PML 6. Bayesian Neural Networks

Probabilistic Machine Learning Reading Group

Daniel Corrales

January 21, 2026

Instituto de Ciencias Matemáticas (ICMAT)

Contents

Contents

Contents

Predictive Models

Problem: Learn to predict outputs y from inputs x using some function f.
Method: Estimate function f from a labeled training set $\mathcal{D} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n) : n = 1 : N\}$, for $\boldsymbol{x}_n \in \mathcal{X} \subseteq \mathbb{R}^D$ and $\boldsymbol{y}_n \in \mathcal{Y} \subseteq \mathbb{R}^C$.
Uncertainty: We may model our uncertainty about the correct output given input using a *conditional probability model* $p(\boldsymbol{y}|f(\boldsymbol{x}))$.
Cases: *Parametric* vs Nonparametric, Deterministic vs *Bayesian*

Predictive parametric models i

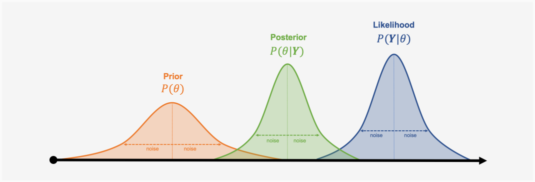
In supervised parametric learning we use a labeled dataset $\mathcal{D} = \{(\boldsymbol{x}_n, \boldsymbol{y}_n) : n = 1 : N\}$ and a probabilistic model $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta})$.

Deterministic: Prediction is done by calculating $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}^*)$, where $\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta} \in \boldsymbol{\theta}} \frac{1}{N} \sum_{i=1}^{N} \ell_n(\boldsymbol{\theta}) + \lambda C(\boldsymbol{\theta})$, where $\ell_n(\boldsymbol{\theta})$ is a *loss function* which measures goodness of fit, e.g. log-loss $\ell_n(\boldsymbol{\theta}) = -\log p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta})$; $C(\boldsymbol{\theta})$ is a *regularization factor* and $\lambda \in \mathbb{R}$.

If we use the log-loss, and define $C(\boldsymbol{\theta}) = -\log \pi_0(\boldsymbol{\theta})$, where $\pi_0(\boldsymbol{\theta})$ is some prior distribution, and we use $\lambda = 1$, we recover the Maximum a Posteriori (MAP) estimate.

Predictive parametric models ii

Bayesian: Instead of predicting using just one value of $\boldsymbol{\theta}$, compute the posterior predictive distribution using Bayesian model averaging $p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta})p(\boldsymbol{\theta}|D)d\boldsymbol{\theta}$. We average over different possible models (possible values of $\boldsymbol{\theta}$), weighted depending on parameter posterior probability $p(\boldsymbol{\theta}|D)$, where $p(\boldsymbol{\theta}|D) \propto p(\boldsymbol{\theta})p(D|\boldsymbol{\theta})$, that is, the posterior is proportional to the likelihood times the prior.



Motivations of using Bayesian predictive models

$$p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|D) d\boldsymbol{\theta}.$$

$$p(\boldsymbol{\theta}|D) = \frac{p(\boldsymbol{\theta}) p(D|\boldsymbol{\theta})}{\int p(\boldsymbol{\theta}) p(D|\boldsymbol{\theta}) d\boldsymbol{\theta}}$$

Pros:

- We define a distribution over parameters, which lets us represent uncertainty more accurately.
- Considering different models together can lead to improved accuracy

Cons:

- Integration has no closed form in general
- Problem is usually intractable

Ingredients for doing Bayesian inference

1. **Prior** $p(\boldsymbol{\theta})$: Reflects our beliefs about parameters before seeing the data.
2. **Likelihood** $p(D|\boldsymbol{\theta})$: Represents how well the data supports a specific hypothesis.
3. **Posterior inference** $p(\boldsymbol{\theta}|D)$: What we believe after seen the data
4. **Evidence / marginal likelihood** $p(D)$: Total probability of the data under all possible hypothesis
5. **Predictive** $p(\boldsymbol{y}|\boldsymbol{x}, D)$: Marginalize parameters to predict

