Fully Connected, Cambridge, UK

Imperial College London

A Smarter Neuron
Automating Neural Architecture Design

Mark van der Wilk

Department of Computing
Imperial College London
`https://mvdw.uk`

@markvanderwilk
`m.vdwilk@imperial.ac.uk`

Apr 25, 2022

About our research group

- 2020–: Lecturer (Assistant Prof) at Imperial College London.
- Currently growing a research group.
- Research focus:
  - Gaussian process inference, backed by theory to make it **reliable**.
  - Automatic learning of inductive bias in neural networks.
    Central question: **When should neurons be connected?**


Anish Dhir


Artem Artemev


Jose Pablo Folch


Ruby Sedgwick


Seth Nabarro


Tycho van der Ouderaa

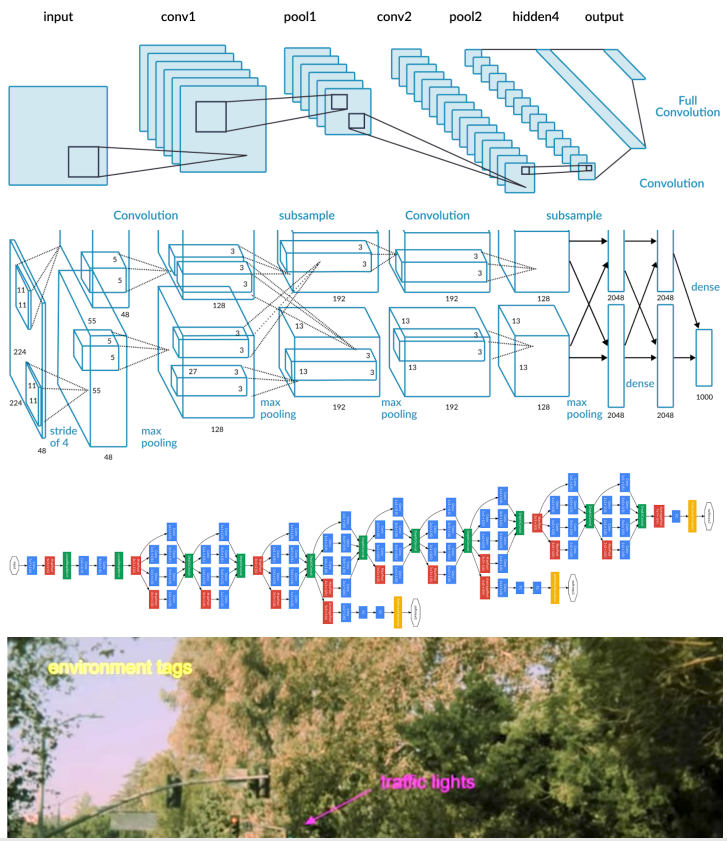
Hyperparameter Selection & Architecture Design

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

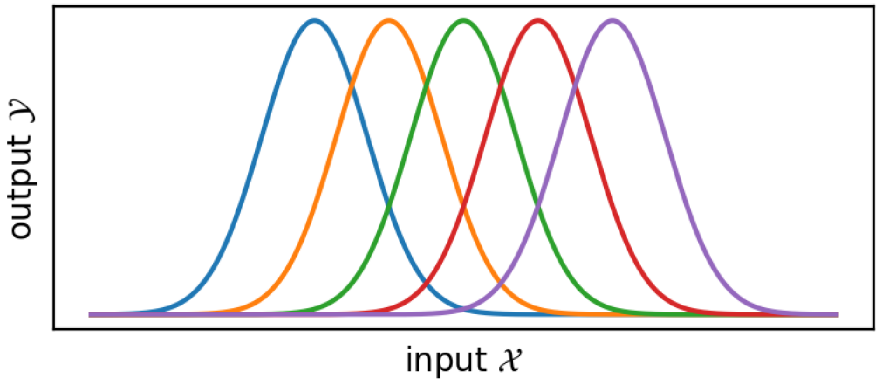
- How many layers? How many units in a layer?
- What layer structure? Convolutional? Skip connections?
- Data augmentation parameters?

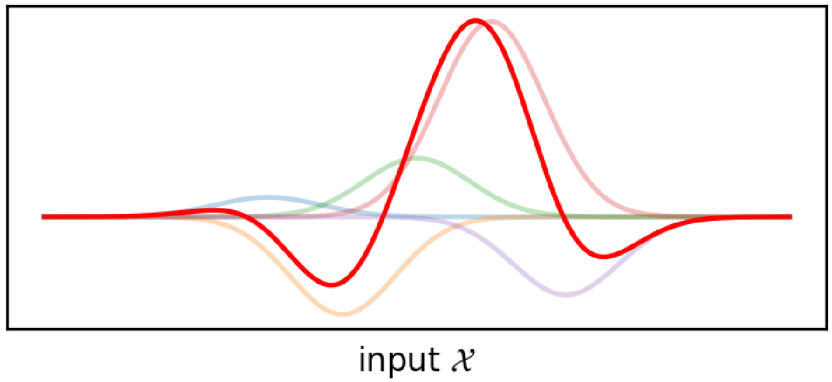
As architectures get more complex, so does design! E.g. multitask.

