contour lines show the negative log-joint of the data. We observe that VL and SWA behave similar to VB by converging outside the local minimum found by MAP.

because the *posterior* distribution is intractable for neural networks and requires considerable computational effort Graves [2011] Daxberger et al. [2021].

Laplace approximations Mackay [1992]Ritter et al. [2018] provide an alternative approach by only approximating a distribution at the MAP estimate without affecting the training procedure. Still, LA are rarely considered as an Bayesian approach, most likely due to misconceptions like beeing too simple or computationally expensive compared to VB Daxberger et al. [2021]. However Khan and Rue [2021] have shown that LA and VB are closely connected at the point of convergence. This however assumes that training with MAP and VB converging to a similar point, which is not the case as shown in fig. 1 for a small Toy dataset.

Using these insights, we investigate the differences between MAP and VB and propose an approach for LA to resemble VB more closely. Our main contributions are as follows:

- We theoretically investigate the differences between MAP and VB and conclude that retaining high entropy or equivalently converging to flat minima is the main advantage of VB.
- We propose that combining the LA with algorithms to bias MAP towards high entropy or flat minima will result in the LA predictive distributions to resemble VB closer.
- We test our hypothesis on multiple established algorithms biasing MAP towards these properties for different datasets. Our evaluation focuses on both the training behaviour as well as the resulting changes in the predictive distribution of LA.

2 Background

2.1 MAP inference

Bayesian learning amounts to finding the *posterior* distribution $p(\theta|D)$ over the parameters of a model θ given some data D. Rearranging with Bayes Rule requires us to know the evidence $p(D)$, which is generally intractable. *Maximum a-posteriori* inference tackles this by instead finding the *posterior* mode

$$\arg\max_{\theta} p(\theta|D) = \arg\max_{\theta} \frac{p(D,\theta)}{p(D)} = \arg\max_{\theta} p(D|\theta) \cdot p(\theta) \tag{1}$$

where we can drop the evidence $p(D)$ since it is a constant. We can futher split the joint $p(\theta, D)$ in the likelihood $p(D|\theta)$ and the prior $p(\theta)$. The likelihood is task dependent - regression usually uses a normal and classification a categorical distribution as likelihood. The prior can be chosen freely and is often set to a normal distribution centered at 0 with the variance fine tuned to the task. Finally, we usually take the logarithm and reformulate the problem as an minimization problem leading to

$$\arg\min_{\theta} -\log(p(D,\theta)). \tag{2}$$

The main problem with MAP estimation is that it results in a point-estimate instead of a full distribution over the parameters, which removes the uncertainty over model parameters.

2.2 Variational inference

Variational Bayesian inference amounts to minimizing the KL-divergence $D_{KL}[q(\theta)\|p(\theta|D)]$ between the intractable *posterior* $p(\theta|D)$ and a freely selectable variational distribution $q(\theta)$, by optimizing the parameters of q. Further it can be shown that this is equivalent to minimizing the evidence lower bound (ELBO) F (shown in its two common forms, $H(q)$ denotes the entropy)

$$F(q) = E_q[-\log(p(D,\theta))] - H(q(\theta)) \tag{3}$$

$$= E_q[-\log(p(D|\theta))] + D_{\mathrm{KL}}(q(\theta)\|p(\theta)) \tag{4}$$

by noting that the evidence $\log(p(D))$ is constant and the KL-divergence is always greater or equal to 0.

$$-\log(p(D)) = F(q) - D_{KL}[q(\theta)\|p(\theta|D)] \leq F(q) \tag{5}$$

The advantage of VB is that one can choose a tractable distribution for q, which is often set to a normal distribution for simplicity resulting in $q(\theta) = \mathcal{N}(\theta, \mu, \sigma^2)$ with parameters μ and σ.

