Neural Tangent Kernels for Equivariant Neural Networks

Talk at the GeUmetric Deep Learning Workshop @Umeå

Philipp Misof

Department of Mathematical Sciences, Division of Algebra and Geometry

June 12, 2024

UNIVERSITY OF GOTHENBURG

CHALMERS UNIVERSITY OF TECHNOLOGY

WASP WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM

Joint work with

Jan Gerken

Pan Kessel

arXiv:2406.06504

Outline

Outline

Outline

Outline

Outline

Outline

Neural Tangent Kernel (NTK) encodes training dynamics

Neural Tangent Kernel (NTK) encodes training dynamics

- Assuming **Gradient Flow** of a NN $\mathcal{N} : \mathbb{R}^{n_{\text{in}}} \to \mathbb{R}^{n_{\text{out}}}$

$$\dot{\mathcal{N}}(x) = -\eta \sum_{i=1}^{n_{\text{train}}} \Theta_t(x, x_i) \frac{\partial \mathcal{L}}{\partial \mathcal{N}(x_i)}$$

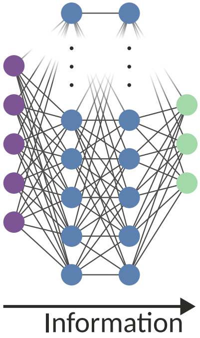
  - $\mathcal{N} \ldots$ Neural Network (NN)
  - $\Theta_t(x, x_i) \ldots$ NTK
  - $\mathcal{L} \ldots$ loss function
  - $\eta \ldots$ learning rate

Neural Tangent Kernel (NTK) encodes training dynamics

- Assuming **Gradient Flow** of a NN $\mathcal{N} : \mathbb{R}^{n_{\text{in}}} \to \mathbb{R}^{n_{\text{out}}}$

$$\dot{\mathcal{N}}(x) = -\eta \sum_{i=1}^{n_{\text{train}}} \Theta_t(x, x_i) \frac{\partial \mathcal{L}}{\partial \mathcal{N}(x_i)}$$

  - $\mathcal{N}$ . . . Neural Network (NN)
  - $\Theta_t(x, x_i)$ . . . NTK
  - $\mathcal{L}$ . . . loss function
  - η . . . learning rate



- **Infinite width limit** $\to \Theta_t(x, x_i)$ becomes
  - deterministic
  - time-independent
- Combine with **MSE loss** $\to$ linear ODE with **analytical solution**

What do we gain?

What do we gain?

- Analytical **study of training** independent of particular initialization
  - Influence of **hyperparameters** *(Geiger et al. 2020)*
  - Impact of **data augmentation** *(Gerken and Kessel 2024)*
  - Study of spectral **bias** of specific architectures and data distributions *(Bowman and Montufar 2022)*

What do we gain?

- Analytical **study of training** independent of particular initialization
  - Influence of **hyperparameters** *(Geiger et al. 2020)*
  - Impact of **data augmentation** *(Gerken and Kessel 2024)*
  - Study of spectral **bias** of specific architectures and data distributions *(Bowman and Montufar 2022)*
- New **kernel method** performant on small datasets *(Arora et al. 2019)*

What do we gain?

- Analytical **study of training** independent of particular initialization
  - Influence of **hyperparameters** *(Geiger et al. 2020)*
  - Impact of **data augmentation** *(Gerken and Kessel 2024)*
  - Study of spectral **bias** of specific architectures and data distributions *(Bowman and Montufar 2022)*
- New **kernel method** performant on small datasets *(Arora et al. 2019)*

$\longrightarrow$ Let's extend this to **equivariant neural networks**!

The empirical NTK

Empirical NTK

$$\Theta_t(x, x_i) = \frac{\partial \mathcal{N}(x)}{\partial \theta} \left(\frac{\partial \mathcal{N}(x_i)}{\partial \theta} \right)^{\mathsf{T}}$$

The empirical NTK

Empirical NTK

$$\Theta_t(x, x_i) = \frac{\partial \mathcal{N}(x)}{\partial \theta} \left(\frac{\partial \mathcal{N}(x_i)}{\partial \theta} \right)^{\mathsf{T}}$$

- In general **complicated**

The empirical NTK

Empirical NTK

$$\Theta_t(x, x_i) = \frac{\partial \mathcal{N}(x)}{\partial \theta} \left(\frac{\partial \mathcal{N}(x_i)}{\partial \theta} \right)^{\mathsf{T}}$$

