Emergent Equivariance in Deep Ensembles

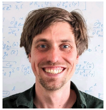
Jan E. Gerken

CHALMERS
UNIVERSITY OF TECHNOLOGY

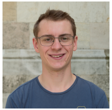
UNIVERSITY OF GOTHENBURG

WASP
WALLENBERG AI,
AUTONOMOUS SYSTEMS
AND SOFTWARE PROGRAM

in collaboration with



Pan Kessel



Philipp Misof

Data augmentation

👍 Easy to implement

👍 No specialized architecture necessary

Data augmentation

- 👍 Easy to implement
- 👍 No specialized architecture necessary
- 👎 No exact equivariance

Data augmentation

👍 Easy to implement

👍 No specialized architecture necessary

👎 No exact equivariance

Can we understand data augmentation theoretically?

Empirical NTK

Training dynamics under continuous gradient descent:

learning rate

loss

$$\frac{\mathrm{d}\mathcal{N}_{\theta}(x)}{\mathrm{d}t} = -\frac{\eta}{N}\sum_{i=1}^{N} \Theta_{\theta}(x, x_i)\frac{\partial L}{\partial \mathcal{N}(x_i)}$$

training sample

Empirical NTK

Training dynamics under continuous gradient descent:

learning rate

loss

$$\frac{\mathrm{d}\mathcal{N}_\theta(x)}{\mathrm{d}t} = -\frac{\eta}{N}\sum_{i=1}^{N} \Theta_\theta(x, x_i)\frac{\partial L}{\partial \mathcal{N}(x_i)}$$

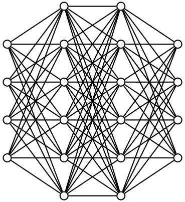
training sample

with the empirical neural tangent kernel (NTK)

$$\Theta_\theta(x, x') = \sum_{\mu} \frac{\partial \mathcal{N}(x)}{\partial \theta_\mu}\frac{\partial \mathcal{N}(x')}{\partial \theta_\mu}$$

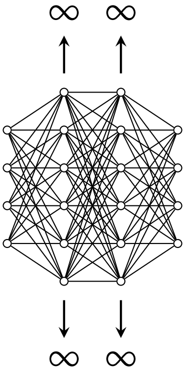
Infinite width limit

[Jacot et al. 2018]



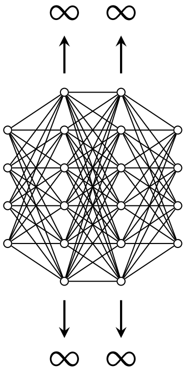
Infinite width limit

[Jacot et al. 2018]



Infinite width limit

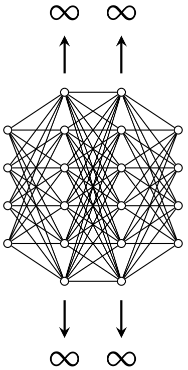
[Jacot et al. 2018]



👍 NTK becomes independent of initialization

Infinite width limit

[Jacot et al. 2018]

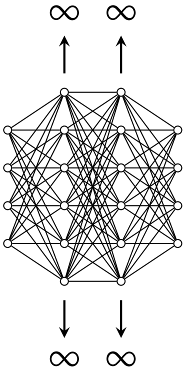


👍 NTK becomes independent of initialization

👍 NTK becomes constant in training

Infinite width limit

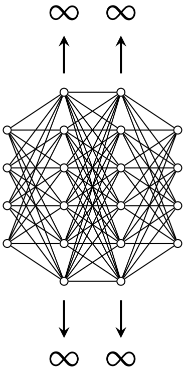
[Jacot et al. 2018]



- 👍 NTK becomes independent of initialization
- 👍 NTK becomes constant in training
- 👍 NTK can be computed for most networks

Infinite width limit

[Jacot et al. 2018]



- 👍 NTK becomes independent of initialization
- 👍 NTK becomes constant in training
- 👍 NTK can be computed for most networks
- ✓ Training dynamics can be solved

Mean prediction from NTK

[Jacot et al. 2018]

❗ At infinite width, the mean prediction is given by

$$\mu_t(x) = \Theta(x, X)\Theta(X, X)^{-1}(\mathbb{I} - e^{-\eta\Theta(X,X)t})Y$$

Mean prediction from NTK

[Jacot et al. 2018]

ⓘ At infinite width, the mean prediction is given by

neural tangent kernel

$$\mu_t(x) = \Theta(x, X)\Theta(X, X)^{-1}(\mathbb{I} - e^{-\eta\Theta(X,X)t})Y$$

Mean prediction from NTK

[Jacot et al. 2018]

At infinite width, the mean prediction is given by

neural tangent kernel

$$\mu_t(x) = \Theta(x, X)\Theta(X, X)^{-1}(\mathbb{I} - e^{-\eta\Theta(X,X)t})Y$$

train data

Mean prediction from NTK

[Jacot et al. 2018]

❗ At infinite width, the mean prediction is given by

neural tangent kernel

$$\mu_t(x) = \Theta(x, X)\Theta(X, X)^{-1}(\mathbb{I} - e^{-\eta\Theta(X,X)t})Y$$

learning rate

train data

Mean prediction from NTK

[Jacot et al. 2018]

At infinite width, the mean prediction is given by

neural tangent kernel

train labels

$$\mu_t(x) = \Theta(x, X)\Theta(X, X)^{-1}(\mathbb{I} - e^{-\eta\Theta(X,X)t})Y$$

learning rate

train data

Data augmentation

Data augmentation at infinite width

$$\mu_t(x) = \Theta(x, X)\Theta(X, X)^{-1}(\text{▯} - e^{-\eta\Theta(X,X)t})Y$$

Data augmentation at infinite width

$$\mu_t(x) = \Theta(x, X)\Theta(X, X)^{-1}(\mathbb{I} - e^{-\eta\Theta(X,X)t})Y$$

augmented data

augmented labels

Data augmentation at infinite width

group transformation

$$\mu_t(\rho(g)x) = \Theta(\rho(g)x, X)\Theta(X, X)^{-1}(\mathbb{I} - e^{-\eta\Theta(X,X)t})Y$$

augmented data

augmented labels

Data augmentation at infinite width

group transformation

for augmented data

$$\mu_t(\rho(g)x) = \Theta(\rho(g)x, X)\Theta(X, X)^{-1}(\mathbb{1} - e^{-\eta\Theta(X,X)t})Y$$

augmented data

augmented labels

Data augmentation at infinite width

group transformation

$$\mu_t(\rho(g)x) = \Theta(x, X)\Theta(X, X)^{-1}(\mathbb{I} - e^{-\eta\Theta(X,X)t})\rho(g)Y$$

augmented data

augmented labels

Data augmentation at infinite width

group transformation

augmented labels

$$\mu_t(\rho(g)x) = \Theta(x,X)\Theta(X,X)^{-1}(\mathbb{1} - e^{-\eta\Theta(X,X)t})\underbrace{\rho(g)Y}_{=Y}$$

for invariance

Data augmentation at infinite width

group transformation

$$\begin{aligned}\mu_t(\rho(g)x) &= \Theta(x,X)\Theta(X,X)^{-1}(\mathbb{I} - e^{-\eta\Theta(X,X)t})\underbrace{\rho(g)Y}_{=Y} \\ &= \mu_t(x)\end{aligned}$$

for invariance

Mean prediction

$\mu_t(x)$

Mean prediction

$$\mu_t(x) = \mathbb{E}_{\theta_0 \sim \text{initializations}}[\mathcal{N}_{\theta_t}(x)]$$

Mean prediction

$$\mu_t(x) = \mathbb{E}_{\theta_0 \sim \text{initializations}}[\mathcal{N}_{\theta_t}(x)] = \lim_{n\to\infty} \frac{1}{n} \sum_{\theta_0=\text{init}_1}^{\text{init}_n} \mathcal{N}_{\theta_t}(x)$$

Mean prediction

$$\mu_t(x) = \mathbb{E}_{\theta_0 \sim \text{initializations}}[\mathcal{N}_{\theta_t}(x)] = \lim_{n \to \infty} \underbrace{\frac{1}{n} \sum_{\theta_0 = \text{init}_1}^{\text{init}_n} \mathcal{N}_{\theta_t}(x)}_{\text{mean prediction of deep ensemble}}$$

Main conclusion

Deep ensembles trained with data augmentation are equivariant.