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



Hyperparameter Selection Example





$$f_{\mathbf{w},\boldsymbol{\theta}}(x) = \boldsymbol{\phi}_{\boldsymbol{\theta}}(x)^{\mathsf{T}}\mathbf{w} = \sum_{i=1}^{K} \phi_{\boldsymbol{\theta}}^{(i)}(x) w_i \tag{1}$$

$$\mathcal{L}_{\text{train}} = \sum_{n=1}^{N_{\text{train}}} (f_{\mathbf{w},\boldsymbol{\theta}}(x_n) - y_n)^2 + \lambda ||\mathbf{w}||^2 \tag{2}$$

- Sum basis functions with weights $\mathbf{w}$, hyperparameters $\boldsymbol{\theta}$ control "wigglyness".
- Normally, we minimize $\mathcal{L}_{\text{train}}$ w.r.t. $\mathbf{w}$, while keeping hyperparameters $\boldsymbol{\theta}$ fixed.
- Architectural choices change the **inductive bias**, like hyperparameters $\boldsymbol{\theta}$ here change the width of basis functions.

Why do we need cross-validation?

What happens if we minimise $\mathcal{L}_{\text{train}}$ w.r.t. both $\mathbf{w}$ **and** $\boldsymbol{\theta}$?

- Training loss learns weights, only with **fixed hyperparameters**.
- Why? Inductive bias is a **restriction** on functions.
  Least restriction is best for training loss.
- Cross-validation: Try different values of $\boldsymbol{\theta}$,
  measure performance on separate **validation set**.
- Goal: Find **objective function** for hyperparameters
  that we can optimise with **gradients**.

Bayesian Model Selection

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

$$p(f, \theta|\mathbf{y}) = \frac{p(\mathbf{y}|f)p(f|\theta)p(\theta)}{p(\mathbf{y})} = \underbrace{\frac{p(\mathbf{y}|f)p(f|\theta)}{p(\mathbf{y}|\theta)}}_{\text{usual posterior}} \underbrace{\frac{p(\mathbf{y}|\theta)p(\theta)}{p(\mathbf{y})}}_{\text{hyper posterior}} \tag{3}$$

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

$$p(\mathbf{y}|\theta) = \int p(\mathbf{y} \mid f)p(f \mid \theta)\mathrm{d}\theta \tag{4}$$

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

$$\theta^* = \operatorname*{argmax}_{\theta} \log p(\mathbf{y} \mid \theta)\,, \qquad \text{predict using } p(f|\mathbf{y}, \theta^*) \tag{5}$$

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

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

Bayesian Model Selection: Example

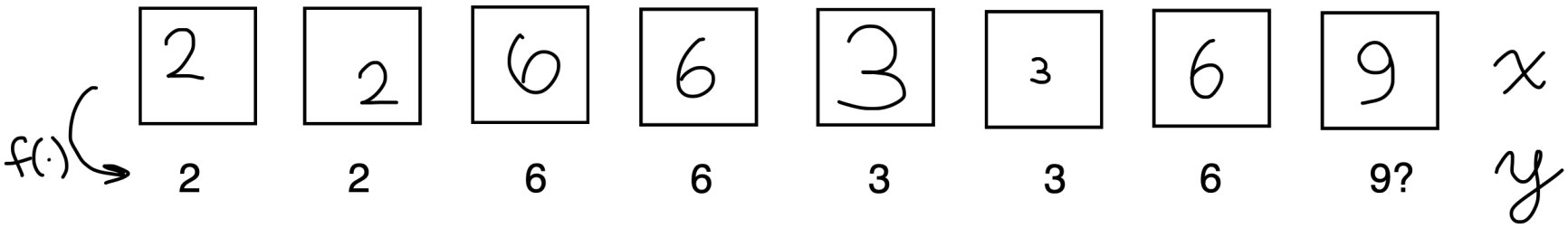
- Here, we optimise **marginal likelihood** instead of **training loss**.
- Can still be computed on **training data only**.
- And, we can **compute gradients**!
- No more trial-and-error, here hyperparameter selection is **just as easy as learning weights**!

Idea:

Learn NN architectural parameters in this way!

Model Selection in NNs: Learning Invariances

Architecture choice helps determine **invariances** in **inductive bias**:



- Convolutions are a common solution
- Can convolve according to other transformations too (e.g. rotations)

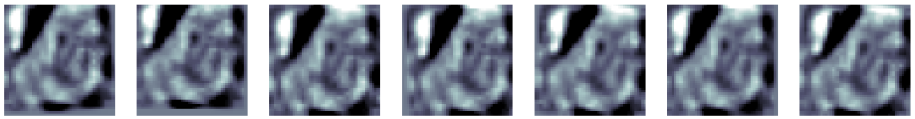
Can we automatically adjust invariance properties in layers?