- In general **complicated**
- Depends on particular **initialization** $\theta(0)$

The empirical NTK

Empirical NTK

$$\Theta_t(x, x_i) = \frac{\partial \mathcal{N}(x)}{\partial \theta} \left(\frac{\partial \mathcal{N}(x_i)}{\partial \theta} \right)^{\mathsf{T}}$$

- In general **complicated**
- Depends on particular **initialization** $\theta(0)$
- **time-dependent**

NNs become Gaussian processes

- At $t = 0$, all parameters $\theta_i(0)$ are *iid*
- **law of large numbers**

$$\frac{1}{n}\sum_{i=1}^{n} X_i \to \mathbb{E}[X], \qquad X_i \text{ iid}$$

NNs become Gaussian processes

- At $t = 0$, all parameters $\theta_i(0)$ are *iid*
- **law of large numbers**

$$\frac{1}{n}\sum_{i=1}^{n} X_i \to \mathbb{E}[X], \qquad X_i \text{ iid}$$

Neural Network Gaussian Process (Lee et al. 2018)

Gaussian process with zero mean and deterministic covariance

$$K(x, x') = \mathbb{E}[\mathcal{N}(x)\mathcal{N}(x')^{\mathsf{T}}]$$

. . . **NNGP kernel**

The Limit NTK

The Limit NTK

Limit NTK *(Jacot, Gabriel, and Hongler 2018)*

The empircal NTK converges to a deterministic kernel $\Theta_0(x, x') \stackrel{\text{prob}}{\to} \Theta(x, x')$

The Limit NTK

Limit NTK *(Jacot, Gabriel, and Hongler 2018)*

The empircal NTK converges to a deterministic kernel $\Theta_0(x, x') \overset{\text{prob}}{\rightarrow} \Theta(x, x')$

- Only dependent on distribution of parameters θ_0

The Limit NTK

Limit NTK *(Jacot, Gabriel, and Hongler 2018)*

The empircal NTK converges to a deterministic kernel $\Theta_0(x, x') \overset{\text{prob}}{\to} \Theta(x, x')$

- Only dependent on distribution of parameters θ_0
- Can be computed recursively

$$K^{(\ell+1)} = f(K^{(\ell)})$$
$$\Theta^{(\ell+1)} = g(\Theta^{(\ell)}, K^{(\ell+1)}, \dot{K}^{(\ell+1)})$$

where

$$\dot{K}^{(\ell+1)}(x, x') = \mathbb{E}[\sigma'(\mathcal{N}^{(\ell)}(x))\sigma'(\mathcal{N}^{(\ell)}(x'))],$$

$\sigma \ldots$ non-linearity

The limit NTK is frozen

= time-independent

The limit NTK is frozen

= time-independent

if non-linearity σ is Lipschitz, $\mathcal{C}^2$ and has bounded second derivative

Frozen NTK *(Jacot, Gabriel, and Hongler 2018)*

$$\Theta_t(x, x') \to \Theta(x, x')$$

uniformly in t when taking the limit layer by layer

The limit NTK is frozen

= time-independent

if non-linearity σ is Lipschitz, $\mathcal{C}^2$ and has bounded second derivative

Frozen NTK *(Jacot, Gabriel, and Hongler 2018)*

$$\Theta_t(x, x') \to \Theta(x, x')$$

uniformly in t when taking the limit layer by layer

- Greatly simplifies ODE!

The limit NTK is frozen

= time-independent

if non-linearity σ is Lipschitz, $\mathcal{C}^2$ and has bounded second derivative

Frozen NTK *(Jacot, Gabriel, and Hongler 2018)*

$$\Theta_t(x, x') \to \Theta(x, x')$$

uniformly in t when taking the limit layer by layer

- Greatly simplifies ODE!
- MSE loss $\to$ linear ODE with analytical solution

NTK as a kernel method

NTK as a kernel method

Kernel $\Phi(x, x') : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ similarity measure
Kernel regression:

$$f(x) = \alpha(\mathcal{Y})\Phi(x, \mathcal{X})$$

with $(\mathcal{X}, \mathcal{Y}) \ldots$ whole training set, $\alpha(\mathcal{X}, \mathcal{Y}) \ldots$ coefficients of the regression