Main conclusion

Deep ensembles trained with data augmentation are equivariant.

✓ Proof of exact equivariance for

  - full data augmentation
  - infinite ensembles

Main conclusion

Deep ensembles trained with data augmentation are equivariant.

✓ Proof of exact equivariance for

  - full data augmentation
  - infinite ensembles

✓ Equivariance holds for all training times

Main conclusion

Deep ensembles trained with data augmentation are equivariant.

- ✓ Proof of exact equivariance for
  - full data augmentation
  - infinite ensembles
- ✓ Equivariance holds for all training times
- ✓ Equivariance holds away from the training data

Main conclusion

Deep ensembles trained with data augmentation are equivariant.

✓ Proof of exact equivariance for

- full data augmentation
- infinite ensembles

✓ Equivariance holds for all training times

✓ Equivariance holds away from the training data

✓ Holds also for finite-width networks

[Nordenfors, Flinth 2024]

Intuitive explanation

- ✓ Equivariance holds for all training times
- ✓ Equivariance holds away from the training data

Intuitive explanation

✓ Equivariance holds for all training times

✓ Equivariance holds away from the training data

ⓘ At infinite width, the mean output at initialization is zero everywhere.

Intuitive explanation

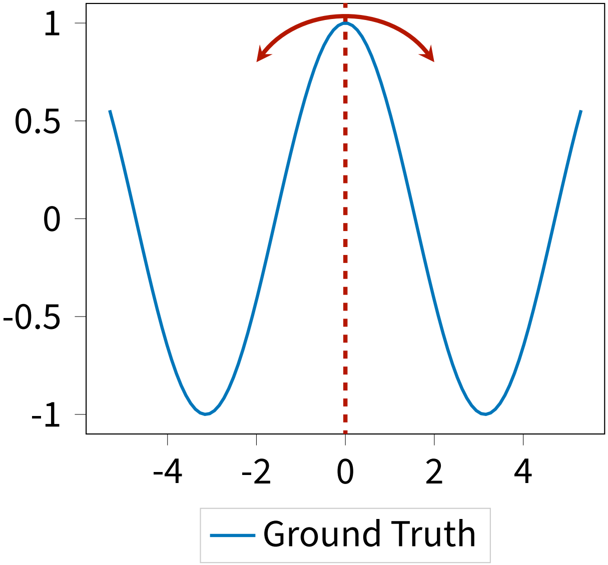
✓ Equivariance holds for all training times

✓ Equivariance holds away from the training data

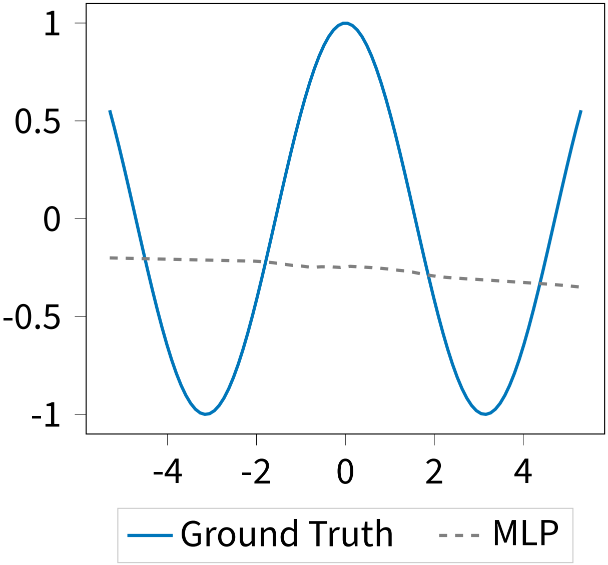
❗ At infinite width, the mean output at initialization is zero everywhere.

⇨ Training with full data augmentation leads to an equivariant function.