Even when choosing a normal distribution, evaluating the ELBO can be very challenging, especially in the context of neural networks. Since the likelihood is a nonlinear function in this case, the expectation cannot be decomposed and is in practice often approximated by Monte-Carlo (MC) sampling like in eq. (6). However this is quite expensive and therefore in practice a low number of samples n need to be used to reach sufficient speed, resulting in a noisy training which can become unstable and requires careful hyperparameter tuning.

$$E_q[-\log(p(D|\theta))] \approx \frac{1}{n}\sum_{i=1}^{n} -\log(p(D|\theta_i)) \quad \text{with} \quad \theta_i \sim q \tag{6}$$

2.3 Flat minima

Which properties are required for deep learning models to generalize well has been an interest of research for a long time. An early work often cited is Hochreiter and Schmidhuber [1997], who show that convergence to a flat minimum results in a lower generalization error. How to measure flatness is still under debate, Hochreiter and Schmidhuber [1997] use the size of the neighbourhood around the estimate in which the loss does not change too much while Chaudhari et al. [2019] and Sagun et al. [2016] use the eigenvalue spectrum of the Hessian. Convergence towards flat minima can be achieved in various ways, for example through small batch training Keskar et al. [2016] or smoothing of the loss landscape Chaudhari et al. [2019].Khan and Rue [2021] claim that VB converges to a flat minimum as well. This is because a flat minimum is more robust to retain low loss under the weight pertubations of q in the expected log-likelihood. This in turn makes choosing q with high entropy $H(q)$ possible. Better generalization has been empirically confirmed by Blundell et al. [2015] however Wenzel et al. [2020] show that the reguralization effect can be too strong, leading to underfitting.

2.4 Laplace approximations

The Laplace approximation Mackay [1992] Ritter et al. [2018] is derived by a second-order Taylor expansion of the *posterior* around the MAP estimate θ_{MAP}. The resulting approximation is a normal distribution of the following form:

$$p(\theta|D) = \mathcal{N}(\theta, \theta_{MAP}, \Sigma) \tag{7}$$

$$\Sigma = (-\nabla^2_\theta \log(p(D,\theta))|_{\theta_{\text{MAP}}})^{-1} \tag{8}$$

Compared to VB, the LA is a post-hoc method and therefore requires no modification during training. This especially means no additional complexity during training as well as beeing able to apply other optimization research without changes. Σ contains a hessian over θ which can be expensive to compute and needs to be positive-definite to be valid for a normal distribution. Therefore approximations like the Generalized-Gauss-Newton (GGN) Schraudolph [2002] are used in practice.

Even tough LA and VB seems to be quite different approaches at first-glance Khan and Rue [2021] Khan et al. [2018] investigated their connection. Both showed, that using Natural-Gradient Descent for VB results in updating the precision s with learning rate ρ by

$$s_{t+1} = (1-\rho_t)s_t + \rho_t E_q[-\nabla^2_\theta \log(p(D,\theta))|_{\theta_t}] \tag{9}$$

which contains the hessian similar to eq. (8), however inside an expectation. Positive definiteness needs to hold as well which is why the authors propose to approximate the hessian by GGN or gradient magnitude (GM).

3 Entropy - The missing ingredient in Laplace

Section 2.4 showed how using Natural Gradients for VB results in variance updates similar to LA. Khan and Rue [2021] also derived the mean update steps

$$\theta_{t+1} = \theta_t - \rho_t \frac{1}{s_{t+1}} \nabla_\theta - \log(p(D, \theta))|_{\theta_t} \tag{10}$$

which resembles Newton's method with variance estimate s_{t+1} from eq. (9). If at some timestep c (usually convergence) $\nabla_\theta log(p(D, \theta))|_{\theta_c} = 0$ then it follows that

$$\theta_{c+i} = \theta_c \quad \text{for } i \in \mathbb{N} \qquad s_{c+i} \xrightarrow{i \to \infty} E_q[-\nabla^2_\theta \log(p(D, \theta))|_{\theta_c}]. \tag{11}$$

We can identify the right side of s as the LA up to the expectation. Therefore we can conclude that given the same point of convergence $\theta_{\text{MAP}} = \theta_{\text{VB}}$, the LA will be equal to Natural Gradient VB. This equivalence however relies on the assumption that both MAP and VB training converge to the same mean estimate μ_{MAP}, which Khan and Rue [2021] argue is not the case. Therefore, we investigate the differences between the training objective of VB and MAP to close this gap.

First, VB computes the expectation over the log joint, whereas MAP only uses the current weight estimate θ_t. However, we can view MAP as approximating the expectation at its mode through the delta approximation

$$E_q[-\log(p(D, \theta))] \approx -\log(p(D, \theta_{\text{mode}})). \tag{12}$$

Although using a delta approximation appears to be inaccurate, in practice VB also needs to approximate the expectation, for example through MC sampling as in eq. (6), often with a single sample. Additionally, MAP does not contain the entropy term $H(q)$ from the ELBO, eq. (3). Encouraging high entropy however results in convergence to flat minima as shown in section 2.2 which is also preferable for generalization. Figure 2 illustrates these differences visually in an example loss landscape. MAP will prefer the left, sharp minimum since it is the global minimum. Evaluating the expected log-likelihood of VB at this minimum will result in a higher loss if the variational distribution should have high entropy, since the minimum is surrounded by walls of high loss. Therefore VB chooses the right local minimum which allows high entropy while still having low loss.