NTK as a kernel method

Kernel $\Phi(x, x') : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ similarity measure
Kernel regression:

$$f(x) = \alpha(\mathcal{Y})\Phi(x, \mathcal{X})$$

with $(\mathcal{X}, \mathcal{Y}) \ldots$ whole training set, $\alpha(\mathcal{X}, \mathcal{Y}) \ldots$ coefficients of the regression

Take **infinite width infinite time limit** and **MSE** loss $\to$ Mean of Gaussian process is

$$\mu(x) = \Theta(x, \mathcal{X})\Theta(\mathcal{X}, \mathcal{X})^{-1}\mathcal{Y}$$

$\to$ NTK yields kernel method

Ingredients for Equivariant Neural Networks

Equivariance under group G

$$\mathcal{N}(\rho_{\text{reg}}(g)f) = \rho_{\text{reg}}(g)\mathcal{N}(f)\,, \qquad g \in G$$

$\rightarrow$ **group convolutional neural networks** (GCNN)

Ingredients for Equivariant Neural Networks

Equivariance under group G

$$\mathcal{N}(\rho_{\text{reg}}(g)f) = \rho_{\text{reg}}(g)\mathcal{N}(f)\,, \qquad g \in G$$

$\rightarrow$ **group convolutional neural networks** (GCNN)

- Group convolutional layers
- Lifting layer
- elementwise non-linearity σ
- group pooling layer (to achieve **invariance**)

$\rightarrow$ Need recursive relations for the NTK for all of them

Group Convolution (GCNN)

- Data given in form of feature maps $f : X \to \mathbb{R}^{n_{\text{in}}}$
- For grey-scale images: $f : \mathbb{Z}^2 \to \mathbb{R}$

Group Convolution (GCNN)

- Data given in form of feature maps $f : X \to \mathbb{R}^{n_{\text{in}}}$
- For grey-scale images: $f : \mathbb{Z}^2 \to \mathbb{R}$
- group convolutional **lifting layer**

$$[\mathcal{N}^{(1)}(f)](g) = [\kappa * f](g) = \frac{1}{\sqrt{n_{\text{in}} \text{vol}(S_\kappa)}} \int_X \mathrm{d}x \, \kappa\big(\rho(g^{-1})x\big) f(x)$$

$\kappa : X \to \mathbb{R}^{n_{\text{in}} \times n_1}$ . . . filter with support S_κ

Group Convolution (GCNN)

- Data given in form of feature maps $f : X \to \mathbb{R}^{n_{\text{in}}}$
- For grey-scale images: $f : \mathbb{Z}^2 \to \mathbb{R}$
- group convolutional **lifting layer**

$$[\mathcal{N}^{(1)}(f)](g) = [\kappa * f](g) = \frac{1}{\sqrt{n_{\text{in}} \text{vol}(S_\kappa)}} \int_X \mathrm{d}x \, \kappa\big(\rho(g^{-1})x\big) f(x)$$

$\kappa : X \to \mathbb{R}^{n_{\text{in}} \times n_1}$ ... filter with support S_κ

- consecutive **group convolutional layer**

$$[\mathcal{N}^{(\ell+1)}(f)](g) = [\kappa * \mathcal{N}^{(\ell)}(f)](g) = \frac{1}{\sqrt{n_\ell \text{vol}(S_\kappa)}} \int_G \mathrm{d}h \; \kappa\big(g^{-1}h\big) [\mathcal{N}^{(\ell)}(f)](h)$$

$\kappa : G \to \mathbb{R}^{n_\ell \times n_{\ell+1}}$ ... filter

Group Convolution (GCNN)

- Data given in form of feature maps $f : X \to \mathbb{R}^{n_{\text{in}}}$
- For grey-scale images: $f : \mathbb{Z}^2 \to \mathbb{R}$
- group convolutional **lifting layer**

$$[\mathcal{N}^{(1)}(f)](g) = [\kappa * f](g) = \frac{1}{\sqrt{n_{\text{in}}\mathrm{vol}(S_\kappa)}} \int_X \mathrm{d}x \, \kappa\big(\rho(g^{-1})x\big) f(x)$$

$\kappa : X \to \mathbb{R}^{n_{\text{in}} \times n_1}$ ... filter with support S_κ