Some examples of Bayesian predictive parametric models i

Linear regression with known σ: $y = \boldsymbol{\theta}^T \boldsymbol{x} + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2)$

Likelihood : $p(y|\boldsymbol{X}, \boldsymbol{\theta}) = \prod_{n=1}^{N} \mathcal{N}(y_n|\boldsymbol{\theta}^T \boldsymbol{x}_n, \sigma^2)$

Prior : $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|0, \sigma_0^2)$

Posterior : $p(\boldsymbol{\theta}|\boldsymbol{X}, y, \sigma^2) = \mathcal{N}\left(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma})\right)$

$$\boldsymbol{\mu} = \frac{1}{\sigma^2}\boldsymbol{\Sigma}\boldsymbol{X}^T y, \qquad \boldsymbol{\Sigma} = \left(\sigma_0^{-1} + \frac{1}{\sigma^2}\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}$$

Predictive : $p(\boldsymbol{y}^*|\boldsymbol{x}^*, D) = \mathcal{N}\left(\boldsymbol{y}^*|\boldsymbol{\mu}^T\boldsymbol{x}^*, \sigma_0^2 + (\boldsymbol{x}^*)^T\boldsymbol{\Sigma}\boldsymbol{x}^*\right)$

Some examples of Bayesian predictive parametric models ii

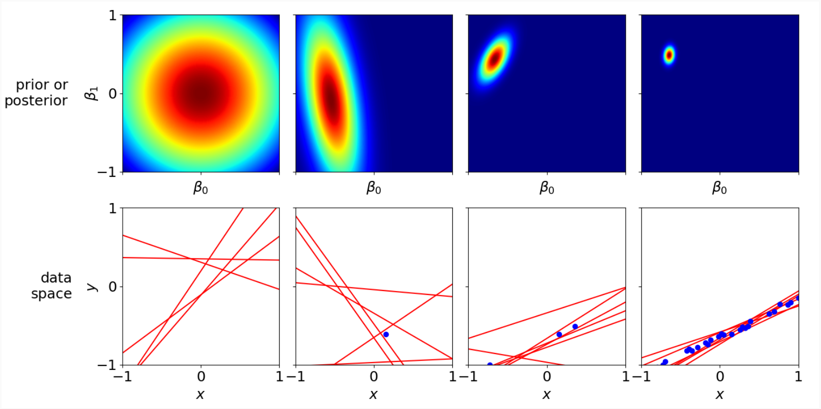


Figure 1: Source

Contents

Contents

Deep Neural Networks + Bayesian Model Averaging

What if we need more complex modelling?

Idea: Leverage deep neural networks (DNNs) for modeling the conditional distribution.

$$p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}) = f_{NN}(\boldsymbol{\theta}; \boldsymbol{x})$$

But DNNs are large flexible models which can represent many functions, corresponding to different parameter settings (**underspecification**, [D'A+22]), which fit the training data well, yet generalize in different ways. We can benefit from Bayesian model averaging for improved accuracy and uncertainty quantification.

$$p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|D) d\boldsymbol{\theta}$$

Classical vs Bayesian approach to DL

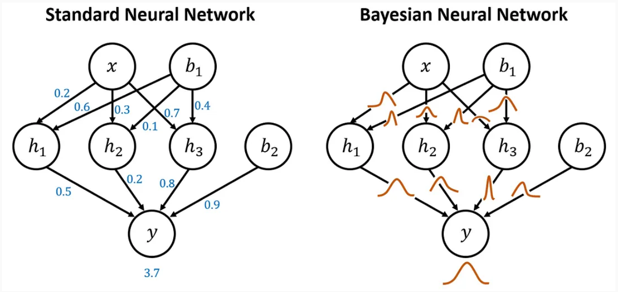


Figure 2: . Standard NN vs Bayesian NN

Deep Neural Networks + Bayesian Model Averaging

BNNs inherit the ingredients for Bayesian inference:

- Prior parameter distribution $p(\boldsymbol{\theta})$
- Likelihood distribution $p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta})$
- Posterior distribution $p(\boldsymbol{\theta}|D)$
- Predictive posterior distribution $p(\boldsymbol{y}|\boldsymbol{x}, D)$

Our model becomes more general, but that comes at a cost. We lose convexity, as NNs have nonlinear transformations, and there is a dimensionality explosion, which increases expressivity but compromises inference.

$$p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}) p(\boldsymbol{\theta}|D) d\boldsymbol{\theta}.$$

$$p(\boldsymbol{\theta}|D) = \frac{p(\boldsymbol{\theta}) p(D|\boldsymbol{\theta})}{\int p(\boldsymbol{\theta}) p(D|\boldsymbol{\theta}) d\boldsymbol{\theta}}$$

Contents

Predictive posterior

Nearly all approaches in Bayesian deep learning for estimating the integral

$$p(\boldsymbol{y}|\boldsymbol{x}, D) = \int p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta})p(\boldsymbol{\theta}|D)d\boldsymbol{\theta} \tag{1}$$

when it cannot be computed in closed form, involve a *simple Monte Carlo* approximation

$$p(\boldsymbol{y}|\boldsymbol{x}, D) \approx \frac{1}{J}\sum_{j=1}^{J} p(\boldsymbol{y}|\boldsymbol{x}, \boldsymbol{\theta}_j) \qquad \boldsymbol{\theta}_j \sim p(\boldsymbol{\theta}|D) \tag{2}$$

This involves samples from the posterior, which in practice will also be approximate as the true posterior will be too complicated to sample from.

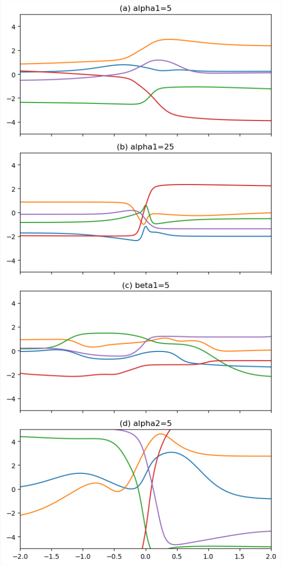
Contents

Priors i

1. **Gaussian priors**
   Consider an MLP with L-1 hidden layers and a linear output $f(\boldsymbol{\theta}; \boldsymbol{x}) = W_L(...\psi(W_1 x + b)) + b_L$. The most common choice is to use a factored Gaussian prior:

$$W_l \sim \mathcal{N}(0, \alpha_l^2 I),\ b_l \sim \mathcal{N}(0, \beta_l^2 I)$$

   Some examples for α_l and β_l include Xavier initialization and LeCun initialization.



Priors ii

2. **Priors in function space**
   Establishing priors over parameters leads to different possible functions (e.g., case of Gaussian priors). However, the function characteristics preferred by the prior are not straightforward to understand.
   Some approaches attempt to design informative priors that lead to desired invariances, locality, independencies or symmetries.

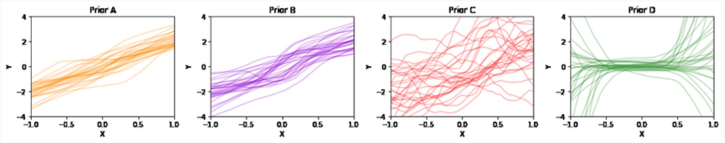


Figure 3: Obtained from [GCO25]

Priors iii

3. **Architectural priors**
   Neural network architecture can also have a large effect on the induced distribution over functions (inductive bias):
   - Convolutional NNs encode prior knowledge about translation equivariance due to their use of convolutions. E.g. a car at the top of an image is the same as a car at the bottom".
   - Recurrent NNs share weights accross time, leading to time-translation equivariance, and have a sequentiality bias (time order).
   - With a suitable architecture, we may get good results using random (untrained) models.