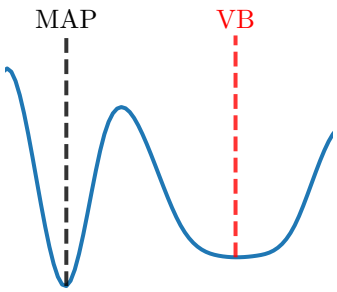

Figure 2: Example loss landscape: When initializing at the left, MAP chooses the narrow but deep minimum, while VB chooses the flatter but higher.

In general, there exist various ways to decrease these differences. We argue that the entropy term is more important to retain than the expectation. While the delta approximation of MAP can be inaccurate, in practice VB uses MC with a low number of samples as a rather simplistic approximation while achieving good results. Entropy on the other hand is not included in the MAP loss at all, which is why we focus on retaining this property. We propose that when retaining high entropy, MAP in combination with LA reflects the behaviour of VB. We try to verify this by combining the MAP objective with different algorithms for high entropy. A method closely related to VB is Variational Laplace (VL) which directly plugs the LA into the ELBO. We further investigate methods retaining flat minima, which is one of the effects of VB as shown. Relaxing the entropy property allows for a broader spectrum of methods to be considered, of which we choose Entropy-SGD and Stochatic Weight Averaging (SWA). Here, especially SWA provides a very cost effective approach.

3.1 Variational Laplace

Variational Laplace (Daunizeau [2017]) is a method closely related to VB, since it arises from plugging the Laplace solution into the ELBO. More detailed, the authors use a second-order Taylor approximation of the ELBO:

$$F(q) \approx -\log(p(D, \mu)) + \frac{1}{2}\text{tr}(-\nabla^2_\theta \log(p(D, \theta))|_\mu \cdot \Sigma) + \frac{1}{2}\log(|\Sigma|) + c \tag{13}$$

By replacing $\Sigma = (-\nabla^2_\theta \log(p(D,\theta))|_\mu)^{-1}$ from the Laplace approximation, the second-term cancels resulting in the VL training objective

$$F_{\text{VL}}(q) = \log(p(D,\mu)) + \tau \log(|(-\nabla^2_\theta \log(p(D,\theta))|_\mu)^{-1}|) \tag{14}$$

The authors motivate VL further by showing that using Laplace maximizes the ELBO in the case of Gaussian variational distributions. However, this is restricted to the point of convergence. We also note that eq. (14) can be derived from the ELBO in eq. (3) by using a delta approximation of the expected log-joint and plugging the LA into the variance estimate, since the entropy of a multivariate normal distribution is given by

$$H(\mathcal{N}(\mu,\Sigma)) = \frac{1}{2}\log(|\Sigma|) + c. \tag{15}$$

VL provides an inutitive link between high entropy and flat minima, by using the Hessian as a measure of flatness as in section 2.3. Further simplifying

$$\log(|(-\nabla^2_\theta \log(p(D,\theta))|_\mu)^{-1}|) = -\log(|(-\nabla^2_\theta \log(p(D,\theta))|_\mu)|) \tag{16}$$

shows that that maximizing entropy means minimizing Hessian eigenvalues. Compared to VB, VL has less parameters, since the variance approximation is defined through the mean. The objective contains no expectation over the variational distribution and is therefore less noisy. However, VL relies on the computation of second-order derivatives for the loss function, and thus third-order derivatives for the weight update. Using approximations like the Generalized-Gauss-Newton eases this by replacing the Hessian with first order derivatives. Additionally, this ensures that the Hessian stays positive-definite, which is required because the objective contains a logarithm of the determinant.

