Qualcomm Amsterdam

Imperial College London

Bayesian Model Selection
in Deep Learning

Mark van der Wilk

Department of Computing
Imperial College London

@markvanderwilk
m.vdwilk@imperial.ac.uk

Feb 24, 2021

Outline

Goal: Towards automatic model selection in deep learning.

Outline

Goal: Towards automatic model selection in deep learning.

Talk outline:

1. The promises of Bayesian Model Selection

Outline

Goal: Towards automatic model selection in deep learning.

Talk outline:

1. The promises of Bayesian Model Selection
2. Difficulties with Bayesian Inference in Deep Learning

Outline

Goal: Towards automatic model selection in deep learning.

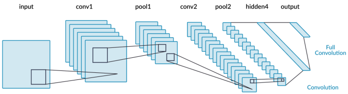
Talk outline:

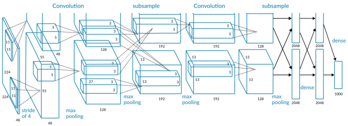
1. The promises of Bayesian Model Selection
2. Difficulties with Bayesian Inference in Deep Learning
3. Other approaches: Ensembles and Architecture Search

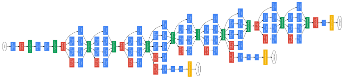
Model Selection

Every time we train a NN we need to decide on hyperparameters:

- How many layers? How many units in a layer?
- What layer structure? Convolutional? Skip connections?
- Data augmentation parameters?







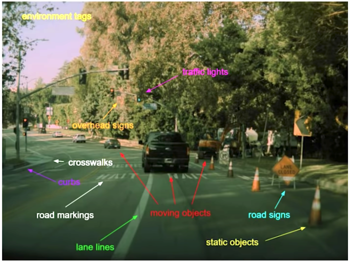
Model Selection

Every time we train a NN we need to decide on hyperparameters:

- How many layers? How many units in a layer?
- What layer structure? Convolutional? Skip connections?
- Data augmentation parameters?

As architectures get more complex, so does design! E.g. multitask.

- Which layers to share?
- What kind of task-specific layers?
- How much capacity to assign to each task?



[Karpathy, ICML 2019]

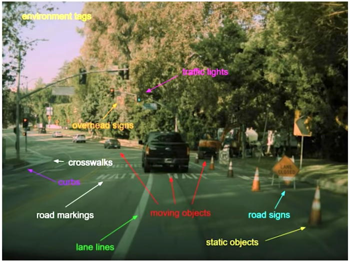
Model Selection

Every time we train a NN we need to decide on hyperparameters:

- How many layers? How many units in a layer?
- What layer structure? Convolutional? Skip connections?
- Data augmentation parameters?

As architectures get more complex, so does design! E.g. multitask.

- Which layers to share?
- What kind of task-specific layers?
- How much capacity to assign to each task?



[Karpathy, ICML 2019]

Main tool is **crossvalidation**.

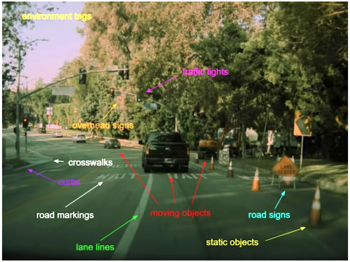
Model Selection

Every time we train a NN we need to decide on hyperparameters:

- How many layers? How many units in a layer?
- What layer structure? Convolutional? Skip connections?
- Data augmentation parameters?

As architectures get more complex, so does design! E.g. multitask.

- Which layers to share?
- What kind of task-specific layers?
- How much capacity to assign to each task?



[Karpathy, ICML 2019]

Main tool is **crossvalidation**.

Goal: Make it as easy as learning weights.

Overview

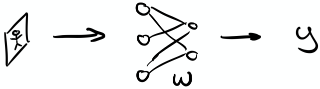
The Promises of Bayesian Model Selection

Difficulties with Bayesian Inference in Deep Learning

Ensembles and Architecture Search

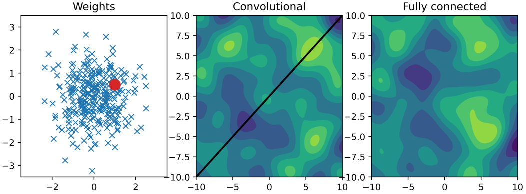
Conclusion

Bayesian Inference



$$f_{\mathbf{w}} : \mathbb{R}^D \to \mathbb{R}^C \qquad (1)$$

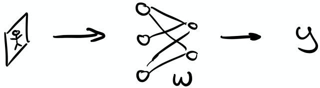
$$\mathbf{x} \mapsto f_{\mathbf{w}}(\mathbf{x}) \qquad (2)$$



- A prior on parameters leads to a prior on functions
- Architectural **hyperparameters** influence prior on functions
- BDL focusses mostly on uncertainty in the function:

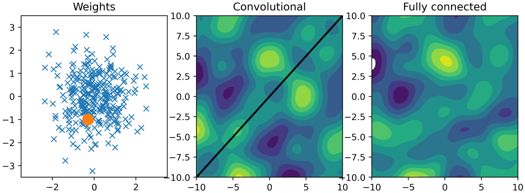
$$p(f|\mathbf{y}, \theta) = \frac{p(\mathbf{y}|f)p(f|\theta)}{p(\mathbf{y}|\theta)} \qquad (3)$$