   This field that studies this is called **Neural Architecture Search** [EMH19]

Contents

Posteriors: Laplace approximation i

Laplace approximations [Mac92; Dax+21]: Compute a Gaussian approximation of the posterior centered at the MAP estimate, $\boldsymbol{\theta}_{MAP}$. Note we can write the posterior as

$$p(\boldsymbol{\theta}|D) = \frac{1}{Z} e^{-U(\boldsymbol{\theta})}, \qquad \text{with } U(\boldsymbol{\theta}) = -\log p(\boldsymbol{\theta}, D)$$

Perform a 2nd-order Taylor expansion around the mode $\boldsymbol{\theta}$:

$$U(\boldsymbol{\theta}) \approx U(\hat{\boldsymbol{\theta}}) + (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T g + \frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T H (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})$$

where g is the gradient, H is the Hessian. This results in

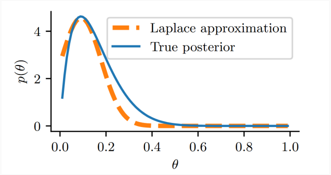
$$\hat{p}(\boldsymbol{\theta}, D) = e^{-U(\boldsymbol{\theta})} \exp\left\{-\frac{1}{2}(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})^T H (\boldsymbol{\theta} - \hat{\boldsymbol{\theta}})\right\}$$

$$\hat{p}(\boldsymbol{\theta}|D) = \mathcal{N}(\boldsymbol{\theta}|\hat{\boldsymbol{\theta}}, H^{-1})$$

Posteriors: Laplace approximation ii

Properties:

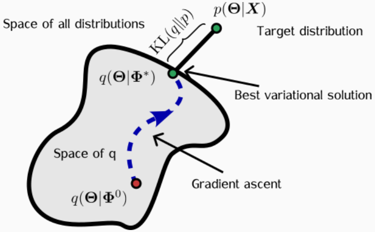
- Simple approximation, leverages $\boldsymbol{\theta}_{MAP}$, obtained through optimization
- Bayesian estimate for any pretrained model. Curvature information is only used after the model has been estimated, not during the optimization process.
- Computing the Hessian can be expensive, fast second-order optimisation methods exist
- Highly local and unimodal.

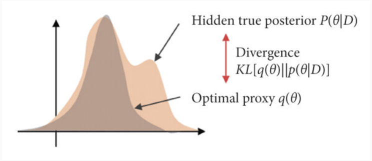


Posteriors: Variational Inference i

Variational Inference [HVC93; Blu+15]: For posterior approximation of $p(\boldsymbol{\theta}|D)$, choose the distribution $q_\psi(\boldsymbol{\theta})$ that minimizes $D_{KL}(q_\psi(\boldsymbol{\theta})||p(\boldsymbol{\theta}|D))$ with respect to ψ.

We often choose a Gaussian approximate posterior $q_\psi(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\mu, \Sigma)$, which lets us use the reparametrization trick to create a low variance estimator of the gradient of the ELBO.


Space of all distributions
$KL(q||p)$
$p(\boldsymbol{\Theta}|\boldsymbol{X})$
Target distribution
$q(\boldsymbol{\Theta}|\boldsymbol{\Phi}^*)$
Best variational solution
Space of q
Gradient ascent
$q(\boldsymbol{\Theta}|\boldsymbol{\Phi}^0)$

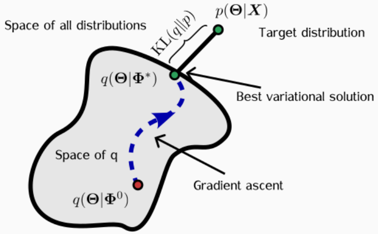

Hidden true posterior $P(\theta|D)$
Divergence
$KL[q(\theta)||p(\theta|D)]$
Optimal proxy $q(\theta)$

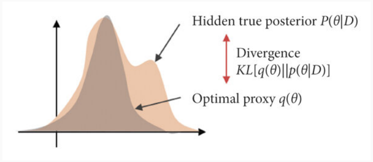
Posteriors: Variational Inference ii

However, these approximations often **underestimate the uncertainty**. Using normalizing flows (Ch. 23) or implicit distributions partly mitigate this underestimations.

Properties:

- Simple method, optimization-based.
- Curvature is used at every step of gradient descent.
- Local and unimodal when Gaussian is used.