3.2 Entropy-SGD

Entropy-SGD Chaudhari et al. [2019] is an algorithm which biases SGD towards flat regions of the loss landscape by replacing the original objective with a loss they call local entropy

$$F(\theta,\gamma) = \log \int_{\theta' \in \mathbb{R}^n} exp(-f(\theta') - \frac{\gamma}{2}\|\theta - \theta'|_2^2) d\theta' \tag{17}$$

where $f(\theta) \propto \log(p(D,\theta))$. Because local entropy integrates over the networks weights, the loss landscape gets smoothed. This means that flat regions mostly stay the same, while a sharp minimum gets smoothed by the surrounding walls of high loss, resulting in an increase in loss. Therefore, minimal local entropy will be reached at a point which has both low loss in the original landscape and is sufficiently flat, similar to VB in fig. 2. The hyperparameter γ controls how much the loss landscape gets smoothed - A low γ smooths strongly while a high value mostly retains the original landscape.

By reformulating $F(\theta,\gamma)$ (where the last step follows from Jennsen's inequality)

$$\begin{aligned} F(\theta,\gamma) &\propto \log \int_{\theta' \in \mathbb{R}^n} exp(-f(\theta')) * \mathcal{N}(\theta', \theta, \frac{1}{\gamma}) d\theta' && (18)\\ &= \log E_{\mathcal{N}(\theta', \theta, \frac{1}{\gamma})}[exp(-f(\theta'))] && (19)\\ &\geq E_{\mathcal{N}(\theta', \theta, \frac{1}{\gamma})}[-log(p(D,\theta'))] && (20)\\ & && (21) \end{aligned}$$

, we can see that local entropy as an lower bound to the expected log-joint of the ELBO in eq. (3), with variational distribution $q(\theta) = \mathcal{N}(\theta', \theta, \frac{1}{\gamma})$. Note that in contrast to VB, the variance estimate and therefore the entropy $H(q)$ is fixed, which is why Entropy-SGD converges towards a flat minimum by using the expectation over the model parameters without explicitly penalizing low entropy. Due to the integral over the model parameters in the expectation, Entropy-SGD runs into similar computational problems to VB. The authors tackle this by employing Langevin Dynamics to approximate the expectation. This results in a nested loop, where in the outer loop the weights get updated as usual, and the inner loop estimates the expected gradient by using Langevin Dynamics.

3.3 Stochastic Weight Averaging

Stochastic weight averaging Izmailov et al. [2018] is a popular technique for improving generalization in deep learning. The main idea is to average the networks weights at different points. More specifically, if θ_{SWA} is the previous average over n_{models}, new snapshots of the current weights θ get added by

$$\theta_{\text{SWA}} = \frac{\theta_{\text{SWA}} n_{\text{models}} + \theta}{n_{\text{models}+1}}. \tag{22}$$

In order to ensure these snapshots differ from each other enough, the averaging is used in combination with a high learning rate. The authors propose either using a constant high or a cyclic learning rate. In the case of the constant learning rate, the model gets trained as usual and afterwards the learning rate is set to a constant high value while taking snapshots at every epoch. When using a cyclic learning rate on the other hand, snapshots get taken at the minimum learning rate within each cycle.

Similar to Entropy-SGD, SWA does not directly include entropy into the loss function, but converges towards a flat minimum. They show this convergence behaviour by moving in random directions from the final weight estimate and observe that the test error increases substiantially faster for SGD than for SWA, the further they move. SWA improves generalization, which they also attribute to the flatness. In combination to VB however, SWA is much easier to implement, train and requires less computational cost. If a pretrained network is given, SWA can start from that solution, otherwise the authors propose to just train for a reduced number of epochs before applying SWA which still yields comparable performance. SWA only adds computational cost for the averaging step, which is done at most once per epoch and is easy to implement in common deep learning frameworks. The memory requirement is slightly higher due to the θ_{SWA} beeing stored in addition to the current weights, however this resolves after training since we just need θ_{SWA}, which also means testing cost is the same as for MAP.

4 Experiments

4.1 2D toy dataset

We use a 2 dimensional toy dataset from Hernández-Lobato and Adams [2015] with 50 datapoints and a 2 layer MLP with 20 hidden units and *tanh* activation function. We compute the full batch gradient and take 2000 steps using SGD with momentum and a learning rate of $1e-4$.

Figure 1 compares the training trajectory of the first two principal components for different methods with same initializations. The contourlines and colours visualize the negative log-joint. The MAP estimate converges to the local minimum in the middle of the plot. Although VB initally walks in the same direction as MAP, it deviates and eventually converges outside the local minimum, however retaining low loss. In contrast, the trajectories from SWA and VB both lead through the MAP estimate, but then continue to move outside the minimum similar to VB. Only Entropy-SGD takes a different direction, resulting in an estimate with high loss.

