An Improved Variational Approximate Posterior for the Deep Wishart Process

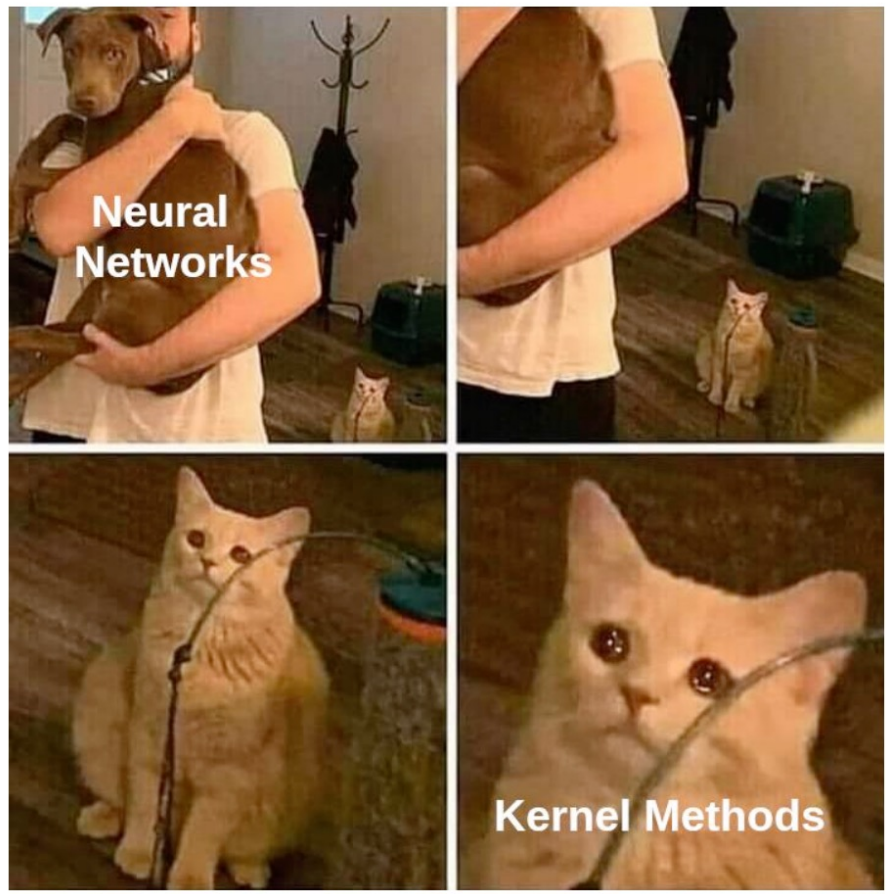
Sebastian W. Ober


Laurence Aitchison


Adam Yang

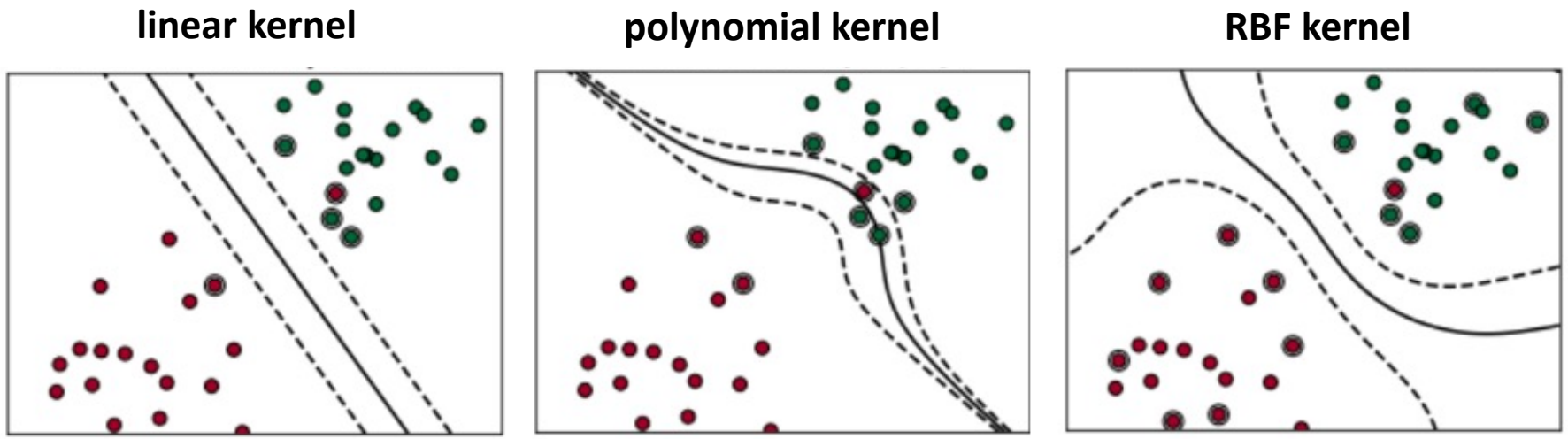

Edward Milsom


Ben Anson

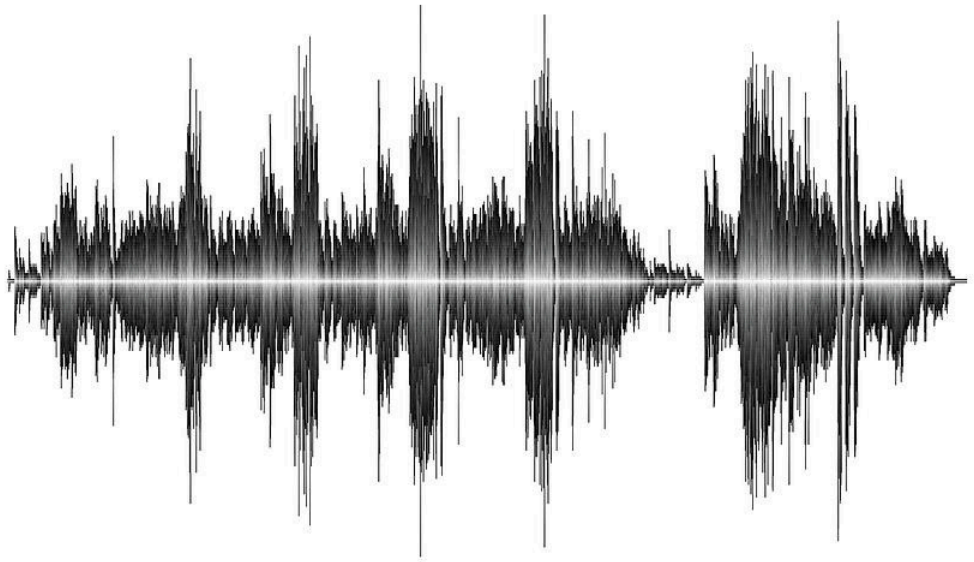

Neural
Networks
Kernel Methods

	shallow	deep
feature	linear regression	
kernel	kernel ridge regression	

For good performance, we need to choose a choosing a good feature extractor / kernel

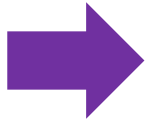


Problem: can’t choose a good feature extractor/kernel for complex data like images