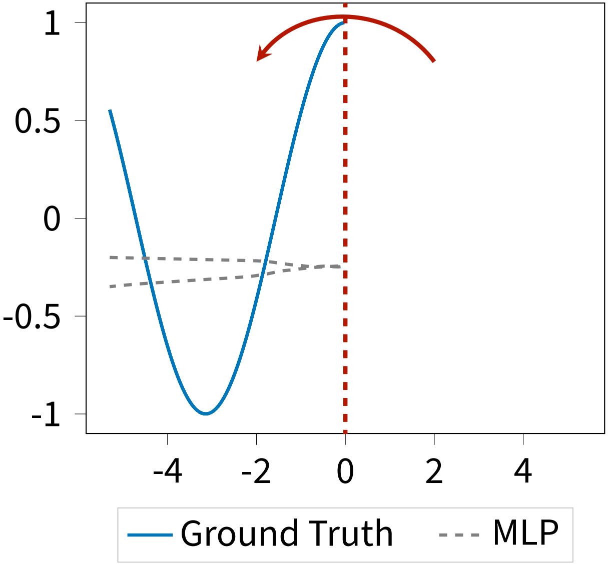
Toy example


1
0.5
0
-0.5
-1
-4
-2
0
2
4
Ground Truth

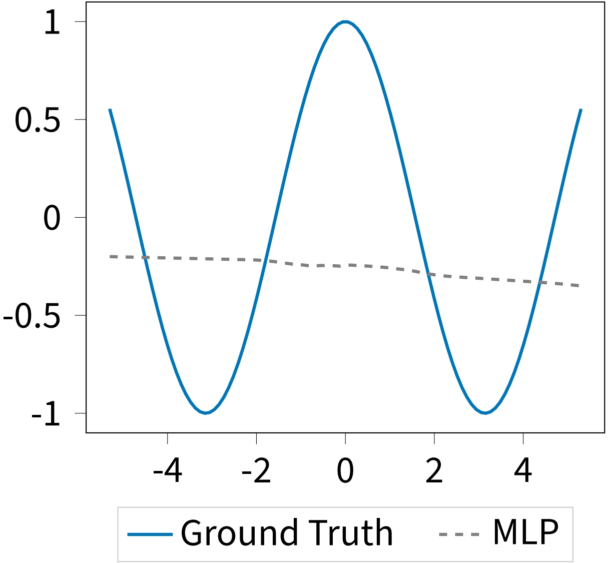
Initialization



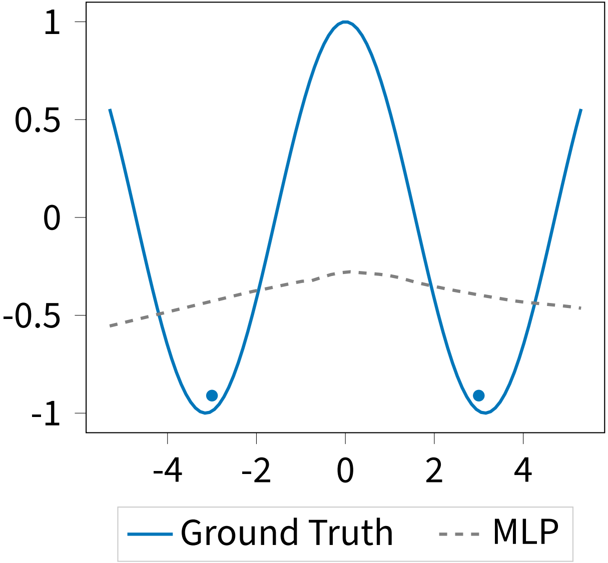
Initialization



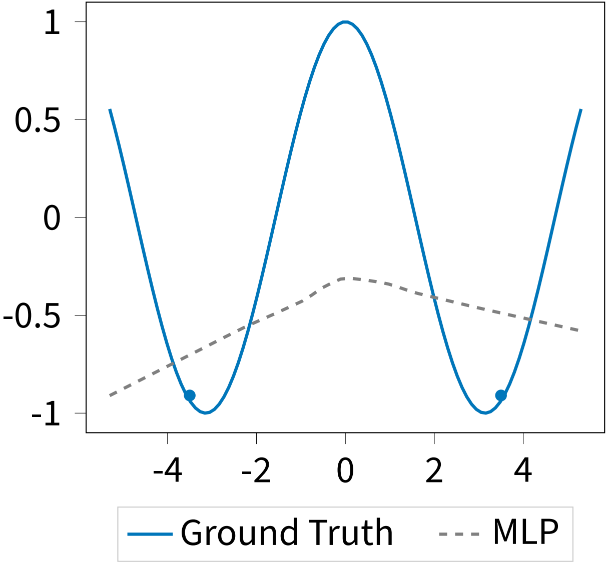
Initialization



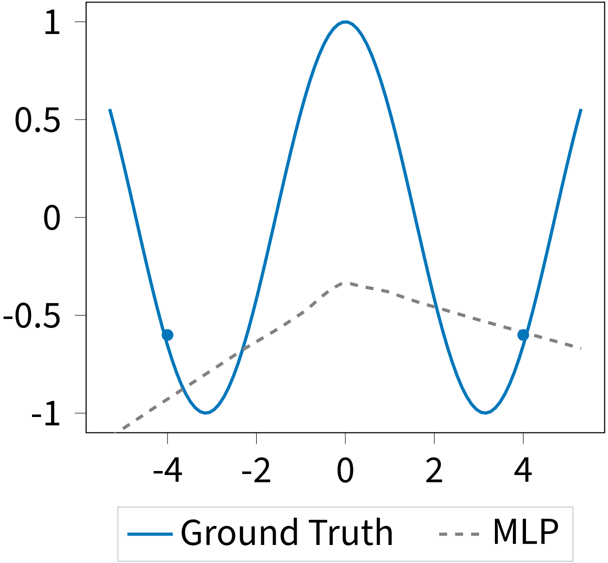
After 1 Training Step



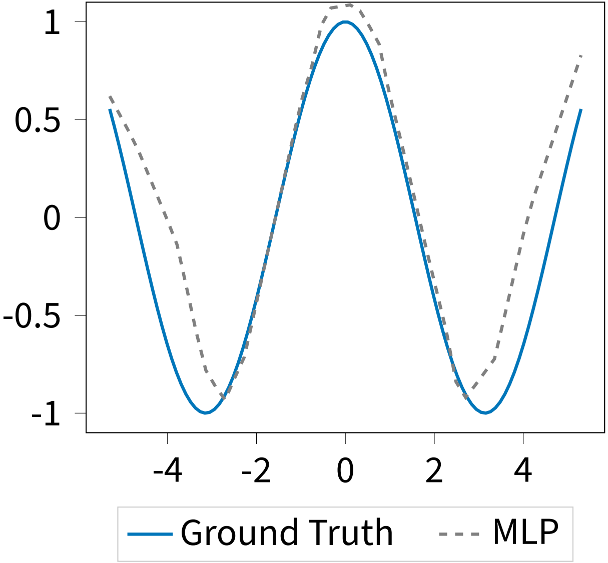
After 2 Training Steps



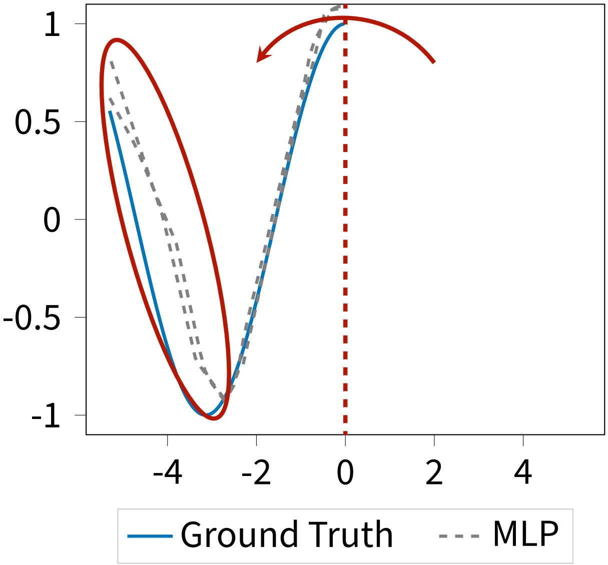
After 3 Training Steps



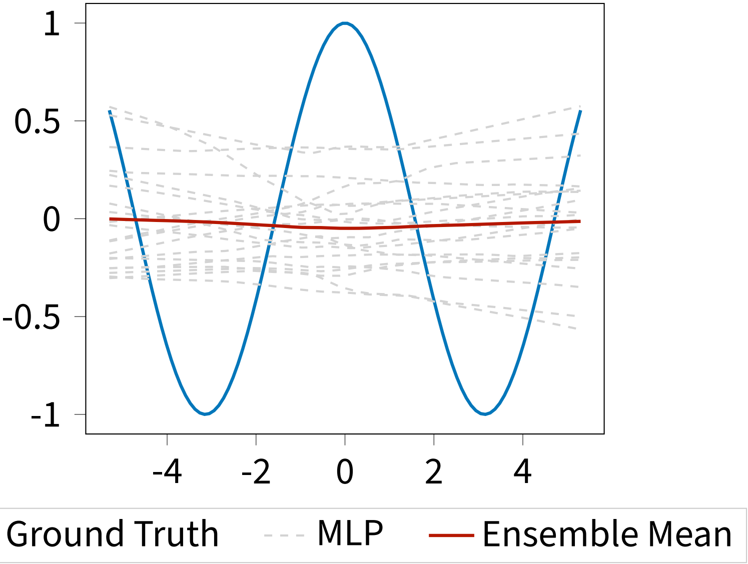