Bayesian Inference



$$f_{\mathbf{w}} : \mathbb{R}^D \to \mathbb{R}^C \quad (1)$$

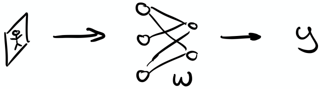
$$\mathbf{x} \mapsto f_{\mathbf{w}}(\mathbf{x}) \quad (2)$$



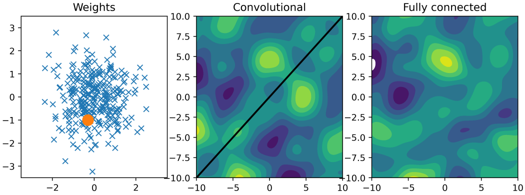
- A prior on parameters leads to a prior on functions
- Architectural **hyperparameters** influence prior on functions
- BDL focusses mostly on uncertainty in the function:

$$p(f|\mathbf{y},\theta) = \frac{p(\mathbf{y}|f)p(f|\theta)}{p(\mathbf{y}|\theta)} \quad (3)$$

Bayesian Inference



$$f_{\mathbf{w}} : \mathbb{R}^D \to \mathbb{R}^C \quad (1)$$
$$\mathbf{x} \mapsto f_{\mathbf{w}}(\mathbf{x}) \quad (2)$$



- A prior on parameters leads to a prior on functions
- Architectural **hyperparameters** influence prior on functions
- BDL focusses mostly on uncertainty in the function:

$$p(f|\mathbf{y}, \theta) = \frac{p(\mathbf{y}|f)p(f|\theta)}{p(\mathbf{y}|\theta)} \qquad (3)$$

But we want to determine the hyperparameters θ too!

Bayesian Model Selection

Bayes tells us: Just find the posterior over all your unknowns!

$$p(f, \theta|\mathbf{y}) = \frac{p(\mathbf{y}|f)p(f|\theta)p(\theta)}{p(\mathbf{y})} \tag{4}$$

Bayesian Model Selection

Bayes tells us: Just find the posterior over all your unknowns!

$$p(f, \theta|\mathbf{y}) = \frac{p(\mathbf{y}|f)p(f|\theta)p(\theta)}{p(\mathbf{y})} = \underbrace{\frac{p(\mathbf{y}|f)p(f|\theta)}{p(\mathbf{y}|\theta)}}_{\text{usual posterior}} \underbrace{\frac{p(\mathbf{y}|\theta)p(\theta)}{p(\mathbf{y})}}_{\text{hyper posterior}} \tag{4}$$

Bayesian Model Selection

Bayes tells us: Just find the posterior over all your unknowns!

$$p(f, \theta|\mathbf{y}) = \frac{p(\mathbf{y}|f)p(f|\theta)p(\theta)}{p(\mathbf{y})} = \underbrace{\frac{p(\mathbf{y}|f)p(f|\theta)}{p(\mathbf{y}|\theta)}}_{\text{usual posterior}} \underbrace{\frac{p(\mathbf{y}|\theta)p(\theta)}{p(\mathbf{y})}}_{\text{hyper posterior}} \tag{4}$$

- Posterior over functions is unchanged!
- Posterior over hyperparams requires **marginal likelihood**:

$$p(\mathbf{y}|\theta) = \int p(\mathbf{y} \mid f)p(f \mid \theta)\mathrm{d}\theta \tag{5}$$

Bayesian Model Selection

Bayes tells us: Just find the posterior over all your unknowns!

$$p(f,\theta|\mathbf{y}) = \frac{p(\mathbf{y}|f)p(f|\theta)p(\theta)}{p(\mathbf{y})} = \underbrace{\frac{p(\mathbf{y}|f)p(f|\theta)}{p(\mathbf{y}|\theta)}}_{\text{usual posterior}} \underbrace{\frac{p(\mathbf{y}|\theta)p(\theta)}{p(\mathbf{y})}}_{\text{hyper posterior}} \tag{4}$$

- Posterior over functions is unchanged!
- Posterior over hyperparams requires **marginal likelihood**:

$$p(\mathbf{y}|\theta) = \int p(\mathbf{y}\,|\,f)p(f\,|\,\theta)\mathrm{d}\theta \tag{5}$$

Bayesian model selection is commonly done by ML-II (Berger, 1985):

$$\theta^* = \operatorname*{argmax}_\theta \log p(\mathbf{y}\,|\,\theta)\,, \qquad \text{predict using } p(f|\mathbf{y},\theta^*) \tag{6}$$

Bayesian Model Selection

Bayes tells us: Just find the posterior over all your unknowns!

$$p(f,\theta|\mathbf{y}) = \frac{p(\mathbf{y}|f)p(f|\theta)p(\theta)}{p(\mathbf{y})} = \underbrace{\frac{p(\mathbf{y}|f)p(f|\theta)}{p(\mathbf{y}|\theta)}}_{\text{usual posterior}} \underbrace{\frac{p(\mathbf{y}|\theta)p(\theta)}{p(\mathbf{y})}}_{\text{hyper posterior}} \tag{4}$$

- Posterior over functions is unchanged!
- Posterior over hyperparams requires **marginal likelihood**:

$$p(\mathbf{y}|\theta) = \int p(\mathbf{y}\,|\,f)p(f\,|\,\theta)\mathrm{d}\theta \tag{5}$$

Bayesian model selection is commonly done by ML-II (Berger, 1985):

$$\theta^* = \operatorname*{argmax}_\theta \log p(\mathbf{y}\,|\,\theta)\,, \qquad \text{predict using } p(f|\mathbf{y},\theta^*) \tag{6}$$

Gradient-based optimisation is **super convenient**!