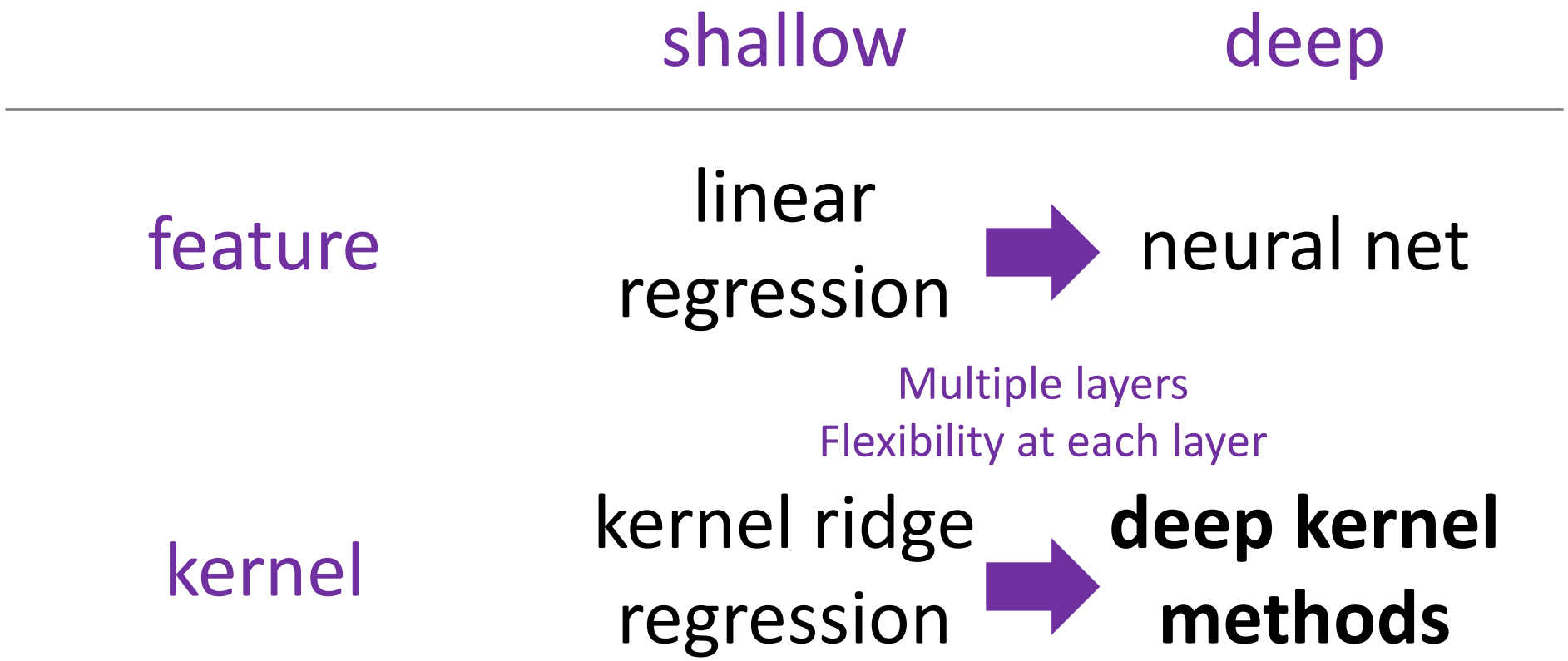


	shallow
feature	linear regression
kernel	kernel ridge regression

	shallow	deep
feature	linear regression	neural net
kernel	kernel ridge regression	



Multiple layers
Flexibility at each layer


shallow
deep
feature
linear regression
neural net
Multiple layers
Flexibility at each layer
kernel
kernel ridge regression
deep kernel methods

Summary

rewrite prior in terms of kernel/Gram matrices

deep Gaussian processes

deep kernel **processes**

deep kernel **machines**

Infinite-width limit of Bayesian method

Part 1

rewrite prior in terms of kernel/Gram matrices

deep Gaussian processes

Infinite-width limit of Bayesian method

deep kernel **processes**

deep kernel **machines**

In deep-kernel methods, we switch to working entirely with Gram matrices

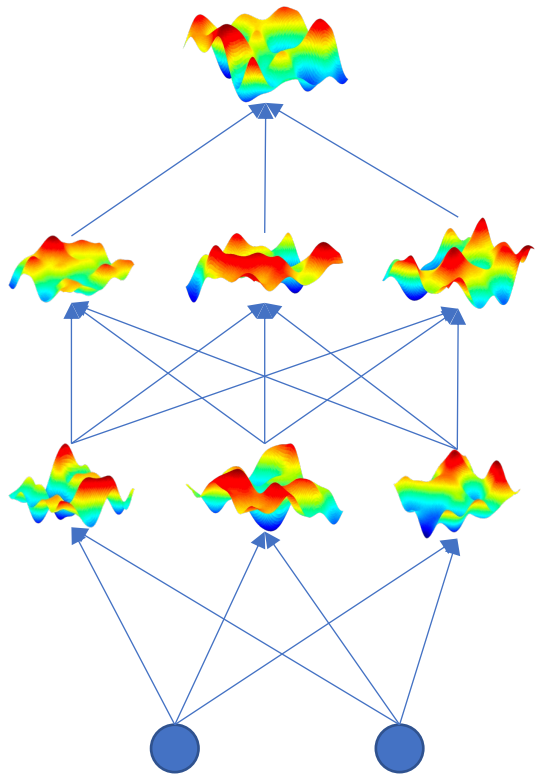
P = number of datapoints
N_ℓ = width of layer ℓ