After 2000 Training Steps



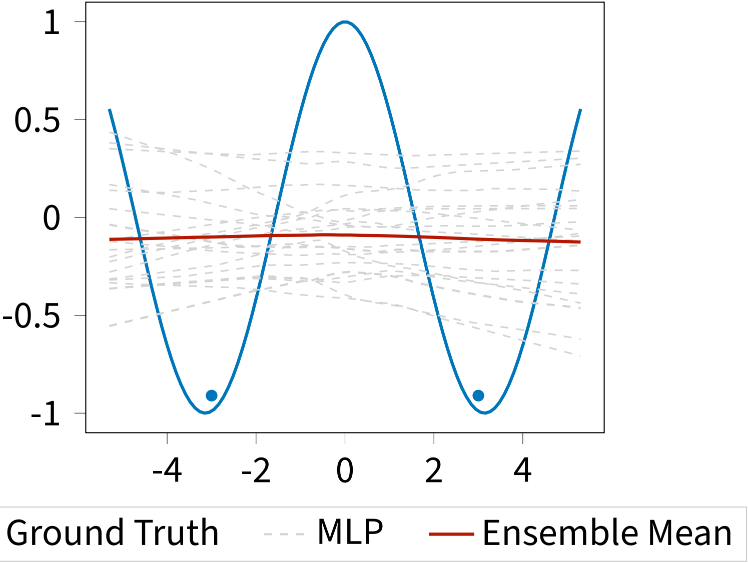
After 2000 Training Steps



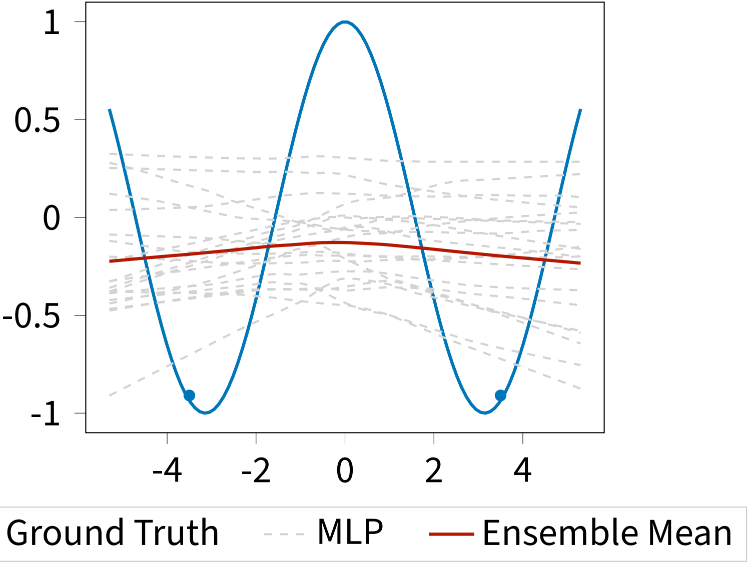
Initialization



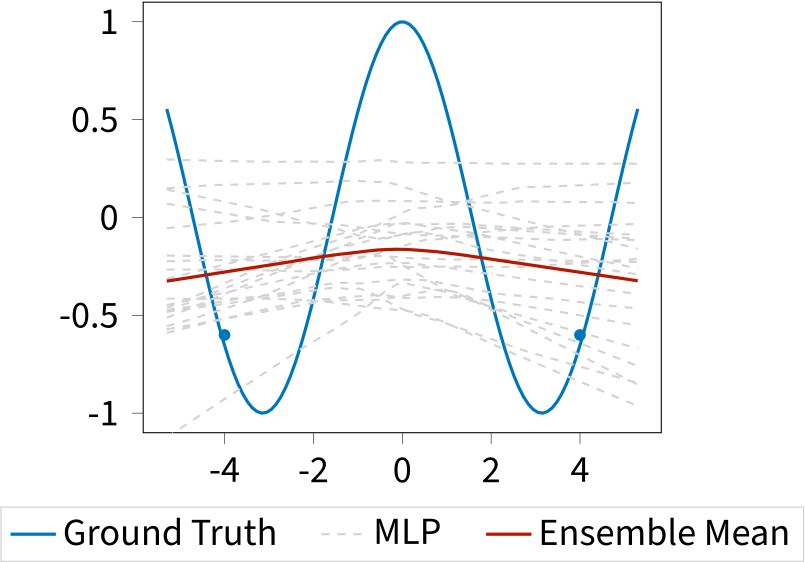
After 1 Training Step



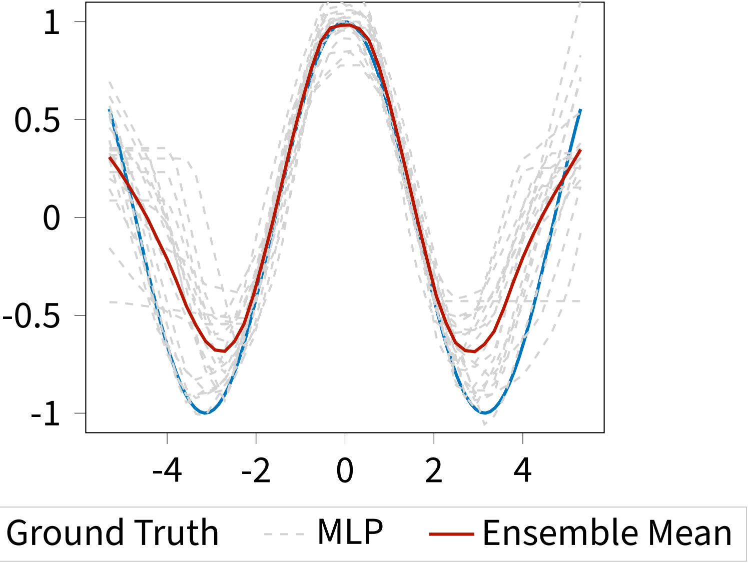
After 2 Training Steps



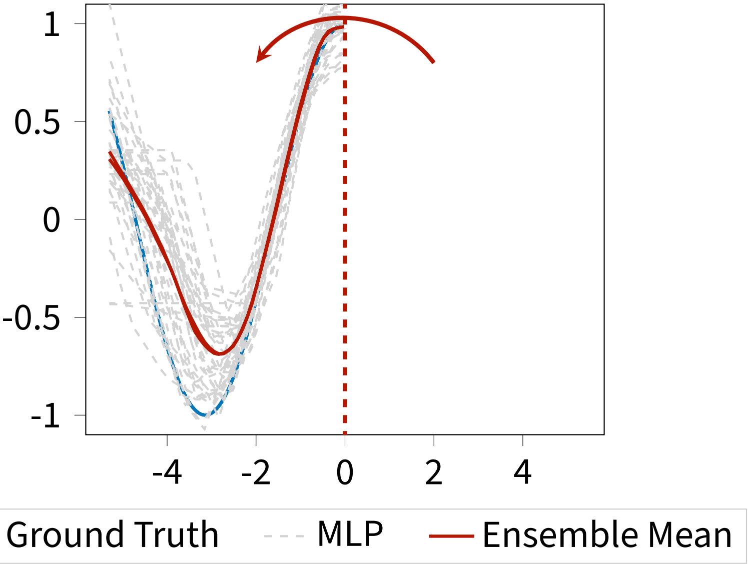
After 3 Training Steps



After 2000 Training Steps



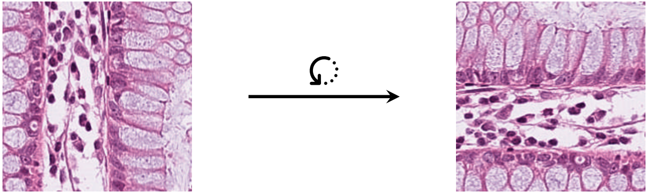
After 2000 Training Steps



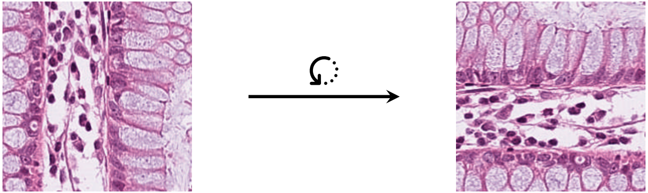
What Does An Augmented Ensemble Converge To?