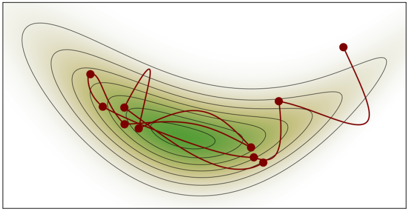




Posteriors: MCMC methods i

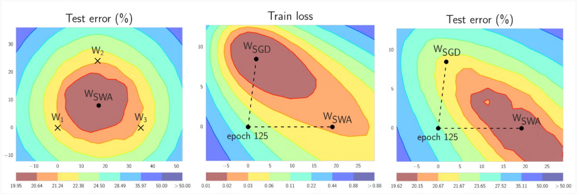
MCMC methods [Nea96]: **Hamiltonian Monte Carlo** is considered the gold standard, as it does not make strong assumptions about the form of the posterior.

However, its use is limited because it requires access to the full training set at each step. Stochastic gradient MCMC methods offer a more scalable alternative as they operate using mini-batches of data (e.g. **SGLD**).



Posteriors: SGD trajectory-based methods i

SGD trajectory-based methods: In **stochastic weight averaging (SWA)**[Izm+18], they noted that SGD solutions, for a fixed learning rate, surround the periphery of points of good generalization. They propose to compute the average the SGD samples, collected after a certain interval, to get $\overline{\boldsymbol{\theta}} = \frac{1}{S}\sum_{s=1}^{S}\boldsymbol{\theta}_s$



Posteriors: SGD trajectory-based methods ii

In [Mad+19], they fit a Gaussian distribution over the set of SGD samples **(SWA-Gaussian)**. Further, **Multi-SWAG** can detect several different modes and approximates the posterior through a mixture of Gaussians.

Properties:

- No additional training overhead compared to standard training
- Better generalization, improves accuracy and calibration.
- Heuristic. Not properly Bayesian.

Posteriors: Deep Ensembles i

Deep Ensembles [LPB17]: Train multiple models with different initializations, then approximate the posterior using an equally weighted mixture of delta functions

$$p(\boldsymbol{\theta}|D) \approx \frac{1}{M} \sum_{m=1}^{M} \delta(\boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_m),$$

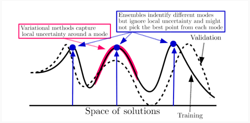
where M is the number of models, and $\hat{\boldsymbol{\theta}}_m$ is the MAP estimate for model m.

It has been argued that deep ensembles can approximate the BNN posterior predictive [WI20].

Posteriors: Deep Ensembles ii

Properties:

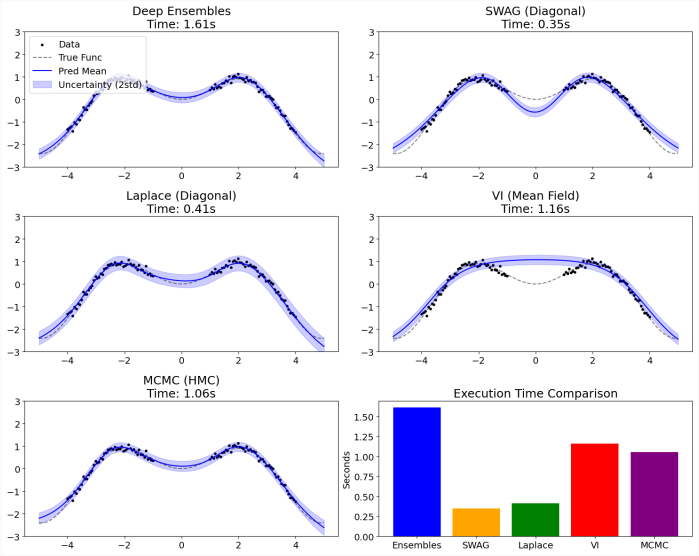
- Highly multi-modal. Parameters in different modes give rise to very different functions.
- Random priors can be combined with bootstrap data sampling.
- Local method (just uses modes)
- High training cost (M times standard training)
- Not a competing approach for doing Bayesian inference, but a compelling mechanism for Bayesian marginalization.