- consecutive **group convolutional layer**

$$[\mathcal{N}^{(\ell+1)}(f)](g) = [\kappa * \mathcal{N}^{(\ell)}(f)](g) = \frac{1}{\sqrt{n_\ell \mathrm{vol}(S_\kappa)}} \int_G \mathrm{d}h \; \kappa\big(g^{-1}h\big) [\mathcal{N}^{(\ell)}(f)](h)$$

$\kappa : G \to \mathbb{R}^{n_\ell \times n_{\ell+1}}$ ... filter

- **infinite width limit:** channels $n_1, \ldots n_{L-1} \to \infty$

Equivariant NTK

GCNN layers

At each layer ℓ

$$K^{(\ell)}_{g,g'}(f,f') = \mathbb{E}\Big[[\mathcal{N}^{(\ell)}(f)](g)\ [\mathcal{N}^{(\ell)}(f')](g')^{\mathsf{T}}\Big]$$

Equivariant NTK

GCNN layers

At each layer ℓ

$$K^{(\ell)}_{g,g'}(f,f') = \mathbb{E}\Big[[\mathcal{N}^{(\ell)}(f)](g)\ [\mathcal{N}^{(\ell)}(f')](g')^\intercal\Big]$$

Recursion for GCNN layer

$$K^{(\ell+1)}_{g,g'}(f,f') = \frac{1}{\mathrm{vol}(S_\kappa)}\int_{S_\kappa} \mathrm{d}h\ K^{(\ell)}_{gh,g'h}(f,f')$$

$$\Theta^{(\ell+1)}_{g,g'}(f,f') = K^{(\ell+1)}_{g,g'}(f,f') + \frac{1}{\mathrm{vol}(S_\kappa)}\int_{S_\kappa} \mathrm{d}h\ \Theta^{(\ell)}_{gh,g'h}(f,f')$$

Equivariant NTK

GCNN layers

Equivariant NTK

GCNN layers

Recursion for GCNN lifting layer

$$K^{(\ell+1)}_{g,g'}(f,f') = \frac{1}{\mathrm{vol}(S_\kappa)} \int_{S_\kappa} \mathrm{d}x\, K^{(\ell)}_{\rho(g)x,\,\rho(g')x'}(f,f')$$

$$\Theta^{(\ell+1)}_{g,g'}(f,f') = K^{(\ell+1)}_{g,g'}(f,f') + \frac{1}{\mathrm{vol}(S_\kappa)} \int_{S_\kappa} \mathrm{d}x\, \Theta^{(\ell)}_{\rho(g)x,\,\rho(g')x'}(f,f')$$

Equivalant NTK

GCNN layers

Recursion for GCNN lifting layer

$$K^{(\ell+1)}_{g,g'}(f,f') = \frac{1}{\mathrm{vol}(S_\kappa)} \int_{S_\kappa} \mathrm{d}x \; K^{(\ell)}_{\rho(g)x,\,\rho(g')x'}(f,f')$$

$$\Theta^{(\ell+1)}_{g,g'}(f,f') = K^{(\ell+1)}_{g,g'}(f,f') + \frac{1}{\mathrm{vol}(S_\kappa)} \int_{S_\kappa} \mathrm{d}x \; \Theta^{(\ell)}_{\rho(g)x,\,\rho(g')x'}(f,f')$$

If first layer $\rightarrow$

$$K^{(0)}_{x,x'}(f,f') = f(x)f'(x')\,, \qquad \Theta^{(0)}_{x,x'}(f,f') = 0$$

Equivariant NTK
Non-linearity

Elementwise non-linearity σ: $\mathcal{N}^{(\ell+1)}(g) = \sigma(\mathcal{N}^{(\ell)})(g)$

Equivariant NTK

Non-linearity

Elementwise non-linearity σ: $\mathcal{N}^{(\ell+1)}(g) = \sigma(\mathcal{N}^{(\ell)})(g)$

Recursion for non-linearity

$$\Lambda^{(\ell)}(f,f') = \begin{pmatrix} K^{(\ell)}(f,f) & K^{(\ell)}(f,f') \\ K^{(\ell)}(f',f) & K^{(\ell)}(f',f') \end{pmatrix}$$

$$K^{(\ell+1)}(f,f') = \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\Lambda^{(\ell)}(f,f'))}[\sigma(u)\sigma(v)]$$

$$\dot{K}^{(\ell+1)}(f,f') = \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\Lambda^{(\ell)}(f,f'))}[\sigma'(u)\sigma'(v)]$$

$$\Theta^{(\ell+1)}(f,f') = \dot{K}^{(\ell+1)}(f,f')\Theta^{(\ell)}(f,f')$$

Equivariant NTK

Non-linearity

Elementwise non-linearity σ: $\mathcal{N}^{(\ell+1)}(g) = \sigma(\mathcal{N}^{(\ell)})(g)$