Rotating images

Rotating images



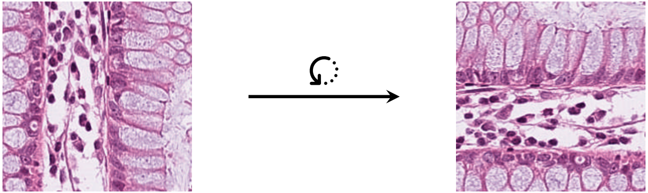
Rotating images



$f(x)$

f : pixels → colors

Rotating images



$$f(x) \xrightarrow{\circlearrowleft} f(\rho(g^{-1})x)$$

f : pixels → colors

$$= [\rho_{\text{reg}}(g)f](x)$$

Data augmentation and NTKs

Data augmentation and NTKs

Consider two ensembles:

trained without data augmentation

trained with data augmentation

Data augmentation and NTKs

Consider two ensembles:

trained without data augmentation

trained with data augmentation

If

$$\Theta^{\mathrm{non-aug}}(f,f') = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g\, \Theta^{\mathrm{aug}}(f, \rho_{\mathrm{reg}}(g) f')$$

Data augmentation and NTKs

Consider two ensembles:

If

$$\overset{\text{trained without data augmentation}}{\Theta^{\text{non}-\text{aug}}}(f, f') = \frac{1}{\text{vol}(G)} \int_G \text{d}g \, \overset{\text{trained with data augmentation}}{\Theta^{\text{aug}}}(f, \rho_{\text{reg}}(g) f')$$

Then

$$\mu_t^{\text{non}-\text{aug}}(x) = \mu_t^{\text{aug}}(x)$$

at infinite width.

Data augmentation and NTKs

Consider two ensembles:

trained without data augmentation

trained with data augmentation

If

$$\Theta^{\text{non-aug}}(f,f') = \frac{1}{\text{vol}(G)} \int_G \mathrm{d}g \, \Theta^{\text{aug}}(f, \rho_{\text{reg}}(g) f')$$

Then

$$\mu_t^{\text{non-aug}}(x) = \mu_t^{\text{aug}}(x) \quad \forall t$$

at infinite width.

Data augmentation and NTKs

Consider two ensembles:

trained without data augmentation

trained with data augmentation

If

$$\Theta^{\text{non-aug}}(f, f') = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g \, \Theta^{\text{aug}}(f, \rho_{\text{reg}}(g) f')$$

Then

$$\mu_t^{\text{non-aug}}(x) = \mu_t^{\text{aug}}(x) \quad \forall t \quad \forall x$$

at infinite width.

Data augmentation and NTKs

$$\Theta^{\mathrm{non-aug}}(f, f') = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g\, \Theta^{\mathrm{aug}}(f, \rho_{\mathrm{reg}}(g) f')$$

Data augmentation and NTKs

$$\Theta^{\mathrm{non-aug}}(f, f') = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g \, \Theta^{\mathrm{aug}}(f, \rho_{\mathrm{reg}}(g) f')$$

- Given an architecture with NTK Θ^{aug}, find an architecture with NTK $\Theta^{\mathrm{non-aug}}$

Group convolutions

[Cohen, Welling 2016]

Group convolutions

[Cohen, Welling 2016]

Group conv's are the (unique) linear layers equivariant wrt ρ_{reg}

Group convolutions

[Cohen, Welling 2016]

Group conv's are the (unique) linear layers equivariant wrt ρ_{reg}

- Ordinary convolutions

$$f'(y) = \int_X \mathrm{d}x\, \kappa(x-y)\, f(x)$$

Group convolutions

[Cohen, Welling 2016]

Group conv's are the (unique) linear layers equivariant wrt ρ_{reg}

- Ordinary convolutions

$$f'(y) = \int_X \mathrm{d}x \, \kappa(x - y) \, f(x)$$

- Group convolutions

$$f'(g) = \int_X \mathrm{d}x \, \kappa(\rho(g^{-1})x) \, f(x) \qquad \text{lifting}$$

Group convolutions

[Cohen, Welling 2016]

Group conv's are the (unique) linear layers equivariant wrt ρ_{reg}

- Ordinary convolutions

$$f'(y) = \int_X \mathrm{d}x\, \kappa(x - y)\, f(x)$$

- Group convolutions

$$f'(g) = \int_X \mathrm{d}x\, \kappa(\rho(g^{-1})x)\, f(x) \qquad \text{lifting}$$

$$f'(g) = \int_G \mathrm{d}g\, \kappa(g^{-1}h)\, f(h) \qquad \text{group convolution}$$

Group convolutions

[Cohen, Welling 2016]

Group conv's are the (unique) linear layers equivariant wrt ρ_{reg}

- Ordinary convolutions

$$f'(y) = \int_X \mathrm{d}x\, \kappa(x - y)\, f(x)$$

- Group convolutions

$$f'(g) = \int_X \mathrm{d}x\, \kappa(\rho(g^{-1})x)\, f(x) \qquad \text{lifting}$$

$$f'(g) = \int_G \mathrm{d}g\, \kappa(g^{-1}h)\, f(h) \qquad \text{group convolution}$$

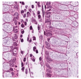
$$f' = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g\, f(g) \qquad \text{group pooling}$$

GCNNs

Stack GConv-layers to obtain an invariant network

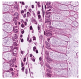
GCNNs

Stack GConv-layers to obtain an invariant network



GCNNs

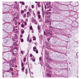
Stack GConv-layers to obtain an invariant network



→ lifting

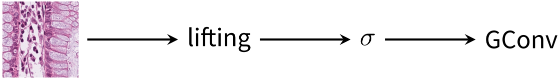
GCNNs

Stack GConv-layers to obtain an invariant network

 $\longrightarrow$ lifting $\longrightarrow$ σ

GCNNs

Stack GConv-layers to obtain an invariant network



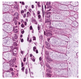
lifting

σ

GConv

GCNNs

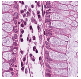
Stack GConv-layers to obtain an invariant network



$$\longrightarrow \text{lifting} \longrightarrow \sigma \longrightarrow \text{GConv} \longrightarrow \sigma$$

GCNNs

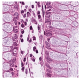
Stack GConv-layers to obtain an invariant network



$\longrightarrow$ lifting $\longrightarrow$ σ $\longrightarrow$ GConv $\longrightarrow$ σ $\longrightarrow$ GConv

GCNNs

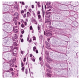
Stack GConv-layers to obtain an invariant network



→ lifting → σ → GConv → σ → GConv → GPool

NTKs for GCNNs

Stack GConv-layers to obtain an invariant network

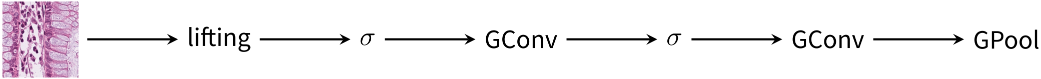


$\longrightarrow$ lifting $\longrightarrow \sigma \longrightarrow$ GConv $\longrightarrow \sigma \longrightarrow$ GConv $\longrightarrow$ GPool

