Deep Convolutional Networks as shallow Gaussian Processes

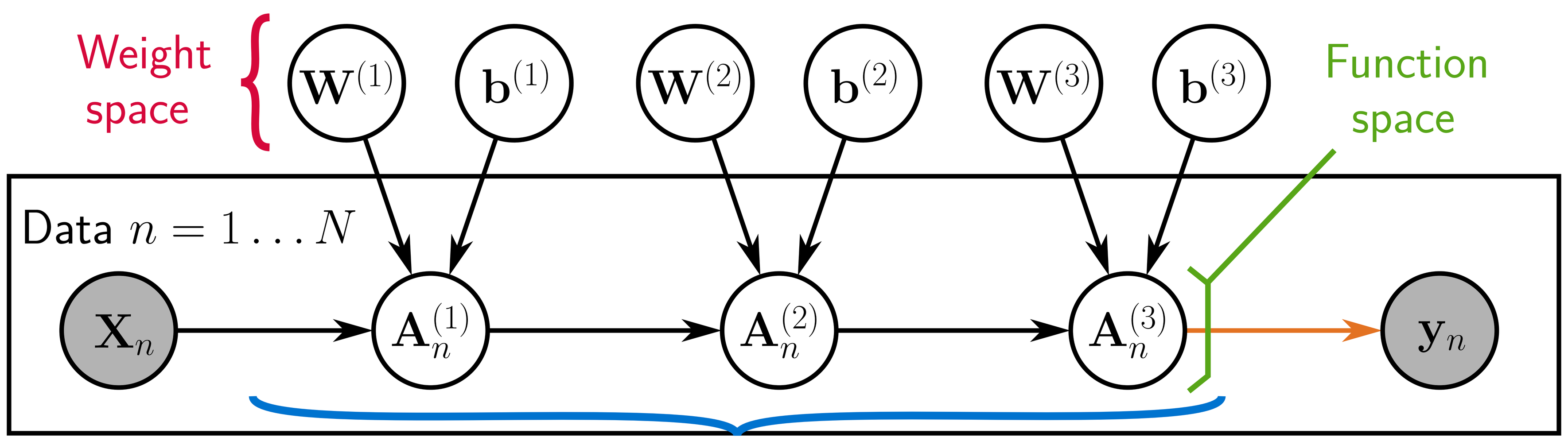
UNIVERSITY OF CAMBRIDGE

Web: agarri.ga/convnets-as-gps

Adrià Garriga-Alonso, Carl Edward Rasmussen & Laurence Aitchison

Example: Bayesian NN for supervised learning

Define a **prior distribution** over **functions** $\mathbf{f}(\mathbf{X})$ by fixing a neural network (NN) architecture, and making its weights and biases $\mathcal{W} := \{\mathbf{W}^{(\ell)}, \mathbf{b}^{(\ell)}\}_{\ell=1}^{3}$ random. $\mathbf{A}_n^{(\ell)} := \mathbf{A}^{(\ell)}(\mathbf{X}_n)$ are pre-activations.



- Distributions over:

  weight space $p(\mathcal{W} \mid \mathcal{D}) \Rightarrow$ function space $p(\mathbf{A}^{(3)} \mid \mathbf{X}, \mathcal{D})$.
- **Can we more easily do Bayesian inference over functions $\mathbf{A}^{(3)}(\mathbf{X})$?**

A convolutional network prior

Network architecture, recursively defined on preactivations for each layer ℓ:

$$\mathbf{a}_i^{(1)}(\mathbf{X}) := b_i^{(1)}\mathbf{1} + \sum_{j=1}^{C^{(0)}} \mathbf{W}_{i,j}^{(1)} * \mathbf{x}_j, \tag{1a}$$

$$\mathbf{a}_i^{(\ell+1)}(\mathbf{X}) := b_i^{(\ell+1)}\mathbf{1} + \sum_{j=1}^{C^{(\ell)}} \mathbf{W}_{i,j}^{(\ell+1)} * \phi\left(\mathbf{a}_j^{(\ell)}(\mathbf{X})\right), \tag{1b}$$

- $C^{(\ell)}$: number of channels, $C^{(\ell)} = f_\ell(n)$ for a monotonically increasing f_ℓ.
- ϕ: elementwise nonlinearity (ReLU, erf), with $|\phi(x)| \le c + m|x|$ for all $x \in \mathbb{R}$.
- i, j: indices over channels.