Bayesian Model Selection

Bayes tells us: Just find the posterior over all your unknowns!

$$p(f,\theta|\mathbf{y}) = \frac{p(\mathbf{y}|f)p(f|\theta)p(\theta)}{p(\mathbf{y})} = \underbrace{\frac{p(\mathbf{y}|f)p(f|\theta)}{p(\mathbf{y}|\theta)}}_{\text{usual posterior}} \underbrace{\frac{p(\mathbf{y}|\theta)p(\theta)}{p(\mathbf{y})}}_{\text{hyper posterior}} \tag{4}$$

- Posterior over functions is unchanged!
- Posterior over hyperparams requires **marginal likelihood**:

$$p(\mathbf{y}|\theta) = \int p(\mathbf{y}\,|\,f)p(f\,|\,\theta)\mathrm{d}\theta \tag{5}$$

Bayesian model selection is commonly done by ML-II (Berger, 1985):

$$\theta^* = \operatorname*{argmax}_\theta \log p(\mathbf{y}\,|\,\theta)\,, \qquad \text{predict using } p(f|\mathbf{y},\theta^*) \tag{6}$$

Gradient-based optimisation is **super convenient**!

... if we can compute $p(\mathbf{y}\,|\,\theta)$

Variational Bayesian Model Selection

Bayes tells us what to do, but not how to do it. Variational inference actually does it, and gives us

- An approximate posterior
- An estimate of the marginal likelihood! (lower bound)

Variational Bayesian Model Selection

Bayes tells us what to do, but not how to do it. Variational inference actually does it, and gives us

- An approximate posterior
- An estimate of the marginal likelihood! (lower bound)

$$
\begin{aligned}
\log p(\mathbf{y} \mid \theta) &= \mathcal{L}(\phi, \theta) + \mathrm{KL}\big[q_\phi(f) || p(f|\mathbf{y}, \theta)\big] \\
&\geqslant \mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q_\phi(f(\mathbf{x}_n))}[\log p(y_n|f(\mathbf{x}_n))] - \mathrm{KL}\big[q_\phi(f) || p(f|\theta)\big]
\end{aligned}
$$

Variational Bayesian Model Selection

Bayes tells us what to do, but not how to do it. Variational inference actually does it, and gives us

- An approximate posterior
- An estimate of the marginal likelihood! (lower bound)

$$\log p(\mathbf{y} \mid \theta) = \mathcal{L}(\phi, \theta) + \mathrm{KL}\big[q_\phi(f) || p(f|\mathbf{y}, \theta)\big]$$
$$\geqslant \mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q_\phi(f(\mathbf{x}_n))}[\log p(y_n|f(\mathbf{x}_n))] - \mathrm{KL}\big[q_\phi(f) || p(f|\theta)\big]$$

- Find posterior and hyperparameters simultaneously by

$$\operatorname*{argmax}_{\phi,\theta} \mathcal{L}(\phi, \theta) \tag{7}$$

Variational Bayesian Model Selection

Bayes tells us what to do, but not how to do it. Variational inference actually does it, and gives us

- An approximate posterior
- An estimate of the marginal likelihood! (lower bound)

$$
\begin{aligned}
\log p(\mathbf{y} \mid \theta) &= \mathcal{L}(\phi, \theta) + \mathrm{KL}\big[q_\phi(f) || p(f | \mathbf{y}, \theta)\big] \\
&\geqslant \mathcal{L} = \sum_{n=1}^{N} \mathbb{E}_{q_\phi(f(\mathbf{x}_n))}[\log p(y_n | f(\mathbf{x}_n))] - \mathrm{KL}\big[q_\phi(f) || p(f | \theta)\big]
\end{aligned}
$$

- Find posterior and hyperparameters simultaneously by

$$
\operatorname*{argmax}_{\phi,\theta} \mathcal{L}(\phi, \theta) \tag{7}
$$

- Quality of posterior is linked to the accuracy of lower bound!

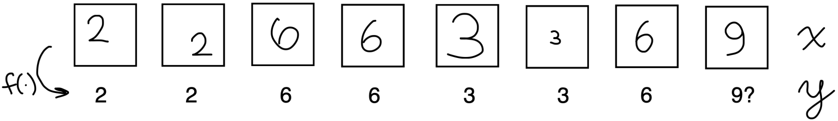
The Ingredients

We need

The Ingredients

We need

1. A way to **constrain** our learnable function to be invariant



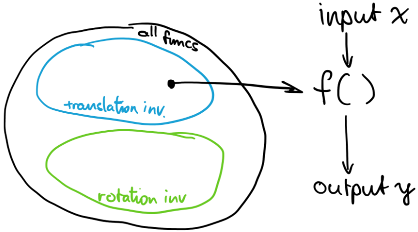
$$f(\mathbf{x}) \approx f(t(\mathbf{x};\alpha)) \qquad \forall\alpha \in \mathcal{A}_\theta$$

$$P\Big([f(t(\mathbf{x};\alpha)) - f(\mathbf{x})]^2 > L\Big) < \delta \qquad \alpha \sim p(\alpha \mid \theta)$$