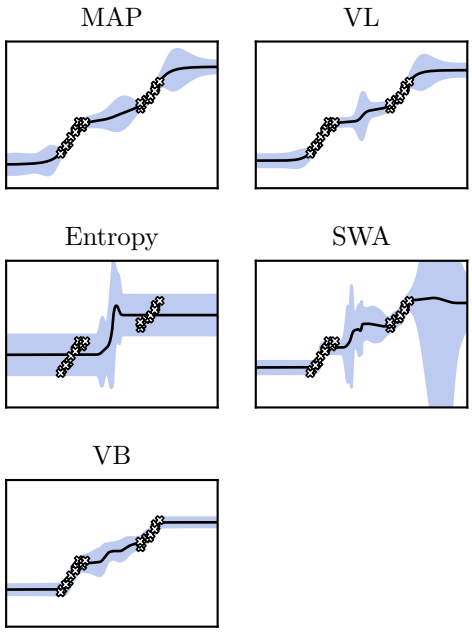

Figure 3: Predictive distributions given by LA on 2D toy dataset for different models. Crosses show training datapoints, black line mean prediction μ_{pred} and blue lines uncertainty given by $\mu_{\text{pred}} \pm 1.96 \cdot \sigma_{\text{pred}}$.

Figure 3 shows the resulting predictive uncertainty when combining these point estimates with LA. MAP shows small uncertainty around and high uncertainty away from data. VB and VL both shows similar behaviour, however with smaller uncertainty than MAP. Entropy-SGD underfits which results in high uncertainty estimates.

SWA also has higher uncertainty estimates which are less smooth, however it fits the data better compared to Entropy-SGD.

4.2 MNIST

We train a small CNN with 3 blocks of Convolutional - MaxPool - RELU with increasing number of channels for 20 epochs on MNIST. We use SGD with an initial learning rate of $1e{-}2$ and decay the learning rate by a factor of 10 every 5 epochs. In total, the model contains 8778 trainable parameters.

Figure 4 shows a histogram of diagonal Hessian eigenvalues of these parameters on the train set. The eigenvalues of the MAP estimate have a long tail towards higher values, which indicates a sharp loss region. In comparison, VB has more smaller and less larger eigenvalues which are usually found within flat loss regions. This behaviour is reflected by SWA and Entropy-SGD. VL in contrast obtains a histogram more similar to VB, having even less small and a similar tail towards higher eigenvalues.

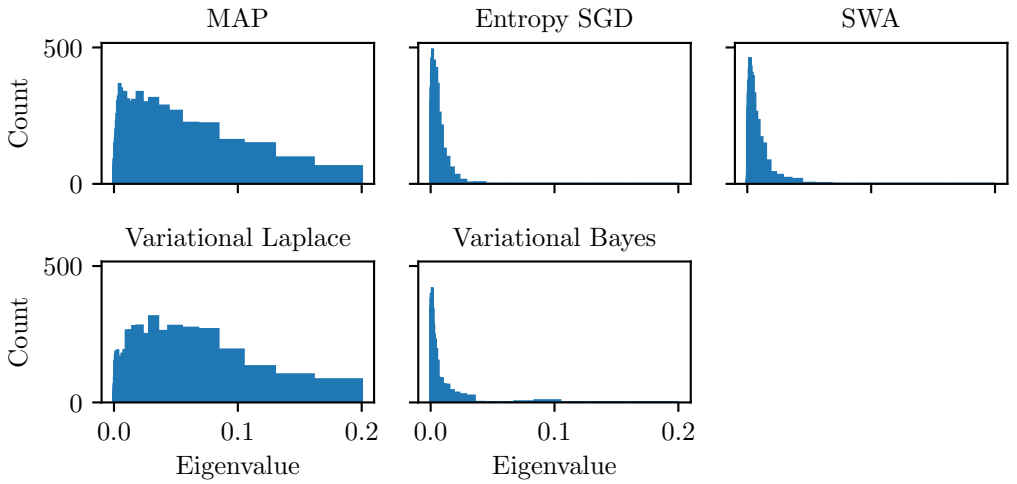

Figure 4: Histogram of diagonal hessian eigenvalues of the training negative log-likelihood for different models trained on MNIST.

4.3 CIFAR-10

We use CIFAR-10 to investigate model performance under various metrics. Here, we train a WideResNet for 100 epochs, using SGD with nesterov momentum and a cosine learning rate decay from $1e{-}1$ to 0. Given the size of the network, we fit the LA only to the last-layer, which is more cost effective but still gives good results Kristiadi et al. [2020].