Learning Invariances

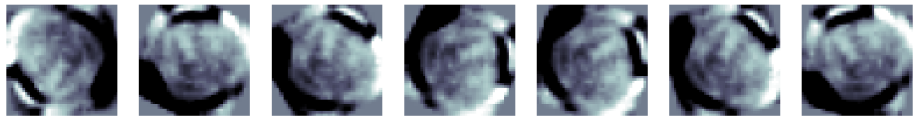
Rotated MNIST dataset: 

Learned filters:

Filters for other transformed MNIST variants:



(a) Sampled filters of affine model trained on regular mnist.



(b) Sampled filters of affine model trained on rotated mnist.

(c) Sampled filters of affine model trained on scaled mnist.

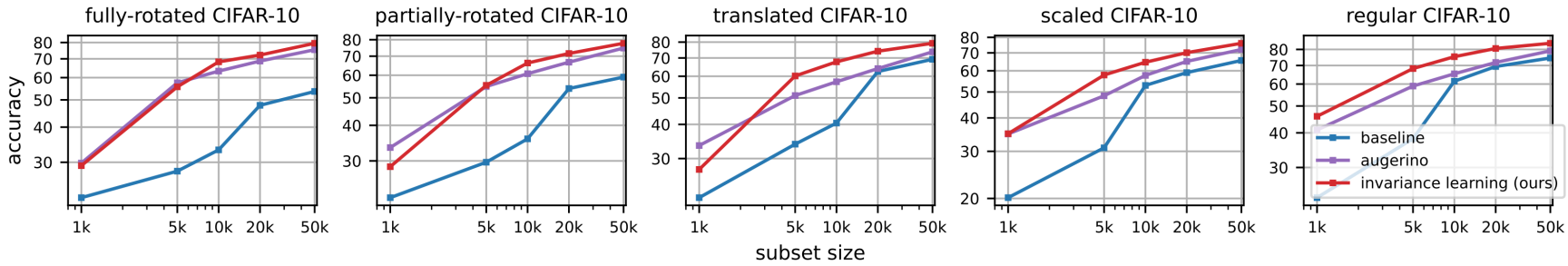
(d) Sampled filters of affine model trained on translated mnist.

Learning Invariances: Papers

- **Learning Invariances using the Marginal Likelihood**
  (van der Wilk et al., 2018)
  Learning invariance by backprop, but Gaussian processes only.
- **Learning Invariant Weights in Neural Networks**
  (van der Ouderaa and van der Wilk, 2021)
  Show how filter banks can be learned, but shallow NNs only.
- **Last Layer Marginal Likelihood for Invariance Learning**
  (Schwöbel et al., 2022)
  We made a start at getting it to work for deep NNs.

Invariance Learning in NNs

- **Invariance Learning in Deep Neural Networks with Differentiable Laplace Approximations**
  (Immer et al., 2022)
  We show that the marginal likelihood works in deep NNs, and is competitive.



- + papers on other approaches e.g. more classical Neural Architecture Search (Ru et al., 2021)

Future: A Smarter Neuron

Goal:

Neural networks that self-tune inductive biases, and all hyperparameters.

- We're starting to automate finding inductive biases that previously had to be searched by trial-and-error.
- Ultimately, inductive bias is determined by **connectivity structure**.
- We need to find **a smarter neuron** that finds *which* neurons it should connect to, in addition to weights through backprop.
- When should neurons connect? When should they appear? When should they be removed?

Outlook

- We want something better than trial-and-error to design NNs.
- Bayesian methods are helping the automation of selecting **invariances**, and making it as easy as **backprop**!
- Can help make NNs **1)** more accurate, **2)** easier to use, **3)** more energy-efficient.
- A lot more to do to get to **the smarter neuron!**
  Meta-learning? More Bayes? Causality? Cellular automata?

Hard to say, but it'll be fun to find out!

Join us!


Anish Dhir


Artem Artemev


Jose Pablo Folch


Ruby Sedgwick


Seth Nabarro


Tycho van der Ouderaa

- There will be an open PhD position in 2023
  (bit late for Oct 2022, but you could still get in touch).
- Check my website (`https://mvdw.uk/`) for tips on applying, and how to get in touch.
- Topics: Invariance, Bayes, Gaussian processes, BayesOpt, PAC-Bayes, causality, meta-learning, model-based RL.

References I

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

Immer, A., van der Ouderaa, T. F. A., Fortuin, V., Rätsch, G., and van der Wilk, M. (2022). Invariance learning in deep neural networks with differentiable laplace approximations.

Ru, B., Lyle, C., Schut, L., Fil, M., van der Wilk, M., and Gal, Y. (2021). Speedy performance estimation for neural architecture search. In Advances in Neural Information Processing Systems (NeurIPS), volume 34.

Schwöbel, P., Jørgensen, M., Ober, S. W., and van der Wilk, M. (2022). Last layer marginal likelihood for invariance learning. In Proceedings of the Twenty Fifth International Conference on Artificial Intelligence and Statistics (AISTATS).

References II

van der Ouderaa, T. and van der Wilk, M. (2021). Learning invariant weights in neural networks. In ICML 2021 Workshop on Uncertainty & Robustness in Deep Learning.

van der Wilk, M., Bauer, M., John, S., and Hensman, J. (2018). Learning invariances using the marginal likelihood. In Advances in Neural Information Processing Systems 31.