The Ingredients

We need

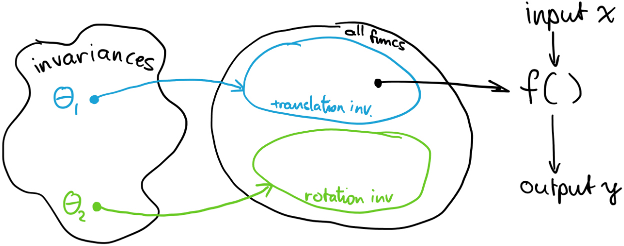
1. A way to **constrain** our learnable function to be invariant



The Ingredients

We need

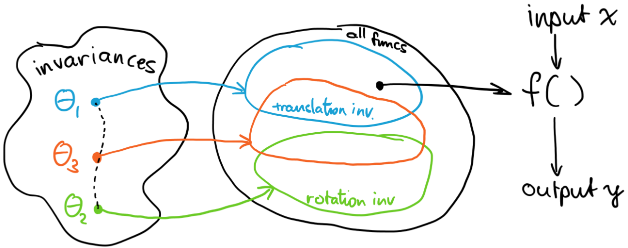
1. A way to **constrain** our learnable function to be invariant
2. A way to **parameterise** different sets of invariant functions.



The Ingredients

We need

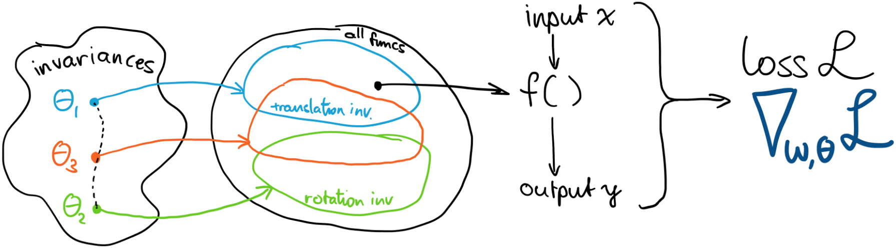
1. A way to **constrain** our learnable function to be invariant
2. A way to **parameterise** different sets of invariant functions. **Differentiably**.



The Ingredients

We need

1. A way to **constrain** our learnable function to be invariant
2. A way to **parameterise** different sets of invariant functions. **Differentiably**.
3. An objective function for learning **both** the function (i.e. weights), and invariance (i.e. θ).



Invariant functions

How do we parameterise the invariant function $f(\cdot)$?

- Sum (convolve) a non-invariant function over set of transformations we want to be invariant to!

Strict invariance (i.e. $f(\mathbf{x}) = f(t(\mathbf{x};\alpha))$ with exact equality):

$$f(\mathbf{x};\mathbf{u},\theta) = \sum_{\alpha \in \mathcal{A}_\theta} g(t(\mathbf{x};\alpha);\mathbf{u})$$

Weak invariance / data augmentation:

$$f(\mathbf{x};\mathbf{u},\theta) = \int g(t(\mathbf{x};\alpha);\mathbf{u})p(\alpha \mid \theta)\mathrm{d}\alpha$$

The function $g(\cdot;\mathbf{u})$ is parameterised by $\mathbf{u}$ and can be seen as a Gaussian process or a single-layer NN.

Training procedure

1. Generate a sample of transformed images (reparam trick $p(\alpha \mid \theta)$):

$$\{\mathbf{x}^{(s)} = t(\mathbf{x}, \alpha^{(s)})\}_{s=1}^{S} \qquad \alpha^{(s)} = h(\epsilon^{(s)}, \theta) \qquad \epsilon^{(s)} \overset{iid}{\sim} p(\epsilon)$$

2. Monte Carlo estimate of invariant function $f(\mathbf{x})$:

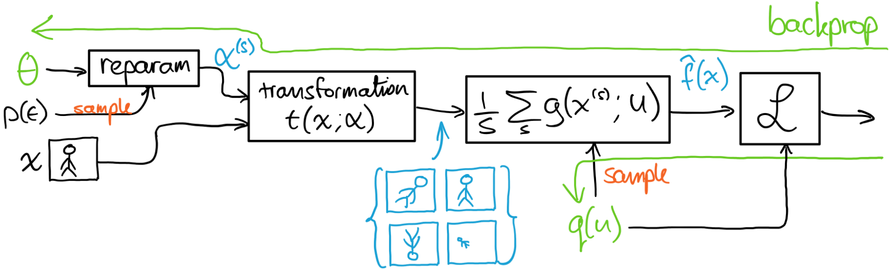
$$\hat{f}(\mathbf{x}) = \frac{1}{S} \sum_{s=1}^{S} g(\mathbf{x}^{(s)}; \mathbf{u})$$

3. Compute unbiased estimate ELBO using MC estimate of $f(\mathbf{x})$:

$$\mathcal{L} = N \cdot \mathbb{E}_{q(\mathbf{u})}\left[\log p(y_n | \hat{f}(\mathbf{x}_n)\right] - \mathrm{KL}[q(\mathbf{u}) || p(\mathbf{u}|\theta)]$$

4. Backpropagate to get gradients!

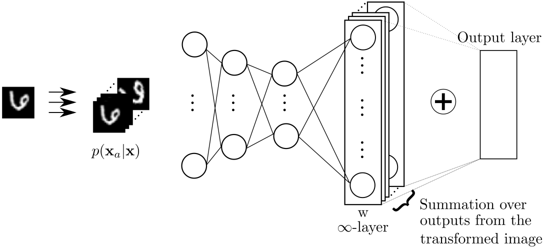
Training procedure



- Be Bayesian about the function $g(\cdot;\mathbf{u})$
- Averaging the output of $g(\cdot;\mathbf{u})$ (data aug)
- Compute an approximation to the marginal likelihood
- Backpropagate

Learning Invariances in DNNs

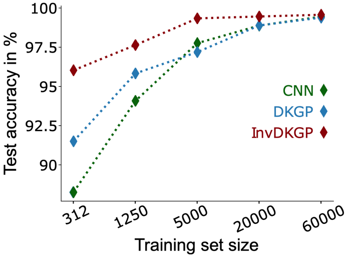
Find invariances through backprop for a Deep Neural Network, by only computing the marginal likelihood for the last layer.