Table 1: Results of the LA applied onto the methods for different metrics and algorithms on CIFAR-10 test set in percent.

Method	Accuracy	Confidence	ECE	AURC
MAP	91.60	91.00	0.85	1.05
SWA	91.52	90.86	0.76	1.03
VL	82.76	87.52	4.76	3.55
VB	90.3	89.75	0.77	1.20
Entropy	57.40	79.88	22.48	14.33

Table 1 shows the results of the LA applied onto the methods for different metrics. MAP reaches the highest accuracy with 91.60%. VB and LA reach similar accuracy while VL only reaches 82.76% and Entropy-SGD fails to learn well. All models are slightly underconfident by having lower confidence than accuracy, again VL and Entropy-SGD beeing the exception. We use expected calibration error (ECE) and area under the risk coverage (AURC) as a metric to evaluate predictive uncertainty. For ECE, SWA and VB reach lower ECE than MAP. VL and Entropy-SGD have a much higher ECE,

probably due to lower accuracy and overconfidence. The AURC of SWA is comparable to MAP, the AURC of VB is slightly higher and of VL, Entropy-SGD significantly higher.

We measure the confidence for different out-of-distribution datasets. First, we pertubate CIFAR-10 by adding gaussian noise while varying the standard deviation. In addition, we use CIFAR-100 as an OOD dataset. Table 2 shows the results. First, we note that all models except Entropy-SGD (probably due to bad general performance) decrease the confidence for every OOD dataset compared to the baseline CIFAR-10 dataset. The higher the gaussian noise, the more the confidence decreases. For a noise of 0.1, the confidence is similar for all models except VB with a lower estimate. SWA and VB show lower confidence than MAP and VL for a noise of 1. On CIFAR-100, the estimate are again similar. Meanwhile the accuracy drops to between 0.8 and 0.85 for a noise of 0.1 and below 0.2 for noise of 1.

Table 2: Confidence of the LA applied onto the methods for different OOD datasets in percent. *Noise* denotes CIFAR-10 with additive noise drawn from $\mathcal{N}(0, \sigma^2)$, σ given in brackets.

Method	Baseline	Noise (0.1)	Noise (1)	CIFAR-100
MAP	91.00	85	64.41	63.43
SWA	90.86	84.96	58.16	64.12
VL	87.52	84.58	73.99	66.29
VB	89.75	82.20	60.54	63.63
Entropy	79.88	79.72	82.10	70.79

5 Discussion

Although the LA has shown to be good and inexpensive, it is claimed to still fall short of VB. Recent work has shown that Natural Gradient VB updates the variance estimate with the Hessian like LA, which however is not true for the mean parameter. Therefore we argue that when biasing MAP towards VB, adding LA will results in similar behaviour to VB. We further argue that we only need to retain properties of VB instead of the exact estimate, and propose that the main missing property in MAP is high entropy, resulting in convergence towards a flat minimum. We verify this in Figure 4 by showing that for a small CNN on MNIST, VB reaches a flatter minimum than MAP which is also in line with other works like Khan and Rue [2021]. Since there are various ways in which a flat hessian can be achieved, we pick 3 methods: VL is mathematically closest to VB by plugging the LA into the ELBO, however computationally expensive. By smoothing the loss function, Entropy-SGD biases towards flat regions and can be seen as equal to the expected log-likelihood from VB, but with fixed entropy. Compared to the other two, SWA is an inexpensive method without theoretical connection but has also shown to converge to a flat minimum.

In the Hessian experiment on MNIST in section 4.2, both SWA and Entropy-SGD show similar properties like VB. Compared to MAP, they also obtain a small eigenvalues, showing that they indeed converge to a flat minimum. VL on the other hand reflects the MAP estimate with larger Hessian eigenvalues. The reason is not clear and requires further investigation, the usage of the GGN as an approximation of the Hessian could be a possible explanation. For a small 2D Toy dataset, we visualize the training trajectories to get an overview on how the methods affect training. VL and SWA show that while they pass through the minimum obtained by MAP, they eventually converge outside. The behaviour is similar to VB, which also does not converge at the local minimum. We further compare how this affects the subsequent LA. Here, LA shows similar uncertainty estimates on VL and MAP, while the estimate on SWA is much higher. One might conclude this is because SWA converges further outside the minimum, however VB has similar convergence while the uncertainty stays small.