Gram matrices

$$\underset{P\times P}{\boldsymbol{G}_2} = \boldsymbol{F}_2\boldsymbol{F}_2^T/N_2$$

$$\underset{P\times P}{\boldsymbol{G}_1} = \boldsymbol{F}_1\boldsymbol{F}_1^T/N_1$$

$$\underset{P\times P}{\boldsymbol{G}_0} = \boldsymbol{X}\boldsymbol{X}^T/N_\mathrm{X}$$



DGP

outputs,
$$\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \underset{P\times P}{\boldsymbol{K}_\mathrm{f}(\boldsymbol{F}_2)} + \sigma^2\boldsymbol{I})$$

hiddens,
$$\underset{P\times N_2}{\boldsymbol{F}_2} \sim \mathcal{N}(\mathbf{0}, \underset{P\times P}{\boldsymbol{K}_\mathrm{f}(\boldsymbol{F}_1)})$$

hiddens,
$$\underset{P\times N_1}{\boldsymbol{F}_1} \sim \mathcal{N}(\mathbf{0}, \underset{P\times P}{\boldsymbol{K}_\mathrm{f}(\boldsymbol{X})})$$

batch of
input vectors, $\underset{P\times N_X}{\boldsymbol{X}}$

GPs from Duvenaud et al. (2014)

Trick 1: most kernels of interest can be computed from the Gram matrix

- True for e.g. arccos kernels used in infinite NNs (Cho and Saul 2009)
- Also true for standard GP kernels that only depend on distance between datapoints i and j, because we can recover distance from the Gram matrix, (Duvenaud et al. 2014)

$$\begin{aligned} R_{ij}(\mathbf{G}) &= \tfrac{1}{N}\textstyle\sum_{\lambda=1}^{N}\left(F_{i\lambda} - F_{j\lambda}\right)^2 \\ &= \tfrac{1}{N}\textstyle\sum_{\lambda=1}^{N}\left(\left(F_{i\lambda}\right)^2 - 2F_{i\lambda}F_{j\lambda} + \left(F_{j\lambda}\right)^2\right) \\ &= G_{ii} - 2G_{ij} + G_{jj} \end{aligned}$$

- Overall:

$$\boldsymbol{K}_f(\boldsymbol{F}_\ell) = \boldsymbol{K}(\boldsymbol{G}_\ell)$$

Trick 2: Gram matrices are Wishart distributed

To get next Gram matrix, we first sample a bunch of features,

$$\boldsymbol{F}_\ell \sim \mathcal{N}\left(\mathbf{0}, \boldsymbol{K}(\boldsymbol{G}_{\ell-1})\right)$$

And then compute the Gram matrix

$$\boldsymbol{G}_\ell = \frac{1}{N_\ell}\boldsymbol{F}_\ell \boldsymbol{F}_\ell^T$$

But this exactly matches the definition of the Wishart distribution!

$$\boldsymbol{G}_\ell \sim \mathcal{W}(\boldsymbol{K}(\boldsymbol{G}_{\ell-1})/N_\ell, N_\ell)$$

(see Wikipedia for pdf, moments etc.)

In deep-kernel methods, we switch to working entirely with Gram matrices

P = number of datapoints
N_ℓ = width of layer ℓ

DKP | **Gram matrices** | **DGP**

outputs,
$\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{K}(\boldsymbol{G}_2) + \sigma^2 \boldsymbol{I})$