- Can find invariance / data augmentation by backpropagation!
- Difficult training procedure...
- Pola Schwöbel, Martin Jørgensen, MvdW.

Learning Invariances in DNNs

Find invariances through backprop for a Deep Neural Network, by only computing the marginal likelihood for the last layer.



- Can find invariance / data augmentation by backpropagation!
- Difficult training procedure...
- Pola Schwöbel, Martin Jørgensen, MvdW.

Next Steps

- Better objective functions that correctly regularise *all* parameters
- Learning convolutions in individual layers
- Decentralising computation

Overview

The Promises of Bayesian Model Selection

Difficulties with Bayesian Inference in Deep Learning

Ensembles and Architecture Search

Conclusion

Variational Bayesian Model Selection in DNNs

How does this work when applied in DNNs?

"Empirically we found optimising the parameters of a prior $p(\mathbf{w})$ *(by taking derivatives of (1)) to not be useful, and yield worse results."*
Weight Uncertainty in Neural Networks, (Blundell et al., 2015)

Variational Bayesian Model Selection in DNNs

How does this work when applied in DNNs?

"Empirically we found optimising the parameters of a prior $p(\mathbf{w})$ (by taking derivatives of (1)) to not be useful, and yield worse results."
Weight Uncertainty in Neural Networks, (Blundell et al., 2015)

- Common failure mode: For $p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}; 0, \sigma^2 \boldsymbol{I}\right)$, $\sigma \to 0$.

Variational Bayesian Model Selection in DNNs

How does this work when applied in DNNs?

"Empirically we found optimising the parameters of a prior $p(\mathbf{w})$ (by taking derivatives of (1)) to not be useful, and yield worse results."
Weight Uncertainty in Neural Networks, (Blundell et al., 2015)

- Common failure mode: For $p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}; 0, \sigma^2 \boldsymbol{I}\right)$, $\sigma \to 0$.
- I have observed when attempting VI in DNNs:

$$\mathcal{L}(\phi_{\text{opt}}, \sigma = 0) \gg \mathcal{L}(\phi_{\text{opt}}, \sigma = \sigma_{\text{sensible}}) \tag{8}$$

Variational Bayesian Model Selection in DNNs

How does this work when applied in DNNs?

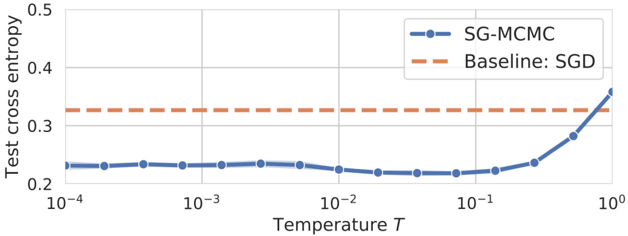
"Empirically we found optimising the parameters of a prior $p(\mathbf{w})$ (by taking derivatives of (1)) to not be useful, and yield worse results."
Weight Uncertainty in Neural Networks, (Blundell et al., 2015)

- Common failure mode: For $p(\mathbf{w}) = \mathcal{N}\left(\mathbf{w}; 0, \sigma^2 \boldsymbol{I}\right)$, $\sigma \to 0$.
- I have observed when attempting VI in DNNs:

$$\mathcal{L}(\phi_{\text{opt}}, \sigma = 0) \gg \mathcal{L}(\phi_{\text{opt}}, \sigma = \sigma_{\text{sensible}}) \tag{8}$$

- Does this mean that $\mathrm{KL}[q(f) || p(f|\mathbf{y}, \theta)]$ is large?

Cold Posteriors



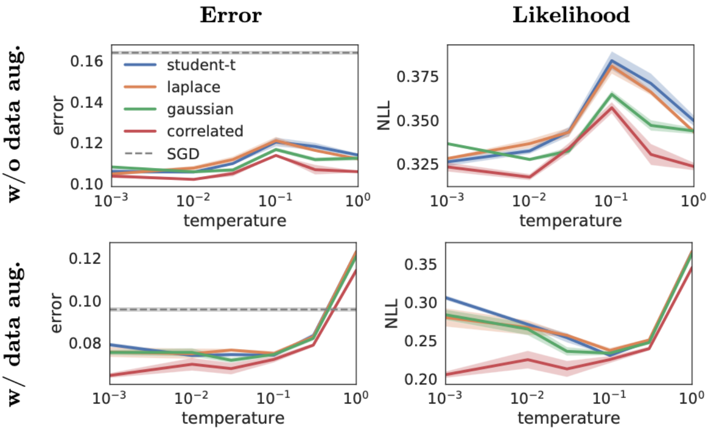
From Wenzel et al. (2020)

$$p_T(\boldsymbol{\theta}|\mathbf{y}) \propto p(\boldsymbol{\theta}|\mathbf{y})^{1/T} \tag{9}$$

- Posterior performs worse than point estimate!
- Bayes is sensitive to the prior as well!
- Does the prior make things worse?