Variational methods capture local uncertainty around a mode
Ensembles indentify different modes but ignore local uncertainty and might not pick the best point from each mode
Validation
Space of solutions
Training

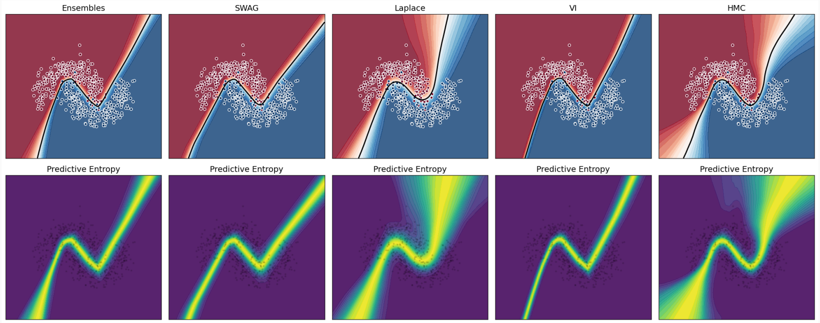
Comparison of BNN Inference Methods

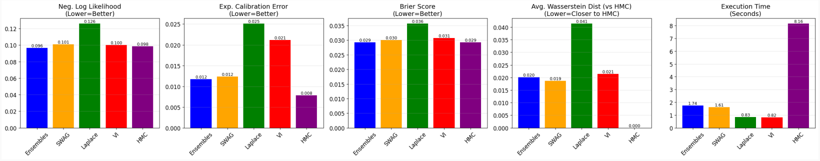
Method	Training	Inference	Scalability	Posterior	Generalization & Quality
MCMC (HMC) *(Theory Only)*	Very High	High	Poor	True (Multimodal)	**Gold Standard** (Perfect calibration)
Variational Inf. *(Bayes by Backprop)*	Medium	Low	High	Unimodal (Gaussian)	**Low / Medium** (Often overconfident)
Laplace Approx. *(Post-hoc)*	Low	Low	High	Unimodal (Gaussian)	**Medium** (Good in-dist, poor OOD)
SGD Trajectory *(SWAG)*	Low	Low	High	Approx. Multimodal	**Good** (Captures flat basins)
Deep Ensembles *(Practical Winner)*	High	Medium	High	Multimodal (Non-param)	**Excellent (SOTA)** (Best OOD robustness)

Comparison of BNN inference methods



Comparison of BNN inference methods





Questions?

Questions?

Contents

Online inference i

An important application of Bayesian inference is in sequential settings, where data arrives in a continuous stream. The posterior is updated recursively, following a simple intuition: today's posterior becomes tomorrow's prior.

$$\log p(\boldsymbol{\theta}|D_{1:t}) \propto \log p(D_t|\boldsymbol{\theta}) + \log p(\boldsymbol{\theta}|D_{1:t-1})$$

A sequential approach for many of the inference methods explained exists, e.g., sequential Laplace or sequential VI. Ideas from Kalman Filters are also relevant in the field.

However, in most of the cases, the effect of the prior and the likelihood is reweighted, to control for how much the model pays attention to new data vs old data.

Online inference ii

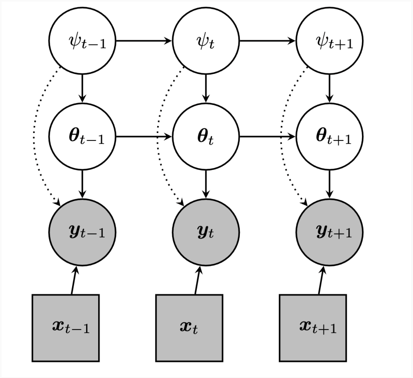


Figure 6: Obtained from [DM+24]

Contents

Generalization in Bayesian DL i

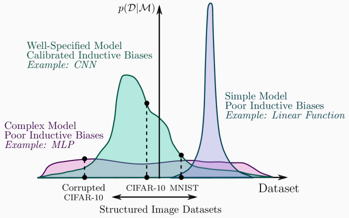
Why being Bayesian can improve predictive accuracy and generalization performance? Again, the key is marginalizing rather than using a single setting of weights. Generalization depends on two properties of the model

1. **Support**
2. **Inductive bias**

The difference between the classical and the Bayesian approach will depend on how sharp the posterior is. In underspecified models, likelihoods are diffuse and do not favour any one setting of parameters. BMA leverages this.