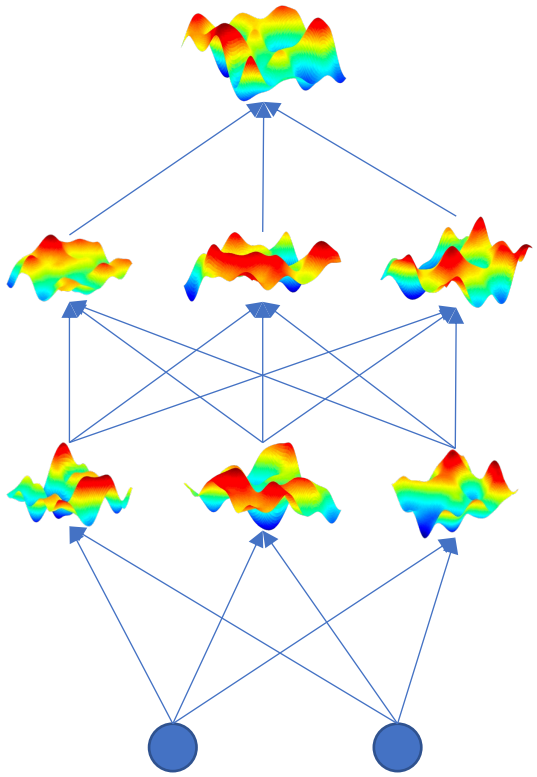
Trick 1: Kernel can be written as a function of the Gram matrix

$\boldsymbol{G}_2 \sim \mathcal{W}(\boldsymbol{K}(\boldsymbol{G}_1)/N_2, N_2)$ | $\underset{P\times P}{\boldsymbol{G}_2} = \boldsymbol{F}_2\boldsymbol{F}_2^T/N_2$

Trick 2: Gram matrices are Wishart distributed

$\boldsymbol{G}_1 \sim \mathcal{W}(\boldsymbol{K}(\boldsymbol{G}_0)/N_1, N_1)$ | $\underset{P\times P}{\boldsymbol{G}_1} = \boldsymbol{F}_1\boldsymbol{F}_1^T/N_1$

$\boldsymbol{G}_0 = \boldsymbol{X}\boldsymbol{X}^T/N_X$ | $\underset{P\times P}{\boldsymbol{G}_0} = \boldsymbol{X}\boldsymbol{X}^T/N_X$



outputs,
$\mathbf{y} \sim \mathcal{N}(\mathbf{0}, \underset{P\times P}{\boldsymbol{K}_f(\boldsymbol{F}_2)} + \sigma^2 \boldsymbol{I})$

hiddens,
$\underset{P\times N_2}{\boldsymbol{F}_2} \sim \mathcal{N}(\mathbf{0}, \underset{P\times P}{\boldsymbol{K}_f(\boldsymbol{F}_1)})$

hiddens,
$\underset{P\times N_1}{\boldsymbol{F}_1} \sim \mathcal{N}(\mathbf{0}, \underset{P\times P}{\boldsymbol{K}_f(\boldsymbol{X})})$

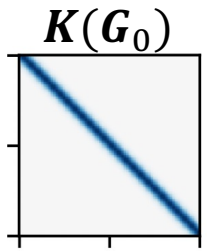
batch of input vectors, $\underset{P\times N_X}{\boldsymbol{X}}$