Weight Priors

Investigate **different weight priors** in neural networks:



Bayesian Neural Network Priors Revisited

Fortuin*, Garriga-Alonso*, Wenzel, Rätsch, Turner, vdW†, Aitchison†

Weight Priors

We observe:

- Better performance with different (e.g. correlated) priors
- Data augmentation is a strong cause of the cold posterior effect

Weight Priors

We observe:

- Better performance with different (e.g. correlated) priors
- Data augmentation is a strong cause of the cold posterior effect

Possible explanations:

- Analysis of infinitely wide neural networks shows that independent weights can destroy spatial activation correlation. Correlated weights can recover this (Garriga-Alonso and van der Wilk, 2021).
- Data augmentation should be expressed as an invariance in the prior (v.d.Wilk et al., 2018).

Successful Deep Bayesian Model Selection

There is one deep model where variational bounds work for Bayesian model selection...

Successful Deep Bayesian Model Selection

There is one deep model where variational bounds work for Bayesian model selection...

- **Deep Gaussian Processes** (Damianou and Lawrence, 2013)!

Successful Deep Bayesian Model Selection

There is one deep model where variational bounds work for Bayesian model selection...

- **Deep Gaussian Processes** (Damianou and Lawrence, 2013)!
- **Theory seems to work**: Derive VI bound, get uncertainty & hyperparameters.

Successful Deep Bayesian Model Selection

There is one deep model where variational bounds work for Bayesian model selection...

- **Deep Gaussian Processes** (Damianou and Lawrence, 2013)!
- **Theory seems to work**: Derive VI bound, get uncertainty & hyperparameters.
- Deep Convolutional GPs (Blomqvist et al., 2020; Dutordoir et al., 2020) are **improving significantly**.

Successful Deep Bayesian Model Selection

There is one deep model where variational bounds work for Bayesian model selection...

- **Deep Gaussian Processes** (Damianou and Lawrence, 2013)!
- **Theory seems to work**: Derive VI bound, get uncertainty & hyperparameters.
- Deep Convolutional GPs (Blomqvist et al., 2020; Dutordoir et al., 2020) are **improving significantly**.
- New methods make DGPs easier to use, and look remarkably similar to DNNs. **Convergence?**

Successful Deep Bayesian Model Selection

There is one deep model where variational bounds work for Bayesian model selection...

- **Deep Gaussian Processes** (Damianou and Lawrence, 2013)!
- **Theory seems to work**: Derive VI bound, get uncertainty & hyperparameters.
- Deep Convolutional GPs (Blomqvist et al., 2020; Dutordoir et al., 2020) are **improving significantly**.
- New methods make DGPs easier to use, and look remarkably similar to DNNs. **Convergence?**

What makes VI bounds work in deep GPs?

Overview

The Promises of Bayesian Model Selection

Difficulties with Bayesian Inference in Deep Learning

Ensembles and Architecture Search

Conclusion

Ensembles and Posteriors

For linear models, ensembles and posteriors can be identical (Matthews et al., 2017):

$$\mathbf{w}^{(k)} \overset{\text{iid}}{\sim} \mathcal{N}(0, \boldsymbol{I}) \tag{10}$$

$$\mathbf{w}_p^{(k)} := \text{GradDesc}(\mathbf{w}^{(k)}, ||\mathbf{y} - \Phi(X)\mathbf{w}||^2) \tag{11}$$

$$\implies \mathbf{w}_p^{(k)} \overset{\text{iid}}{\sim} p(\mathbf{w}|\mathbf{y}) \tag{12}$$

where $p(y_n|\mathbf{w}) = \mathcal{N}\left(y_n; \boldsymbol{\phi}(\mathbf{x}_n)^\intercal \mathbf{w}, \sigma^2\right)$, with $\sigma^2 \to 0$.

Can optimisation do Bayesian Model Selection?

$$\log p(\mathbf{y}) = \sum_{i=1}^{n} \log p(y_i \mid \mathbf{y}_{<i}) = \sum_i \log \mathbb{E}_{p(\mathbf{w}|\mathbf{y}_{<i})}[p(y_i|\mathbf{w})]$$
$$\geqslant \sum_i \mathbb{E}_{p(\mathbf{w}|\mathbf{y}_{<i})}[\log p(y_i|\mathbf{w})]$$

For linear models the answer is an unambiguous **yes**:



Clare Lyle, Lisa Schut, Binxin Ru, Yarin Gal, MvdW

Can optimisation do Bayesian Model Selection?

$$\log p(\mathbf{y}) = \sum_{i=1}^{n} \log p(y_i \mid \mathbf{y}_{<i}) = \sum_i \log \mathbb{E}_{p(\mathbf{w}|\mathbf{y}_{<i})}[p(y_i|\mathbf{w})]$$
$$\geqslant \sum_i \mathbb{E}_{p(\mathbf{w}|\mathbf{y}_{<i})}[\log p(y_i|\mathbf{w})]$$

For linear models the answer is an unambiguous **yes**:

- exact posterior samples can be produced to minimizing an unregularized loss (Matthews et al., 2017),



Clare Lyle, Lisa Schut, Binxin Ru, Yarin Gal, MvdW

Can optimisation do Bayesian Model Selection?

$$
\begin{aligned}
\log p(\mathbf{y}) &= \sum_{i=1}^{n} \log p(y_i \mid \mathbf{y}_{<i}) = \sum_i \log \mathbb{E}_{p(\mathbf{w}|\mathbf{y}_{<i})}[p(y_i|\mathbf{w})] \\
&\geqslant \sum_i \mathbb{E}_{p(\mathbf{w}|\mathbf{y}_{<i})}[\log p(y_i|\mathbf{w})]
\end{aligned}
$$

For linear models the answer is an unambiguous **yes**:

- exact posterior samples can be produced to minimizing an unregularized loss (Matthews et al., 2017),
- so we can get a lower bound to the marginal likelihood by summing training losses.