Recursion for non-linearity

$$\Lambda^{(\ell)}_{g,g'}(f,f') = \begin{pmatrix} K^{(\ell)}_{g,g}(f,f) & K^{(\ell)}_{g,g'}(f,f') \\ K^{(\ell)}_{g',g}(f',f) & K^{(\ell)}_{g',g'}(f',f') \end{pmatrix}$$

$$K^{(\ell+1)}_{g,g'}(f,f') = \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\Lambda^{(\ell)}_{g,g'}(f,f'))}[\sigma(u)\sigma(v)]$$

$$\dot{K}^{(\ell+1)}_{g,g'}(f,f') = \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\Lambda^{(\ell)}_{g,g'}(f,f'))}[\sigma'(u)\sigma'(v)]$$

$$\Theta^{(\ell+1)}_{g,g'}(f,f') = \dot{K}^{(\ell+1)}_{g,g'}(f,f')\Theta^{(\ell)}_{g,g'}(f,f')$$

Equivariant NTK

Group pooling layer

Making equivariant network **invariant** $\rightarrow$ **group pooling**

$$\mathcal{N}^{(\ell+1)}(f) = \frac{1}{\sqrt{\mathrm{vol}(G)}} \int_G \mathrm{d}g \, [\mathcal{N}^{(\ell)}(f)](g)$$

Equivariant NTK

Group pooling layer

Making equivariant network **invariant** $\rightarrow$ **group pooling**

$$\mathcal{N}^{(\ell+1)}(f) = \frac{1}{\sqrt{\mathrm{vol}(G)}} \int_G \mathrm{d}g \, [\mathcal{N}^{(\ell)}(f)](g)$$

Recursion for group pooling layer

$$K^{(\ell+1)}(f,f') = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g \, \mathrm{d}g' \; K^{(\ell)}_{g,g'}(f,f')$$
$$\Theta^{(\ell+1)}(f,f') = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g \, \mathrm{d}g' \; \Theta^{(\ell)}_{g,g'}(f,f')$$

Symmetries of the NTK

$$K^{(\ell)}_{g,g'}(\rho_{\text{reg}}(h)f, \rho_{\text{reg}}(h')f') = K^{(\ell)}_{h^{-1}g,h'^{-1}g'}(f,f')$$

$$\Theta^{(\ell)}_{g,g'}(\rho_{\text{reg}}(h)f, \rho_{\text{reg}}(h')f') = \Theta^{(\ell)}_{h^{-1}g,h'^{-1}g'}(f,f')$$

$C_n \ltimes \mathbb{R}^2$

Discrete roto-translations in the plane

- $g = tr$, where $t \in \mathbb{R}^2$ and $r \in C_n$

$C_n \ltimes \mathbb{R}^2$

Discrete roto-translations in the plane

- $g = tr$, where $t \in \mathbb{R}^2$ and $r \in C_n$
- Keep t as spatial indices, for r extend to nn_ℓ channel indices in layer $\ell \rightarrow$

$$K_{g=tr,g'=t'r'}(f,f') = [K_{rr}(f,f')](t,t')\,, \quad \Theta_{g=tr,g'=t'r'}(f,f') = [\Theta_{rr'}(f,f')](t,t')$$

$C_n \ltimes \mathbb{R}^2$

Discrete roto-translations in the plane

- $g = tr$, where $t \in \mathbb{R}^2$ and $r \in C_n$
- Keep t as spatial indices, for r extend to nn_ℓ channel indices in layer $\ell \to$

$$K_{g=tr,g'=t'r'}(f,f') = [K_{rr}(f,f')](t,t')\,, \quad \Theta_{g=tr,g'=t'r'}(f,f') = [\Theta_{rr'}(f,f')](t,t')$$

- For CNN, `neural-tangents` library provides the following highly optimized operator

$$[\mathcal{A}_{S_\kappa}(K)](t,t') = \frac{1}{S_\kappa}\int_{S_\kappa} \mathrm{d}\tilde{t}\; K(t+\tilde{t}, t'+\tilde{t})$$