The elements of each filter $\mathbf{W}_{i,j}$ and bias b_i are independent with zero mean,

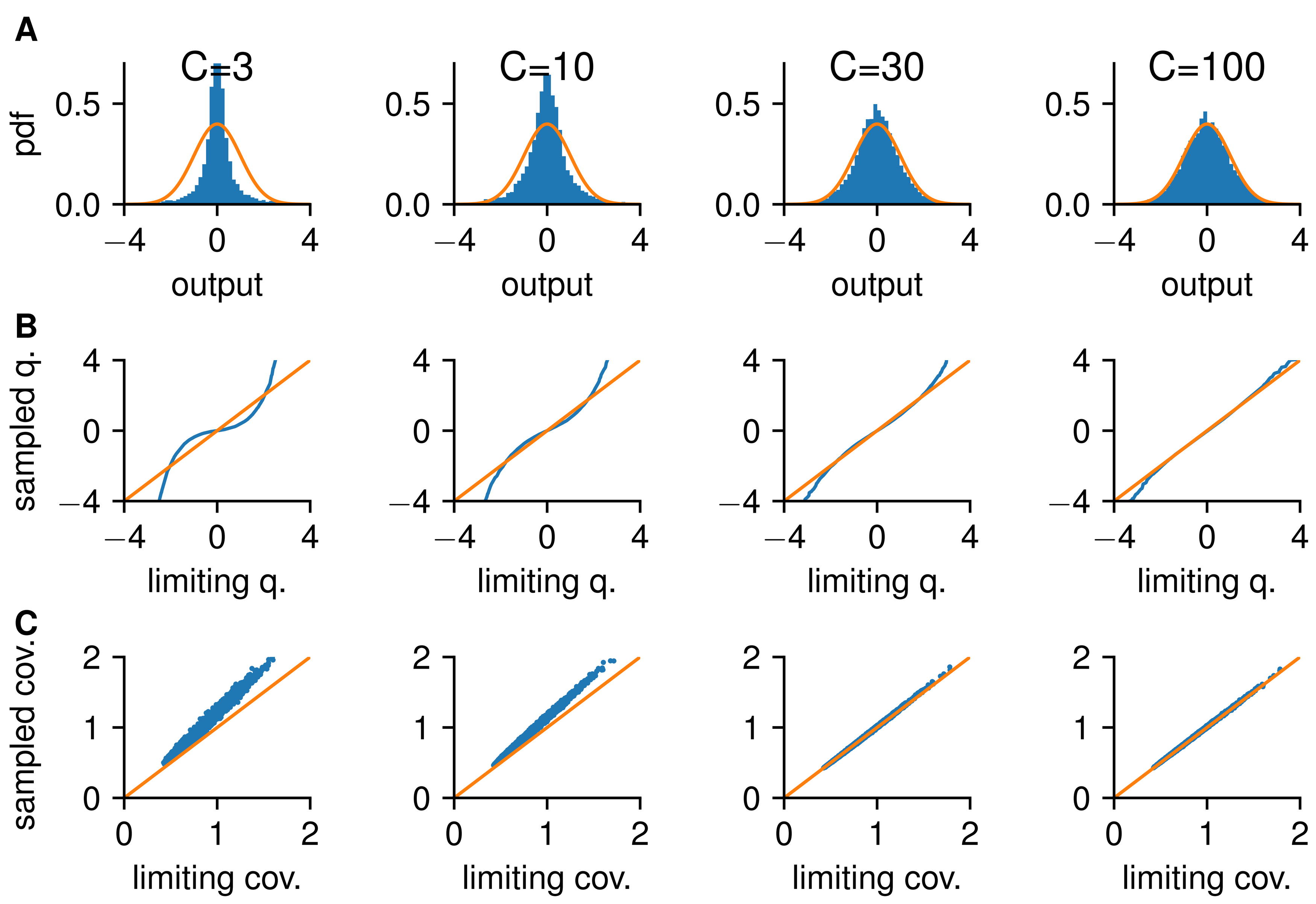
$$W_{i,j,x,y}^{(\ell)} \sim \mathcal{N}\left(0, \sigma_\text{w}^2 / C^{(\ell)}\right), \quad b_i^{(\ell)} \sim \mathcal{N}\left(0, \sigma_b^2\right). \tag{2}$$