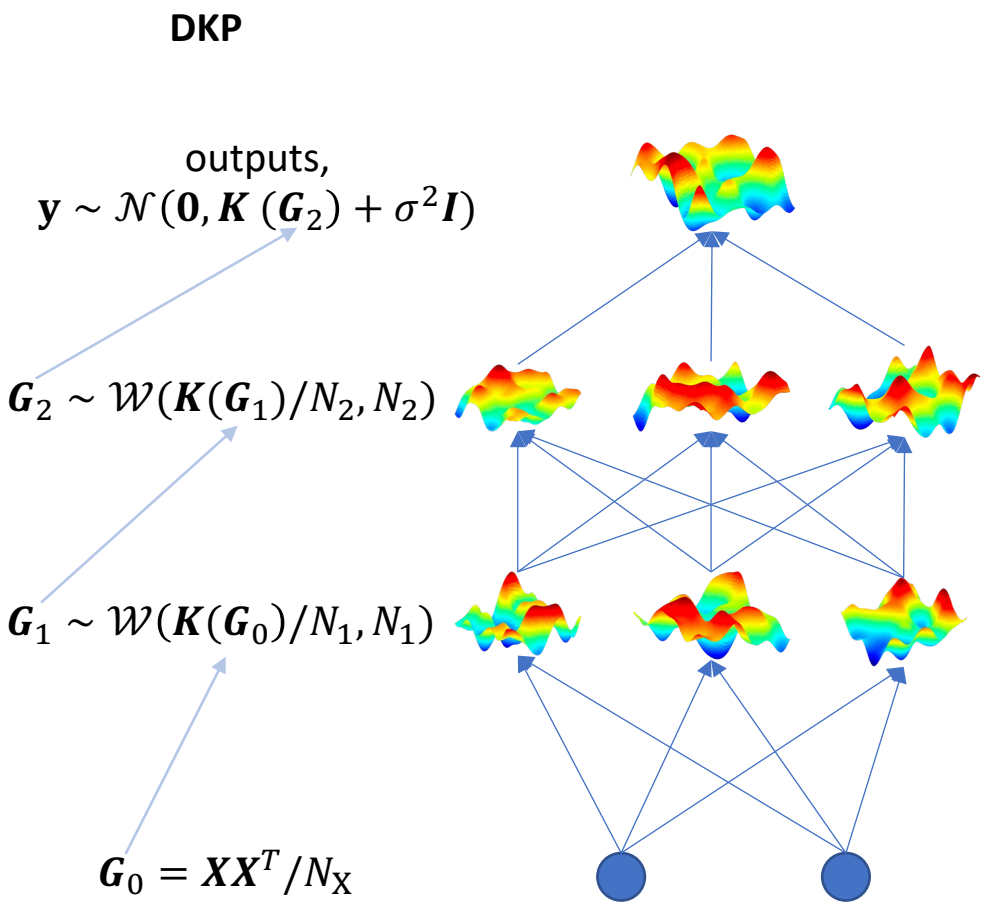
GPs from Duvenaud et al. (2014)

Sampling the prior in the kernelized DGP

- Next Gram matrix is "centered" on the kernel,

$$E[\boldsymbol{G}_1|\boldsymbol{G}_0] = \boldsymbol{K}(\boldsymbol{G}_0)$$





Developing practical methods + our results

We developed:

- Two processes: “deep Wishart process” and “deep *inverse* Wishart process”
- VI with priors + approximate posteriors over Gram matrices, *not* features.
- a bunch of approximate posteriors (e.g. $Q_{\mathcal{GW}}$)

	Dataset	DGP	$Q_{\mathcal{GW}}$	DWP $Q_{\text{A-}\mathcal{GW}}$	DWP $Q_{\text{AB-}\mathcal{GW}}$
LL	BOSTON	-2.43 ± 0.04	**-2.38 ± 0.04**	-2.39 ± 0.05	**-2.38 ± 0.04**
	CONCRETE	-3.13 ± 0.02	-3.13 ± 0.02	**-3.07 ± 0.02**	-3.08 ± 0.02
	ENERGY	-0.71 ± 0.03	-0.71 ± 0.03	**-0.70 ± 0.03**	**-0.70 ± 0.03**
	KIN8NM	1.38 ± 0.00	1.40 ± 0.01	**1.41 ± 0.01**	**1.41 ± 0.01**
	NAVAL	8.28 ± 0.04	8.17 ± 0.07	**8.40 ± 0.02**	8.10 ± 0.19
	POWER	-2.78 ± 0.01	-2.77 ± 0.01	**-2.76 ± 0.01**	**-2.76 ± 0.01**
	PROTEIN	-2.73 ± 0.01	-2.72 ± 0.01	-2.71 ± 0.01	**-2.70 ± 0.00**
	WINE	-0.96 ± 0.01	-0.96 ± 0.01	-0.96 ± 0.01	-0.96 ± 0.01
	YACHT	-0.73 ± 0.07	-0.58 ± 0.06	-0.22 ± 0.09	**-0.18 ± 0.07**

[1] Aitchison, Yang and Ober. “*Deep kernel processes*” ICML (2021)

[2] Ober and Aitchison “*An approximate posterior for the deep Wishart process*” NeurIPS (2021)

[3] Ober, Anson, Milsom and Aitchison “*An improved approximate posterior for the deep Wishart process*” UAI (2023)

Part 1

rewrite prior in terms of kernel/Gram matrices

deep Gaussian processes

Infinite-width limit of Bayesian method

deep kernel **processes**

deep kernel **machines**

Part 2

rewrite prior in terms of kernel/Gram matrices

deep Gaussian processes

Infinite-width limit of Bayesian method

deep kernel **processes**

deep kernel **machines**

Why **less(?) Bayesian** deep kernel **machines**?