Generalization in Bayesian DL ii

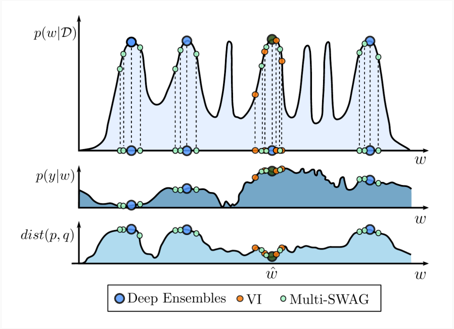
We want the **support** of the model to be large so that we can represent any hypothesis we believe to be possible. We need an **inductive bias** that represents the relative prior probabilities of the hypotheses



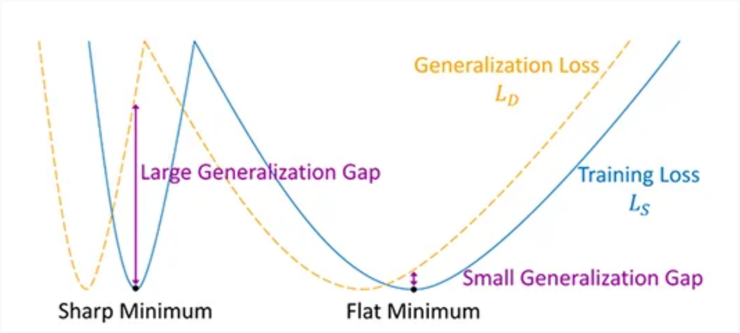
Thus, the flexibility of a model should not be mistaken for the complexity of the model class.

Generalization in Bayesian DL iii

Most Bayesian deep learning methods focus on faithfully approximating a posterior within a single basin of attraction (e.g. using the MAP, fitting a Gaussian). Arguably, **representing multiple basins of attraction in the posterior leads to better generalisation** when marginalizing.



Flat loss


Generalization Loss
L_D
Large Generalization Gap
Training Loss
L_S
Small Generalization Gap
Sharp Minimum
Flat Minimum

Limitations of BNN i

Tempered/cold posterior effect: BNNs give better predictive accuracy if the likelihood, or the full posterior itself, is scaled by some power $T < 1$.

$$p_T(w|D) = \frac{1}{Z(T)} p(D|w)^{1/T} p(w) \tag{3}$$

Performance gains for $T < 1$ point to a potentially resolvable problem with prior, likelihood or inference procedure.

Tempering the posterior distribution helps under model misspecification (which is always the case).

Limitations of BNN ii

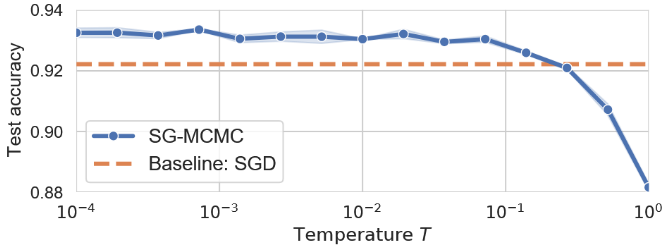


Figure 1. The "**cold posterior**" effect: for a ResNet-20 on CIFAR-10 we can improve the generalization performance significantly by cooling the posterior with a temperature $T \ll 1$, deviating from the Bayes posterior $p(\boldsymbol{\theta}|\mathcal{D}) \propto \exp(-U(\boldsymbol{\theta})/T)$ at $T = 1$.

Limitations of BNN iii

The causes of the cold posterior effect are directly associated with underrepresenting aleatoric uncertainty.