Alternatively to (1b) for a residual network, with a skip connection from layer $\ell - s$:

$$\mathbf{a}_i^{(\ell+1)}(\mathbf{X}) := \mathbf{a}_i^{(\ell-s)}(\mathbf{X}) + b_i^{(\ell+1)}\mathbf{1} + \sum_{j=1}^{C^{(\ell)}} \mathbf{W}_{i,j}^{(\ell+1)} * \phi\left(\mathbf{a}_j^{(\ell)}(\mathbf{X})\right). \tag{3}$$

Convergence to a Gaussian process

Theorem: For any countable input set $\{\mathbf{X}_n\}_{n=1}^{\infty}$, the outputs $\mathbf{A}_n^{(L)}$ of the random CNN converge in distribution to a **Gaussian process** as $n \to \infty$.



Distribution of the output $\mathbf{A}^{(32)}(\mathbf{X})$: limiting density and samples of finite 32-layer ResNets [3] with $C = 3, 10, 30, 100$ channels.

A) Histogram and density function for an input $\mathbf{X}$.
B) Quantile-quantile plot: value of matching quantiles for distributions in A).
C) Covariances of $\mathbf{A}^{(32)}(\mathbf{X})$ and $\mathbf{A}^{(32)}(\mathbf{X}')$ for each pair in 100 input images.

The ConvNet and ResNet kernels

A Gaussian process is fully specified by its mean and covariance functions.
$\Rightarrow$ Therefore, we need the mean and covariance of the outputs of the NN.

$$\mathbb{E}\left[\mathbf{a}_i^{(\ell)}(\mathbf{X})\right] = 0. \qquad \text{If } i \ne j,\ \mathbb{C}\left[\mathbf{a}_i^{(\ell)}(\mathbf{X}), \mathbf{a}_j^{(\ell)}(\mathbf{X}')\right] = \mathbf{0}. \tag{4}$$

Covariance $K_\mu^{(\ell+1)}$ of the μth element of the pre-activation, for inputs $\mathbf{X}$ and $\mathbf{X}'$:

$$K_\mu^{(1)}(\mathbf{X}, \mathbf{X}') = \sigma_\text{b}^2 + \frac{\sigma_\text{w}^2}{C^{(0)}} \sum_{i=1}^{C^{(0)}} \sum_{\nu \in \mu\text{th patch}} X_{i,\nu} X'_{i,\nu}, \tag{5a}$$

$$K_\mu^{(\ell+1)}(\mathbf{X}, \mathbf{X}') = \sigma_\text{b}^2 + \sigma_\text{w}^2 \sum_{\nu \in \mu\text{th patch}} \underbrace{\mathbb{E}\left[\phi(A_{j,\nu}^{(\ell)}(\mathbf{X}))\phi(A_{j,\nu}^{(\ell)}(\mathbf{X}'))\right]}_{\text{depends only on bivariate Gaussian } [A_{j,\nu}^{(\ell)}(\mathbf{X}), A_{j,\nu}^{(\ell)}(\mathbf{X}')]} \tag{5b}$$