$C_n \ltimes \mathbb{R}^2$

Discrete roto-translations in the plane

- $g = tr$, where $t \in \mathbb{R}^2$ and $r \in C_n$
- Keep t as spatial indices, for r extend to nn_ℓ channel indices in layer $\ell \rightarrow$

$$K_{g=tr,g'=t'r'}(f,f') = [K_{rr}(f,f')](t,t')\,, \quad \Theta_{g=tr,g'=t'r'}(f,f') = [\Theta_{rr'}(f,f')](t,t')$$

- For CNN, `neural-tangents` library provides the following highly optimized operator

$$[\mathcal{A}_{S_\kappa}(K)](t,t') = \frac{1}{S_\kappa}\int_{S_\kappa} \mathrm{d}\tilde{t}\; K(t+\tilde{t}, t'+\tilde{t})$$

- Can this be exploited for $C_n \ltimes \mathbb{R}^2$ as well?

$C_n \ltimes \mathbb{R}^2$

Discrete roto-translations in the plane

$\rightarrow$ **Yes!**

$C_n \ltimes \mathbb{R}^2$

Discrete roto-translations in the plane

$\rightarrow$ **Yes!**

$$[K^{(\ell+1)}_{rr'}(f,f')](t,t') = \sum_{\tilde{r}\in C_n}[\mathcal{A}_{\rho(r)S_\kappa}(\tilde{K}^{(\ell)}_{r\tilde{r},\,r'\tilde{r}}(f,f'))](t,\rho(rr'^{-1})t')$$

$$[\Theta^{(\ell+1)}_{rr'}(f,f')](t,t') = [K^{(\ell+1)}_{r\,r'}(f,f')](t,t') + \sum_{\tilde{r}\in C_n}[\mathcal{A}_{\rho(r)\S_\kappa}(\tilde{\Theta}^{(\ell)}_{r\tilde{r},\,r'\tilde{r}}(f,f'))](t,\rho(rr'^{-1})t')\,,$$

$C_n \ltimes \mathbb{R}^2$

Discrete roto-translations in the plane

$\rightarrow$ **Yes!**

$$[K^{(\ell+1)}_{rr'}(f,f')](t,t') = \sum_{\tilde{r}\in C_n} [\mathcal{A}_{\rho(r)S_\kappa}(\tilde{K}^{(\ell)}_{r\tilde{r},\,r'\tilde{r}}(f,f'))](t,\rho(rr'^{-1})t')$$

$$[\Theta^{(\ell+1)}_{rr'}(f,f')](t,t') = [K^{(\ell+1)}_{r\,r'}(f,f')](t,t') + \sum_{\tilde{r}\in C_n} [\mathcal{A}_{\rho(r)\S_\kappa}(\tilde{\Theta}^{(\ell)}_{r\tilde{r},\,r'\tilde{r}}(f,f'))](t,\rho(rr'^{-1})t')\,,$$

where

$$[\tilde{K}^{(\ell)}_{r\,r'}(f,f')](t,t') = [K^{(\ell)}_{r\,r'}(f,f')](t,\rho(r'r^{-1})t')$$

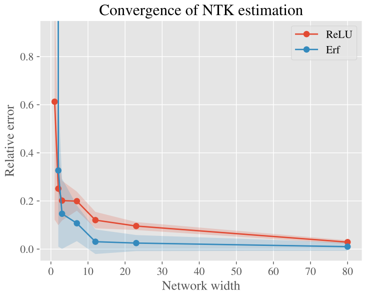
$$\tilde{\Theta}^{(\ell)}_{r\,r'}(f,f') = [\Theta^{(\ell)}_{r\,r'}(f,f')](t,\rho(r'r^{-1})t')$$

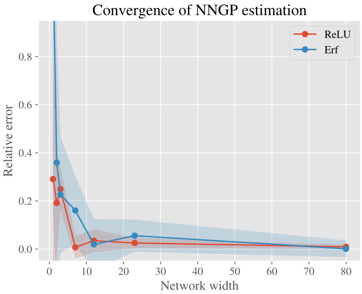
Kernel convergence

- Comparing limit kernels with mean of finite-width kernels
- $G = C_4 \ltimes \mathbb{R}^2$ and random input data
- Relative error averaged over all components of the Gram matrix
- Implemented in the `JAX`-based `neural-tangents` library[1]