Clare Lyle, Lisa Schut, Binxin Ru, Yarin Gal, MvdW

Can optimisation do Bayesian Model Selection?

$$\log p(\mathbf{y}) = \sum_{i=1}^{n} \log p(y_i \mid \mathbf{y}_{<i}) = \sum_i \log \mathbb{E}_{p(\mathbf{w}|\mathbf{y}_{<i})}[p(y_i|\mathbf{w})]$$
$$\geqslant \sum_i \mathbb{E}_{p(\mathbf{w}|\mathbf{y}_{<i})}[\log p(y_i|\mathbf{w})]$$

For linear models the answer is an unambiguous **yes**:

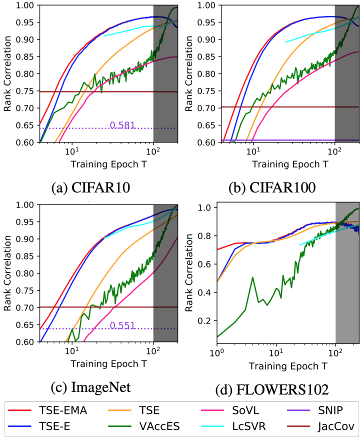
- exact posterior samples can be produced to minimizing an unregularized loss (Matthews et al., 2017),
- so we can get a lower bound to the marginal likelihood by summing training losses.

- Bayesian Perspective on Training Speed and Model Selection

Clare Lyle, Lisa Schut, Binxin Ru, Yarin Gal, MvdW

Neural Architecture Search

Inspired by this, we investigated whether training speed could predict testing accuracy of a network.

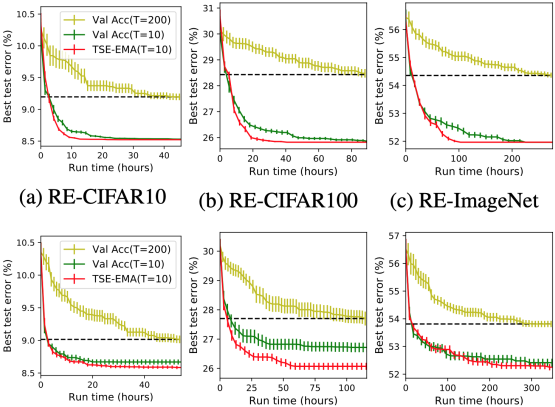


(a) CIFAR10 (b) CIFAR100

(c) ImageNet (d) FLOWERS102

Neural Architecture Search

Inspired by this, we investigated whether training speed could predict testing accuracy of a network.



(a) RE-CIFAR10 (b) RE-CIFAR100 (c) RE-ImageNet

(d) TPE-CIFAR10 (e) TPE-CIFAR100 (f) TPE-ImageNet

Overview

The Promises of Bayesian Model Selection

Difficulties with Bayesian Inference in Deep Learning

Ensembles and Architecture Search

Conclusion

Conclusion & Ways Forward

- Bayesian model selection could provide great benefits in DNNs.

Conclusion & Ways Forward

- Bayesian model selection could provide great benefits in DNNs.
- Inference and prior specification are unsolved.

Conclusion & Ways Forward

- Bayesian model selection could provide great benefits in DNNs.
- Inference and prior specification are unsolved.
- There are examples that show that marginal likelihood maximisation works in deep models!

Conclusion & Ways Forward

- Bayesian model selection could provide great benefits in DNNs.
- Inference and prior specification are unsolved.
- There are examples that show that marginal likelihood maximisation works in deep models!

Possible ways forward:

- Perhaps we still need better approx posteriors?

Conclusion & Ways Forward

- Bayesian model selection could provide great benefits in DNNs.
- Inference and prior specification are unsolved.
- There are examples that show that marginal likelihood maximisation works in deep models!

Possible ways forward:

- Perhaps we still need better approx posteriors?
- Perhaps model misspecification is a problem? (Wenzel et al., 2020)

Conclusion & Ways Forward

- Bayesian model selection could provide great benefits in DNNs.
- Inference and prior specification are unsolved.
- There are examples that show that marginal likelihood maximisation works in deep models!

Possible ways forward:

- Perhaps we still need better approx posteriors?
- Perhaps model misspecification is a problem? (Wenzel et al., 2020)
- Perhaps optimization mechanisms can do the same thing? (Lyle et al., 2020)

Alternative approaches

Two approaches based on back-propagating through a validation set:

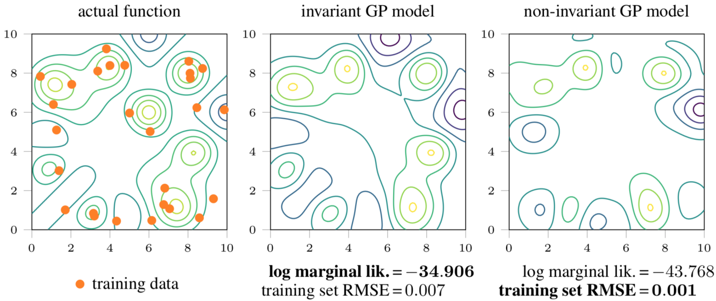
- Meta-learning (Zhou et al., 2020)
- Implicit function theorem (Lorraine et al., 2020)
- Straightforward regularization (Benton et al., 2020)

What is going to be best? Only research will tell!