Compute NTK with layer-wise recursion

NTKs for GCNNs

Stack GConv-layers to obtain an invariant network



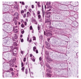
$\longrightarrow$ lifting $\longrightarrow$ σ $\longrightarrow$ GConv $\longrightarrow$ σ $\longrightarrow$ GConv $\longrightarrow$ GPool

Compute NTK with layer-wise recursion

0

NTKs for GCNNs

Stack GConv-layers to obtain an invariant network



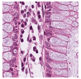
$$\longrightarrow \text{lifting} \longrightarrow \sigma \longrightarrow \text{GConv} \longrightarrow \sigma \longrightarrow \text{GConv} \longrightarrow \text{GPool}$$

Compute NTK with layer-wise recursion

$$0 \longrightarrow \Theta^{(1)}_{g,g'}(f,f')$$

NTKs for GCNNs

Stack GConv-layers to obtain an invariant network



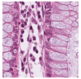
$\longrightarrow$ lifting $\longrightarrow \sigma \longrightarrow$ GConv $\longrightarrow \sigma \longrightarrow$ GConv $\longrightarrow$ GPool

Compute NTK with layer-wise recursion

$$0 \longrightarrow \Theta^{(1)}_{g,g'}(f,f') \longrightarrow \Theta^{(2)}_{g,g'}(f,f')$$

NTKs for GCNNs

Stack GConv-layers to obtain an invariant network



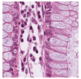
$\longrightarrow$ lifting $\longrightarrow$ σ $\longrightarrow$ GConv $\longrightarrow$ σ $\longrightarrow$ GConv $\longrightarrow$ GPool

Compute NTK with layer-wise recursion

$$0 \longrightarrow \Theta^{(1)}_{g,g'}(f,f') \longrightarrow \Theta^{(2)}_{g,g'}(f,f') \longrightarrow \Theta^{(3)}_{g,g'}(f,f')$$

NTKs for GCNNs

Stack GConv-layers to obtain an invariant network



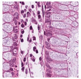
$$\longrightarrow \text{lifting} \longrightarrow \sigma \longrightarrow \text{GConv} \longrightarrow \sigma \longrightarrow \text{GConv} \longrightarrow \text{GPool}$$

Compute NTK with layer-wise recursion

$$0 \longrightarrow \Theta^{(1)}_{g,g'}(f,f') \longrightarrow \Theta^{(2)}_{g,g'}(f,f') \longrightarrow \Theta^{(3)}_{g,g'}(f,f') \longrightarrow \Theta^{(4)}_{g,g'}(f,f')$$

NTKs for GCNNs

Stack GConv-layers to obtain an invariant network



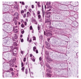
$$\longrightarrow \text{lifting} \longrightarrow \sigma \longrightarrow \text{GConv} \longrightarrow \sigma \longrightarrow \text{GConv} \longrightarrow \text{GPool}$$

Compute NTK with layer-wise recursion

$$0 \longrightarrow \Theta^{(1)}_{g,g'}(f,f') \longrightarrow \Theta^{(2)}_{g,g'}(f,f') \longrightarrow \Theta^{(3)}_{g,g'}(f,f') \longrightarrow \Theta^{(4)}_{g,g'}(f,f') \longrightarrow \Theta^{(5)}_{g,g'}(f,f')$$

NTKs for GCNNs

Stack GConv-layers to obtain an invariant network



$$\longrightarrow \text{lifting} \longrightarrow \sigma \longrightarrow \text{GConv} \longrightarrow \sigma \longrightarrow \text{GConv} \longrightarrow \text{GPool}$$

Compute NTK with layer-wise recursion

$$\begin{tikzcd} 0 \arrow[r] & \Theta^{(1)}_{g,g'}(f,f') \arrow[r] & \Theta^{(2)}_{g,g'}(f,f') \arrow[r] & \Theta^{(3)}_{g,g'}(f,f') \arrow[r] & \Theta^{(4)}_{g,g'}(f,f') \arrow[r] & \Theta^{(5)}_{g,g'}(f,f') \arrow[r] & \Theta(f,f') \end{tikzcd}$$

NTKs of MLPs and GCNNs

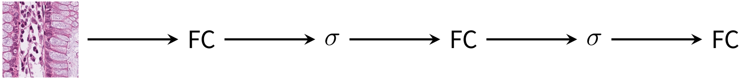
NTKs of MLPs and GCNNs

- Consider two neural networks

NTKs of MLPs and GCNNs

- Consider two neural networks

An MLP

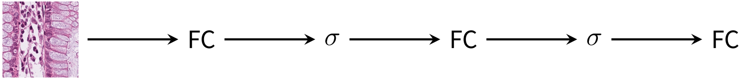


$\longrightarrow$ FC $\longrightarrow$ σ $\longrightarrow$ FC $\longrightarrow$ σ $\longrightarrow$ FC