Less(?) Bayesian approach:

- simplifies implementation
- gives lower-variance updates that converge faster
- provides a cleaner link to NN / neuro-theory
- great preliminary results…

We get a deep kernel machine by taking an infinite-width limit of a DGP

- True posterior over features becomes multivariate Gaussian [1]

$$P(\boldsymbol{F}_1, \ldots, \boldsymbol{F}_L | \boldsymbol{X}, \boldsymbol{Y}) = \prod_{\ell=1}^{L} \sum_{\lambda=1}^{N_\ell} \mathcal{N}(\boldsymbol{f}_\lambda^\ell; 0, \boldsymbol{G}_\ell^*)$$

- We choose a family of approximate posteriors capturing the true posterior:

$$Q(\boldsymbol{F}_1, \ldots, \boldsymbol{F}_L) = \prod_{\ell=1}^{L} \sum_{\lambda=1}^{N_\ell} \mathcal{N}\left(\boldsymbol{f}_\lambda^\ell; 0, \boldsymbol{G}_\ell\right)$$

- Gram matrices, $\boldsymbol{G}_1, \ldots \boldsymbol{G}_L$, are the same kind of thing as in deep kernel process!

$$\boldsymbol{G}_\ell = \frac{1}{N_\ell} \boldsymbol{F}_\ell \boldsymbol{F}_\ell^T$$

- But here, Gram matrices appear as parameters of approximate posterior
- So to find the Gram matrices, we optimize the ELBO!

[1] Yang, Robeyns, Milsom, Anson, Schoots, Aitchison “A theory of representation learning gives a deep generalisation of kernel methods” ICML (2023)

A DKM is an infinite-width limit of a DGP!

ELBO:

$$\mathcal{L}(\boldsymbol{G}_1, \ldots, \boldsymbol{G}_L) = \underbrace{\log \mathrm{P}(\mathbf{Y}|\, \boldsymbol{G}_L)}_{\text{likelihood}} - \sum_{\ell=1}^{L} \nu_\ell D_{\mathrm{KL}}(\underbrace{\mathcal{N}(0, \boldsymbol{G}_\ell)}_{\text{approx post}} \| \underbrace{\mathcal{N}(0, \boldsymbol{K}(\boldsymbol{G}_{\ell-1}))}_{\text{prior}})$$

- Optimizes intermediate layer Gram matrices,
- Encourages good performance
- Keeps approximate posterior covariance, $\boldsymbol{G}_\ell$, similar to prior covariance, $\boldsymbol{K}(\boldsymbol{G}_{\ell-1})$

The objective only talks about $P \times P$ Gram matrices, $\boldsymbol{G}_\ell$, and kernel matrices, $\boldsymbol{K}(\boldsymbol{G}_{\ell-1})$. So it is a deep kernel method!

What is a deep kernel machine?

- A nonlinear function approximator
- With multiple layers
- Parameterised by *Gram matrices*, not features or weights
- Trained using the DKM objective:

$$\mathcal{L}(\boldsymbol{G}_1, \ldots, \boldsymbol{G}_L) = \underbrace{\log \mathrm{P}(\mathbf{Y}|\, \boldsymbol{G}_L)}_{\text{likelihood}} - \sum_{\ell=1}^{L} \nu_\ell D_{\mathrm{KL}}(\underbrace{\mathcal{N}(0, \boldsymbol{G}_\ell)}_{\text{approx post}} \| \underbrace{\mathcal{N}(0, \boldsymbol{K}(\boldsymbol{G}_{\ell-1}))}_{\text{prior}})$$

- Optimizes intermediate layer Gram matrices,
- Encourages good performance
- Keeps approximate posterior covariance, $\boldsymbol{G}_\ell$, similar to prior covariance, $\boldsymbol{K}(\boldsymbol{G}_{\ell-1})$

Preliminary results for convolutional deep kernel machines

Table 3: Comparison of test accuracy against other kernel methods, including NNGP and NTK.