- Because of $\sum_{\mu\text{th patch}}$, evaluating $K_\mu^{(\ell)}$ is a **1-filter convolutional network forward pass** with **fixed weights**.
- If we only need the diagonal of the covariance at ℓ, $\mathbf{K}^{(\ell)}$, we only need the diagonal at $\ell - 1$. **Per-layer cost O(#pixels) instead of $\mathrm{O}(\#\text{pixels}^2)$.**
  - Downside: no longer true in a network with pooling.
- Because of layer-wise independence, the kernel for a ResNet (eq. 2) is

$$K_\mu^{(\ell+1)}(\mathbf{X}, \mathbf{X}') = K_\mu^{(\ell-s)}(\mathbf{X}, \mathbf{X}') + \sigma_\text{b}^2 + \sigma_\text{w}^2 \sum_{\nu \in \mu\text{th patch}} \mathbb{E}\left[\phi_{j,\nu}\phi'_{j,\nu}\right]. \tag{6}$$

- Resulting kernel has **very few parameters**, only σ_b^2, σ_w^2 and NN architecture.

Experiments

- Classification likelihoods not conjugate for GPs $\Rightarrow$ use Gaussian likelihood.
- Regression targets $\mathbf{Y} \in \{-1, 1\}^{N \times 10}$: one-hot encoding of the class.
- MNIST, $N = 50$k training, $M = 10$k validation, $M = 10$k test.

1. Compute $N \times N$ kernel matrix $\mathbf{K}_{xx}$ and $M \times N$ matrices $\mathbf{K}_{x,x_V}$, $\mathbf{K}_{x,x_T}$. For 32-layer ResNet: $O(N^2)$, but 3h 40min on two Tesla P100 GPUs.
2. Training/validation: compute $\mathbf{K}_{xx}^{-1}\mathbf{Y}$, then $\mathbf{K}_{x,x_V}\mathbf{K}_{xx}^{-1}\mathbf{Y}$. Takes $O(N^3)$, but in practice fast! 43.25 ± 8.8 **seconds** on one Tesla P100.
3. Parameter search: compute several $\mathbf{K}_{xx}$ with random parameters, choose best validation accuracy.

Method	#samples	Validation error	Test error
NNGP [5]	≈ 250	–	1.21%
Convolutional GP [6]	SGD	–	1.17%
Deep Conv. GP [4]	SGD	–	1.34%
ConvNet GP (1-12 layers)	27	0.71%	1.03%
Residual CNN GP (1-12 layers)	27	0.71%	0.93%
ResNet GP (32 layers, like [3])	–	0.68%	**0.84%**
GP + parametric deep kernel [1]	SGD	–	0.60%
ResNet [2]	–	–	**0.41%**

Takeaways

- Can do Bayesian inference in **deep** (residual) infinite CNNs, using **shallow** GPs.
- **No gradient descent used here**, but possible to tune parameters with it.
- Unexplained performance gap with NNs trained with SGD.
- Kernel still expensive, hard to scale with inducing points.

References

[1] John Bradshaw, Alexander G de G Matthews, and Zoubin Ghahramani. "Adversarial examples, uncertainty, and transfer testing robustness in gaussian process hybrid deep networks". In: *arXiv preprint arXiv:1707.02476* (2017).

[2] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. "Neural Ordinary Differential Equations". In: *arXiv preprint arXiv:1806.07366* (2018).

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Identity mappings in deep residual networks". In: *European conference on computer vision*. Springer. 2016, pp. 630–645.

[4] Vinayak Kumar, Vaibhav Singh, PK Srijith, and Andreas Damianou. "Deep Gaussian Processes with Convolutional Kernels". In: *arXiv preprint arXiv:1806.01655* (2018).

[5] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. "Deep Neural Networks as Gaussian Processes". In: *arXiv preprint arXiv:1711.00165* (2017).

[6] Mark van der Wilk, Carl Edward Rasmussen, and James Hensman. "Convolutional Gaussian Processes". In: *Advances in Neural Information Processing Systems*. 2017, pp. 2845–2854.