[Wen+20] points to prior misspecification being one of the causes for the cold posterior effect.

[Kap+22] argues that cold posterior in classification is due to using softmax likelihoods, for which there is no modelling of the inherent randomness of the data. They propose a noise Dirichlet model to account for that aleatoric uncertainty.

$$p(y|\mathbf{x},\theta) = \mathcal{N}(y|f(\mathbf{x}),\sigma^2) \quad ; \quad p(y|\mathbf{x},\theta) = \mathsf{Cat}(y|\mathsf{softmax}(f(\mathbf{x})))$$

Final remarks

- Bayesian model averaging is beneficial for increased accuracy and uncertainty quantification in NNs.
- Prior specification and inference approximations are needed. These usually entail assumptions (local, Gaussian, ...)
- Future steps in the field are in the direction of studying model generalization capabilities, representing aleatoric uncertainty, and prior design.

Missing topics in this talk

- Last-layer posterior approximations methods / Partially stochastic networks
- Alternatives to Monte Carlo approximation to the predictive posterior.
- Functional BNNs
- Bayesian model selection and marginal likelihood.
- ...

Questions?

Questions?

Next Session

Beyond the i.i.d. assumption (Ch. 19)

February 4th

Carlos G. Meixide (ICMAT)

References i

[Blu+15] Charles Blundell et al. **"Weight uncertainty in neural network".** In: *International conference on machine learning*. PMLR. 2015, pp. 1613–1622.

[D'A+22] Alexander D'Amour et al. **"Underspecification presents challenges for credibility in modern machine learning".** In: *Journal of Machine Learning Research* 23.226 (2022), pp. 1–61.

[Dax+21] Erik Daxberger et al. **"Laplace redux-effortless bayesian deep learning".** In: *Advances in neural information processing systems* 34 (2021), pp. 20089–20103.

References ii

[DM+24] Gerardo Duran-Martin et al. **“A unifying framework for generalised Bayesian online learning in non-stationary environments”.** In: *arXiv preprint arXiv:2411.10153* (2024).

[EMH19] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. **“Neural architecture search: A survey”.** In: *Journal of Machine Learning Research* 20.55 (2019), pp. 1–21.

References iii

[GCO25] Javad Ghorbanian, Nicholas Casaprima, and Audrey Olivier. **“Empowering approximate Bayesian neural networks with functional priors through anchored ensembling for mechanics surrogate modeling applications”.** In: *Computer Methods in Applied Mechanics and Engineering* 438 (2025), p. 117645.

[HVC93] Geoffrey E Hinton and Drew Van Camp. **“Keeping the neural networks simple by minimizing the description length of the weights”.** In: *Proceedings of the sixth annual conference on Computational learning theory*. 1993, pp. 5–13.

References iv

[Izm+18] Pavel Izmailov et al. **"Averaging weights leads to wider optima and better generalization".** In: *arXiv preprint arXiv:1803.05407* (2018).

[Kap+22] Sanyam Kapoor et al. **"On uncertainty, tempering, and data augmentation in bayesian classification".** In: *Advances in neural information processing systems* 35 (2022), pp. 18211–18225.

[LPB17] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. **"Simple and scalable predictive uncertainty estimation using deep ensembles".** In: *Advances in neural information processing systems* 30 (2017).

References v

[Mac92] David JC MacKay. **“A practical Bayesian framework for backpropagation networks”.** In: *Neural computation* 4.3 (1992), pp. 448–472.

[Mad+19] Wesley J Maddox et al. **“A simple baseline for bayesian uncertainty in deep learning”.** In: *Advances in neural information processing systems* 32 (2019).

[Nea96] Radford M Neal. ***Bayesian learning for neural networks.*** Springer, 1996.

[Wen+20] Florian Wenzel et al. **“How good is the bayes posterior in deep neural networks really?”** In: *arXiv preprint arXiv:2002.02405* (2020).

References vi

[WI20] Andrew G Wilson and Pavel Izmailov. **“Bayesian deep learning and a probabilistic perspective of generalization”.** In: *Advances in neural information processing systems* 33 (2020), pp. 4697–4708.