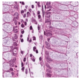
NTKs of MLPs and GCNNs

- Consider two neural networks

An MLP



A GCNN

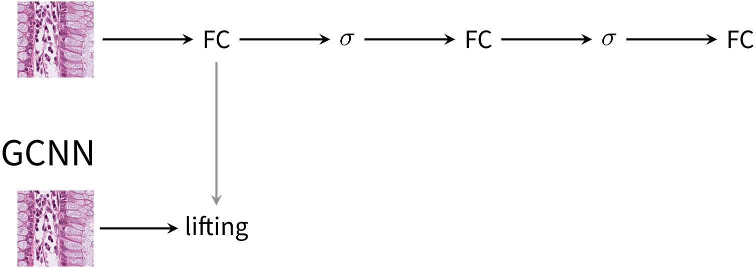


NTKs of MLPs and GCNNs

- Consider two neural networks

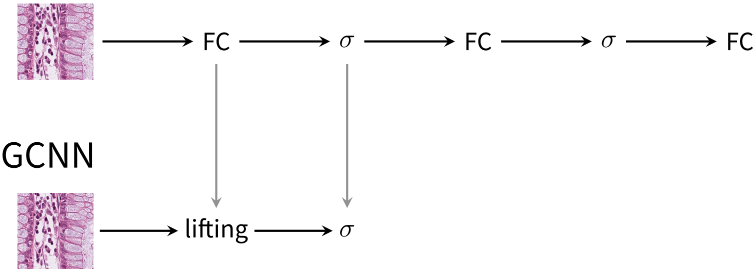
An MLP

A GCNN



NTKs of MLPs and GCNNs

- Consider two neural networks



An MLP

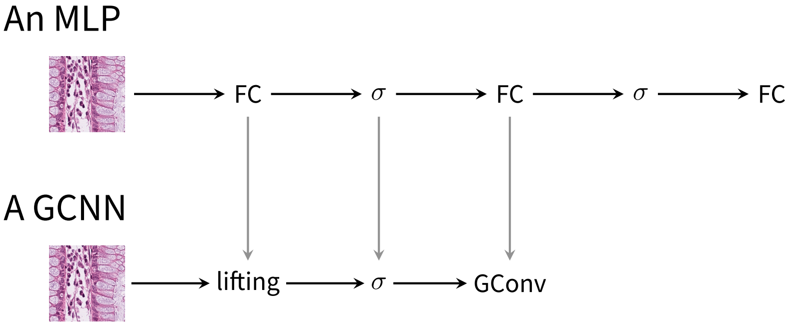
FC → σ → FC → σ → FC

A GCNN

lifting → σ

NTKs of MLPs and GCNNs

- Consider two neural networks



An MLP

FC → σ → FC → σ → FC

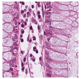
A GCNN

lifting → σ → GConv

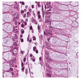
NTKs of MLPs and GCNNs

- Consider two neural networks

An MLP



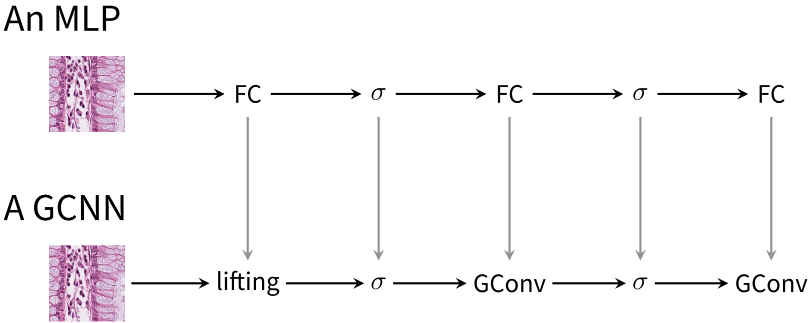
A GCNN



$$\begin{tikzcd}
\cdot \arrow[r] & \text{FC} \arrow[r] \arrow[d] & \sigma \arrow[r] \arrow[d] & \text{FC} \arrow[r] \arrow[d] & \sigma \arrow[r] \arrow[d] & \text{FC} \\
\cdot \arrow[r] & \text{lifting} \arrow[r] & \sigma \arrow[r] & \text{GConv} \arrow[r] & \sigma &
\end{tikzcd}$$

NTKs of MLPs and GCNNs

- Consider two neural networks

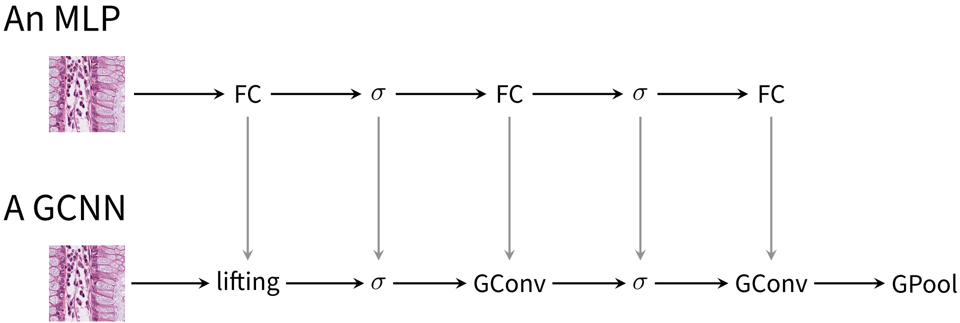


An MLP

A GCNN

NTKs of MLPs and GCNNs

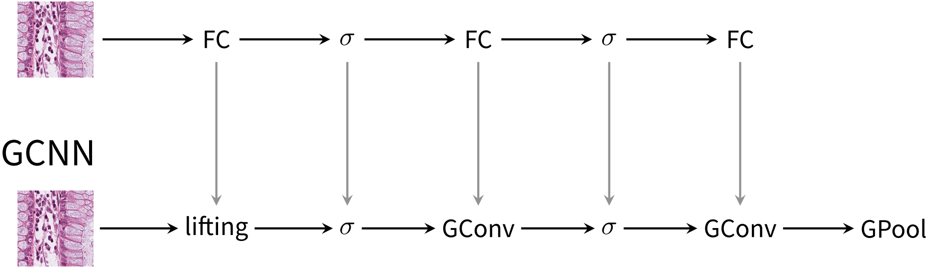
- Consider two neural networks



NTKs of MLPs and GCNNs

- Consider two neural networks

An MLP



→ FC → σ → FC → σ → FC

A GCNN

→ lifting → σ → GConv → σ → GConv → GPool

- Then

$$\Theta^{\mathrm{GCNN}}(f,f') = \frac{1}{\mathrm{vol}(G)}\int_G \mathrm{d}g\,\Theta^{\mathrm{MLP}}(f,\rho_{\mathrm{reg}}(g)f')$$

Data augmentation of MLPs

$$\Theta^{\mathrm{GCNN}}(f, f') = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g \, \Theta^{\mathrm{MLP}}(f, \rho_{\mathrm{reg}}(g) f')$$

Data augmentation of MLPs

before: non-aug

$$\Theta^{\mathrm{GCNN}}(f, f') = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g \, \Theta^{\mathrm{MLP}}(f, \rho_{\mathrm{reg}}(g) f')$$

Data augmentation of MLPs

$$\overset{\text{before: non-aug}}{\Theta^{\text{GCNN}}}(f, f') = \frac{1}{\text{vol}(G)} \int_G \mathrm{d}g \, \overset{\text{before: aug}}{\Theta^{\text{MLP}}}(f, \rho_{\text{reg}}(g) f')$$

Data augmentation of MLPs

before: non-aug

before: aug

$$\Theta^{\mathrm{GCNN}}(f,f') = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g\, \Theta^{\mathrm{MLP}}(f,\rho_{\mathrm{reg}}(g)f')$$

⇨ training the MLP on *G*-augmented data

Data augmentation of MLPs

before: non-aug

before: aug

$$\Theta^{\text{GCNN}}(f, f') = \frac{1}{\text{vol}(G)} \int_G \mathrm{d}g \, \Theta^{\text{MLP}}(f, \rho_{\text{reg}}(g) f')$$

⇨ training the MLP on *G*-augmented data = training the GCNN on unaugmented data

Data augmentation of MLPs

before: non-aug

before: aug

$$\Theta^{\mathrm{GCNN}}(f,f') = \frac{1}{\mathrm{vol}(G)}\int_G \mathrm{d}g\, \Theta^{\mathrm{MLP}}(f,\rho_{\mathrm{reg}}(g)f')$$

⇨ training the MLP on *G*-augmented data = training the GCNN on unaugmented data

in the ensemble mean

Data augmentation of MLPs

before: non-aug

before: aug

$$\Theta^{\mathrm{GCNN}}(f, f') = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g\, \Theta^{\mathrm{MLP}}(f, \rho_{\mathrm{reg}}(g) f')$$

⇨ training the MLP on *G*-augmented data = training the GCNN on unaugmented data

in the ensemble mean, $\forall t$, $\forall x$

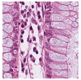
Data augmentation of CNNs

Data augmentation of CNNs

- Consider a CNN and a GCNN invariant wrt. roto-translations

Data augmentation of CNNs

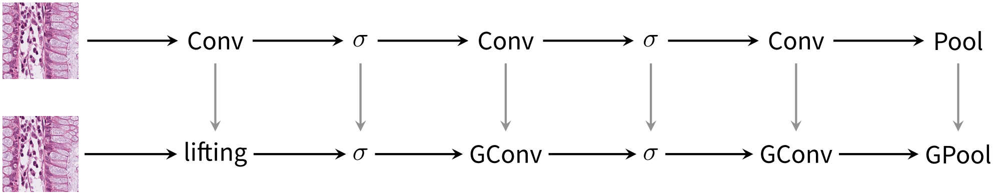
- Consider a CNN and a GCNN invariant wrt. roto-translations



$$\longrightarrow \text{Conv} \longrightarrow \sigma \longrightarrow \text{Conv} \longrightarrow \sigma \longrightarrow \text{Conv} \longrightarrow \text{Pool}$$