Paper	Method	CIFAR-10
This paper	DKM-DA-GAP (512 / 1024 / 2048)	91.98%
Novak et al. (2018)	NNGP-GAP	77.43%
Arora et al. (2019)	NNGP-GAP	83.75%
Lee et al. (2020)	NNGP-GAP-DA	84.8%
Li et al. (2019)	NNGP-LAP-flip	88.92%
Shankar et al. (2020)	Myrtle10	89.80%
Adlam et al. (2023)	Tuned Myrtle10 DA CG	91.2%

Edward Milsom

But how slow are DKMs? Surprisingly fast!

- We develop a novel inducing-point scheme
- Same FLOPs as CNN (computations ultimately look v. similar)
- Slower than a CNN (about 30 hours for largest model), as we're using float64 FLOPs
- Orders of magnitude faster than "full" kernel methods above.

Summary

rewrite prior in terms of kernel/Gram matrices

deep Gaussian processes

deep kernel **processes**

deep kernel **machines**

Infinite-width limit of Bayesian method

Deep kernel landscape + our priorities

Our priorities

- More architectures for DKMs (GNNs + transformers).
- Understanding Bayesian-ness of DKMs
- speed/scale-up:
  - memory efficiency
  - lower-precision
- user-friendly library (we can share preliminary work)

Huge future opportunities:

	shallow	deep
feature	linear regression	neural net
kernel	kernel ridge regression	**deep kernel methods**

If you're interested, get in touch:

laurence.aitchison@bristol.ac.uk

[1] Aitchison, Yang and Ober. "*Deep kernel processes*" ICML (2021)

[2] Ober and Aitchison "*An approximate posterior for the deep Wishart process*" NeurIPS (2021)

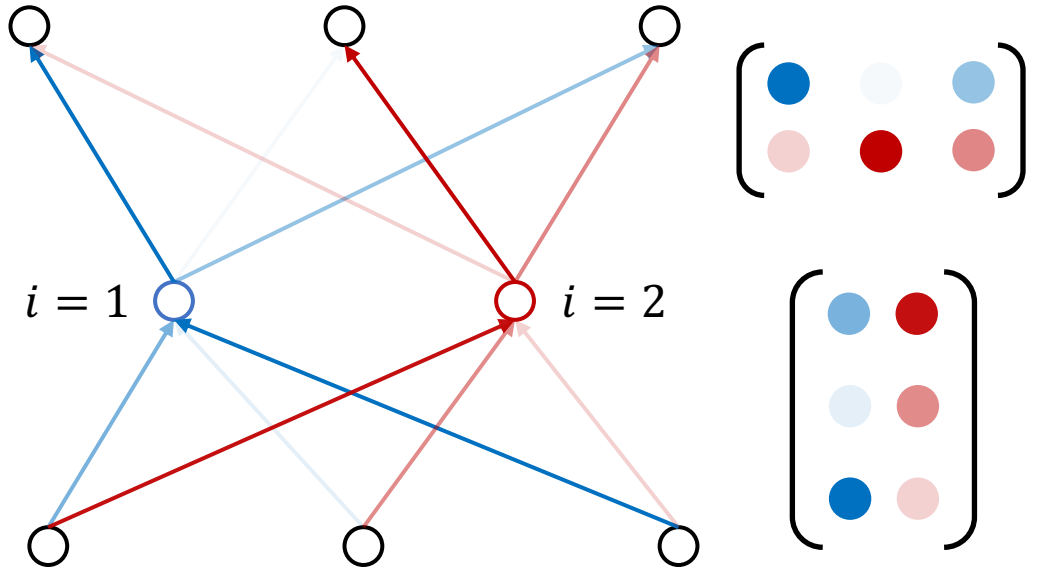
[3] Ober, Anson, Milsom and Aitchison "*An improved approximate posterior for the deep Wishart process*" UAI (2023)

[4] Yang, Robeyns, Milsom, Anson, Schoots, Aitchison "A theory of representation learning gives a deep generalisation of kernel methods" ICML (2023)

[5] Milsom, Anson, Aitchison "Convolutional deep kernel machines" (in prep)

Appendix slides

Deep kernel processes should work better because they have fewer local optima



Deep kernel processes should work better because they have fewer local optima

- Implies loads of symmetric local optima…
- …and local optima are bad if you have unimodal approximate posteriors.
- DKPs don't have these symmetries, so *far* fewer local optima!