Why can't we just use training loss?

For the same reason as why we need cross-validation:

- The training loss is minimised with the most flexible model
- Inductive biases are **constraints**

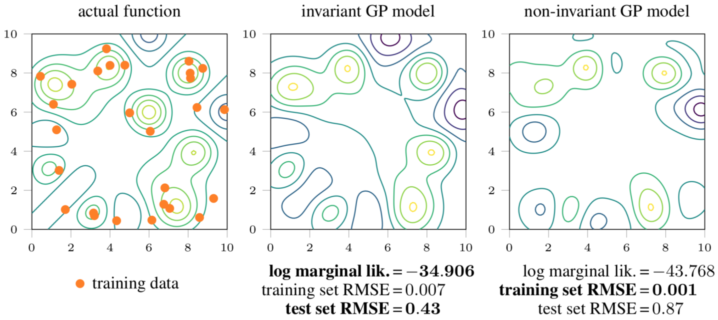


invariant GP model	non-invariant GP model
log marginal lik. = −34.906	log marginal lik. = −43.768
training set RMSE = 0.007	**training set RMSE = 0.001**
test set RMSE = 0.43	test set RMSE = 0.87

Why can't we just use training loss?

For the same reason as why we need cross-validation:

- The training loss is minimised with the most flexible model
- Inductive biases are **constraints**



Log marginal likelihood measures **generalisation**:

$$\log p(\mathbf{y} \mid \theta) = \log p(y_1|\theta) + \log p(y_2|\theta, y_1) + \log p(y_3|\theta, \{y_i\}_{i=1}^{2}) \ldots$$

(It's also related to cross-validation (Fong and Holmes, 2020).)

References I

Benton, G., Finzi, M., Izmailov, P., and Wilson, A. G. (2020). Learning invariances in neural networks.

Berger, J. O. (1985). Statistical decision theory and Bayesian analysis. Springer.

Blomqvist, K., Kaski, S., and Heinonen, M. (2020). Deep convolutional gaussian processes. In Brefeld, U., Fromont, E., Hotho, A., Knobbe, A., Maathuis, M., and Robardet, C., editors, Machine Learning and Knowledge Discovery in Databases, pages 582–597, Cham. Springer International Publishing.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. (2015). Weight uncertainty in neural network. In Bach, F. and Blei, D., editors, Proceedings of the 32nd International Conference on Machine Learning, volume 37 of Proceedings of Machine Learning Research, pages 1613–1622, Lille, France. PMLR.

References II

Damianou, A. C. and Lawrence, N. D. (2013). Deep Gaussian processes. In Proceedings of the 16th International Conference on Artificial Intelligence and Statistics, pages 207–215.

Dutordoir, V., van der Wilk, M., Artemev, A., and Hensman, J. (2020). Bayesian image classification with deep convolutional gaussian processes. In Chiappa, S. and Calandra, R., editors, Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, volume 108 of Proceedings of Machine Learning Research, pages 1529–1539, Online. PMLR.

Fong, E. and Holmes, C. C. (2020). On the marginal likelihood and cross-validation. Biometrika, 107(2):489–496.

Garriga-Alonso, A. and van der Wilk, M. (2021). Correlated weights in infinite limits of deep convolutional neural networks. In Third Symposium on Advances in Approximate Bayesian Inference.

References III

Khan, M. E. E., Immer, A., Abedi, E., and Korzepa, M. (2019). Approximate inference turns deep networks into gaussian processes. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, Advances in Neural Information Processing Systems, volume 32, pages 3094–3104. Curran Associates, Inc.

Lorraine, J., Vicol, P., and Duvenaud, D. (2020). Optimizing millions of hyperparameters by implicit differentiation. In Chiappa, S. and Calandra, R., editors, Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics, volume 108 of Proceedings of Machine Learning Research, pages 1540–1552, Online. PMLR.

Lyle, C., Schut, L., Ru, B., Gal, Y., and van der Wilk, M. (2020). A bayesian perspective on training speed and model selection.

References IV

Matthews, A. G. d. G., Hron, J., Turner, R. E., and Ghahramani, Z. (2017). Sample-then-optimize posterior sampling for bayesian linear models. In NeurIPS Workshop on Advances in Approximate Bayesian Inference.

v.d.Wilk, M., Bauer, M., John, S., and Hensman, J. (2018). Learning invariances using the marginal likelihood. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, Advances in Neural Information Processing Systems 31, pages 9938–9948. Curran Associates, Inc.

Wenzel, F., Roth, K., Veeling, B., Swiatkowski, J., Tran, L., Mandt, S., Snoek, J., Salimans, T., Jenatton, R., and Nowozin, S. (2020). How good is the Bayes posterior in deep neural networks really? In III, H. D. and Singh, A., editors, Proceedings of the 37th International Conference on Machine Learning, volume 119 of Proceedings of Machine Learning Research, pages 10248–10259, Virtual. PMLR.

References V

Zhou, A., Knowles, T., and Finn, C. (2020). Meta-learning symmetries by reparameterization.