We use CIFAR-10 to investigate performance on a more challenging, real-world dataset. In general, the results are similar for all algorithms except VL and Entropy-SGD, which perform worse. LA on SWA and VB have lower ECE than LA on MAP, but higher or similar AURC. In order to get more clear results, this requires further investigation, possibly on a different dataset. However it is interesting that VB does not show major improvements upon the baseline LA on MAP. For OOD detection, we choose to add noise to CIFAR-10 and use CIFAR-100. All algorithms have a similar drop in confidence for gaussian noise of 0.1, which seems justified because the accuracy drops to a similar level. For higher noise where accuracy drops below 0.2, all algorithms are overconfident.

Here LA on SWA and VB have a lower confidence than LA on MAP, indicating a better but still imperfect fit. On CIFAR-100, all algorithms are again close but overconfident.

In summary, the results draw an unclear picture on our hypothesis. On the one hand, the eigenvalue investigation on MNIST and the training trajectories show that there is a difference between MAP and VB estimates, which some of the investigated algorithms seem to be able close. On the other hand, convergence away from the local minimum of MAP on the 2D Toy dataset does not generally imply higher predictive uncertainty, as seen for VB. Therefore further investigation needs to go into the connection between loss landscape and uncertainty. While there are some positive results on CIFAR-10, in general the difference between algorithms is small. This questions if VB is superior to LA on the plain MAP estimate for more real-world applications, which therefore requires further investigation as well. In general, our results show the importance of better understanding the difference between MAP and VB at convergence. Finding algorithms that are able to close the gap will create new alternatives to VB, since LA provide an estimate at convergence with good theoretical connection to VB. If this can be done in a more cost effective or easy to use way than VB, this might help to scale bayesian deep learning to larger problems and make it more attractive for practical applications.

References

C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, 2015.

P. Chaudhari, A. Choromanska, S. Soatto, Y. LeCun, C. Baldassi, C. Borgs, J. Chayes, L. Sagun, and R. Zecchina. Entropy-SGD: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019.

J. Daunizeau. The Variational Laplace approach to approximate Bayesian inference. *arXiv preprint arXiv:1703.02089*, 2017.

E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, and P. Hennig. Laplace Redux-Effortless Bayesian deep learning. *Advances in Neural Information Processing Systems*, 2021.

A. Graves. Practical Variational inference for neural networks. *Advances in neural information processing systems*, 2011.

J. M. Hernández-Lobato and R. Adams. Probabilistic backpropagation for scalable learning of Bayesian neural networks. In *International conference on machine learning*, 2015.

S. Hochreiter and J. Schmidhuber. Flat minima. *Neural computation*, 1997.

P. Izmailov, D. Podoprikhin, T. Garipov, D. Vetrov, and A. G. Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

A. Kendall and Y. Gal. What uncertainties do we need in Bayesian deep learning for computer vision? *Advances in neural information processing systems*, 2017.

N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. *arXiv preprint arXiv:1609.04836*, 2016.

M. Khan, D. Nielsen, V. Tangkaratt, W. Lin, Y. Gal, and A. Srivastava. Fast and scalable Bayesian deep learning by weight-perturbation in Adam. In *International Conference on Machine Learning*, 2018.

M. E. Khan and H. Rue. The Bayesian learning rule. *arXiv preprint arXiv:2107.04562*, 2021.

A. Kristiadi, M. Hein, and P. Hennig. Being Bayesian, even just a bit, fixes overconfidence in Relu networks. In *International conference on machine learning*, 2020.

D. J. C. Mackay. *Bayesian methods for adaptive models*. PhD thesis, 1992.

K. P. Murphy. *Probabilistic machine learning: An introduction*. MIT press, 2022.

R. M. Neal. *Bayesian learning for neural networks*. 2012.

H. Ritter, A. Botev, and D. Barber. A scalable Laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, 2018.

L. Sagun, L. Bottou, and Y. LeCun. Eigenvalues of the hessian in deep learning: Singularity and beyond. *arXiv preprint arXiv:1611.07476*, 2016.

N. N. Schraudolph. Fast curvature matrix-vector products for second-order gradient descent. *Neural computation*, 2002.

F. Wenzel, K. Roth, B. S. Veeling, J. Świątkowski, L. Tran, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, and S. Nowozin. How good is the bayes posterior in deep neural networks really? *arXiv preprint arXiv:2002.02405*, 2020.