Kernel convergence

- Comparing limit kernels with mean of finite-width kernels
- $G = C_4 \ltimes \mathbb{R}^2$ and random input data
- Relative error averaged over all components of the Gram matrix
- Implemented in the `JAX`-based `neural-tangents` library[1]





Histological image classification

The data set

NCT-CRC-HE-100K *(Kather, Halama, and Marx 2018)*

- $100\,000$ histological images of human colorectal cancer (CRC) and normal tissue
- 9 classes, 2 are cancerous
- 224×224 pixels in color

Histological image classification

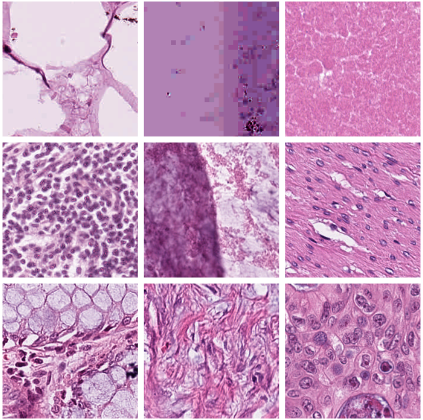
The data set

NCT-CRC-HE-100K *(Kather, Halama, and Marx 2018)*

- $100\,000$ histological images of human colorectal cancer (CRC) and normal tissue
- 9 classes, 2 are cancerous
- 224×224 pixels in color

Here:

- downsampled to 32×32 pixels
- Only up to 1000 training samples used



Histological image classification

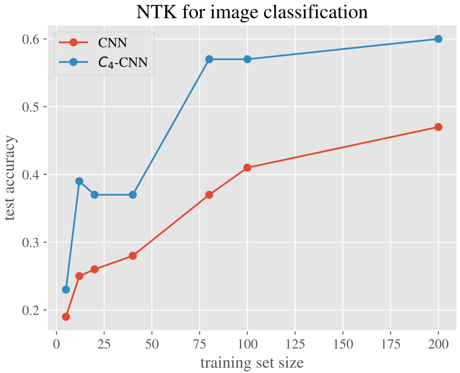
Performance for small training sets

- NTK kernel method
- Corresponds to training with MSE loss
- Comparison
  - **CNN**
  - $C_4 \ltimes \mathbb{R}^2$

Histological image classification

Performance for small training sets

- NTK kernel method
- Corresponds to training with MSE loss
- Comparison
  - **CNN**
  - $C_4 \ltimes \mathbb{R}^2$



Summary

Summary

- Derived **all necessary recursive relations** for the NTK of GCNNs

Summary

- Derived **all necessary recursive relations** for the NTK of GCNNs
- Enabled **analytical study** of training dynamics for equivariant neural networks in the infinite width limit

Summary

- Derived **all necessary recursive relations** for the NTK of GCNNs
- Enabled **analytical study** of training dynamics for equivariant neural networks in the infinite width limit

Summary

- Derived **all necessary recursive relations** for the NTK of GCNNs
- Enabled **analytical study** of training dynamics for equivariant neural networks in the infinite width limit

What's next?

- GCNNs over compact groups utilizing the Fourier domain

Summary

- Derived **all necessary recursive relations** for the NTK of GCNNs
- Enabled **analytical study** of training dynamics for equivariant neural networks in the infinite width limit

What's next?

- GCNNs over compact groups utilizing the Fourier domain
- equivariant graph NNs

Summary

- Derived **all necessary recursive relations** for the NTK of GCNNs
- Enabled **analytical study** of training dynamics for equivariant neural networks in the infinite width limit

What's next?

- GCNNs over compact groups utilizing the Fourier domain
- equivariant graph NNs
- Trainability and generalization regimes of equivariant NNs *(Xiao, Pennington, and Schoenholz 2020)*

Summary

- Derived **all necessary recursive relations** for the NTK of GCNNs
- Enabled **analytical study** of training dynamics for equivariant neural networks in the infinite width limit

What's next?

- GCNNs over compact groups utilizing the Fourier domain
- equivariant graph NNs
- Trainability and generalization regimes of equivariant NNs *(Xiao, Pennington, and Schoenholz 2020)*
- Relations to Quantum Field Theory *(Banta et al. 2023)*

Thanks for your attention!

arXiv:2406.06504