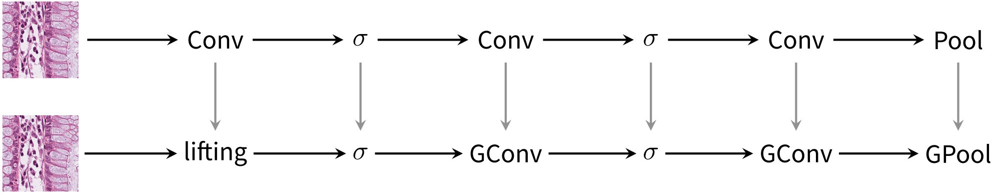
Data augmentation of CNNs

- Consider a CNN and a GCNN invariant wrt. roto-translations



Data augmentation of CNNs

- Consider a CNN and a GCNN invariant wrt. roto-translations

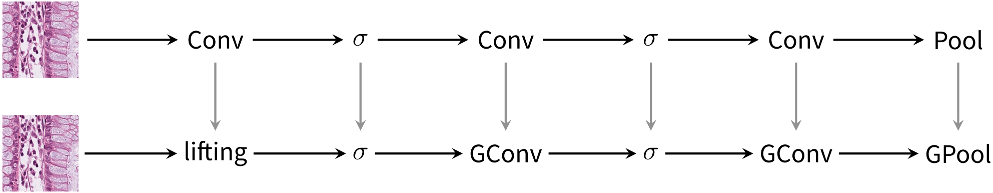


- Then

$$\Theta^{\text{GCNN}}(f, f') = \frac{1}{n} \sum_{r \in C_n} \Theta^{\text{CNN}}(f, \rho_{\text{reg}}(r) f')$$

Data augmentation of CNNs

- Consider a CNN and a GCNN invariant wrt. roto-translations



$$\begin{tikzcd}
{} \arrow[r] & \text{Conv} \arrow[r] \arrow[d] & \sigma \arrow[r] \arrow[d] & \text{Conv} \arrow[r] \arrow[d] & \sigma \arrow[r] \arrow[d] & \text{Conv} \arrow[r] \arrow[d] & \text{Pool} \arrow[d] \\
{} \arrow[r] & \text{lifting} \arrow[r] & \sigma \arrow[r] & \text{GConv} \arrow[r] & \sigma \arrow[r] & \text{GConv} \arrow[r] & \text{GPool}
\end{tikzcd}$$

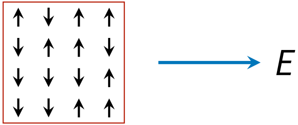
- Then

$$\Theta^{\text{GCNN}}(f, f') = \frac{1}{n} \sum_{r \in C_n} \Theta^{\text{CNN}}(f, \rho_{\text{reg}}(r) f')$$

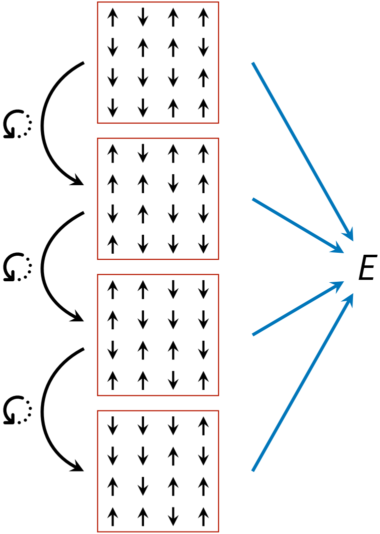
⇨ By training the CNN on augmented images, one obtains a roto-translation invariant GCNN

Experiments

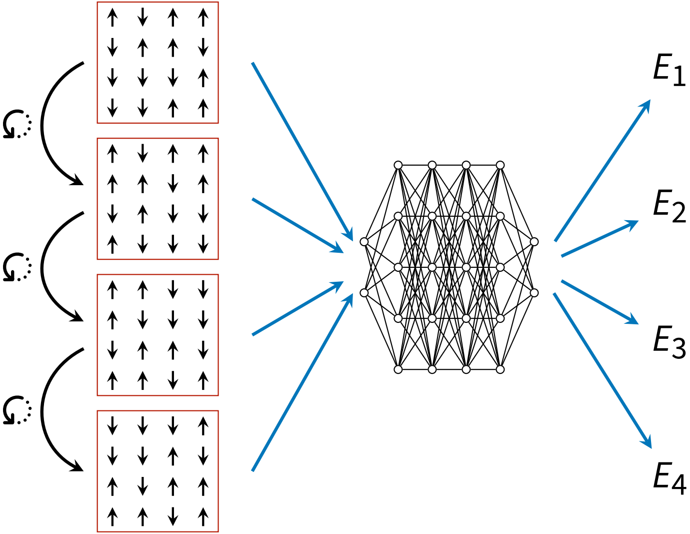
Ising model



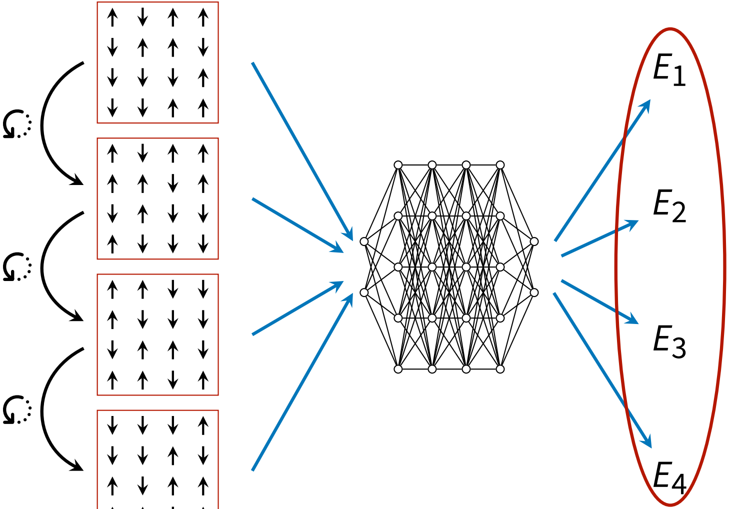
Ising model



Ising model

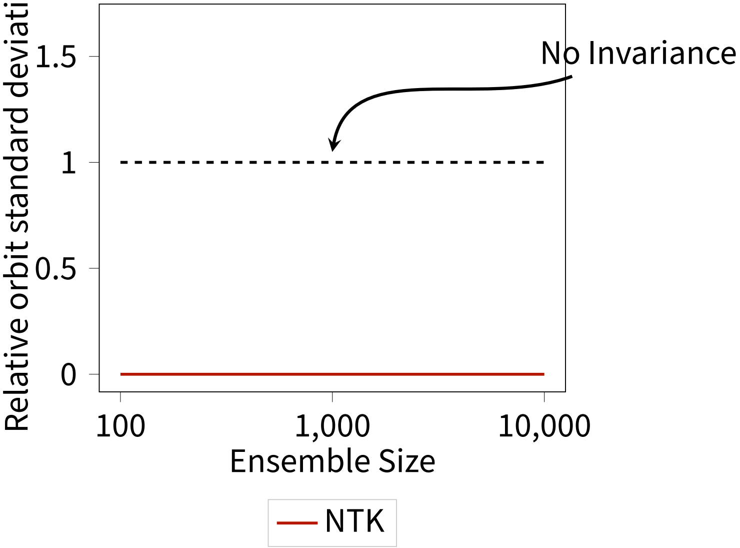


Ising model

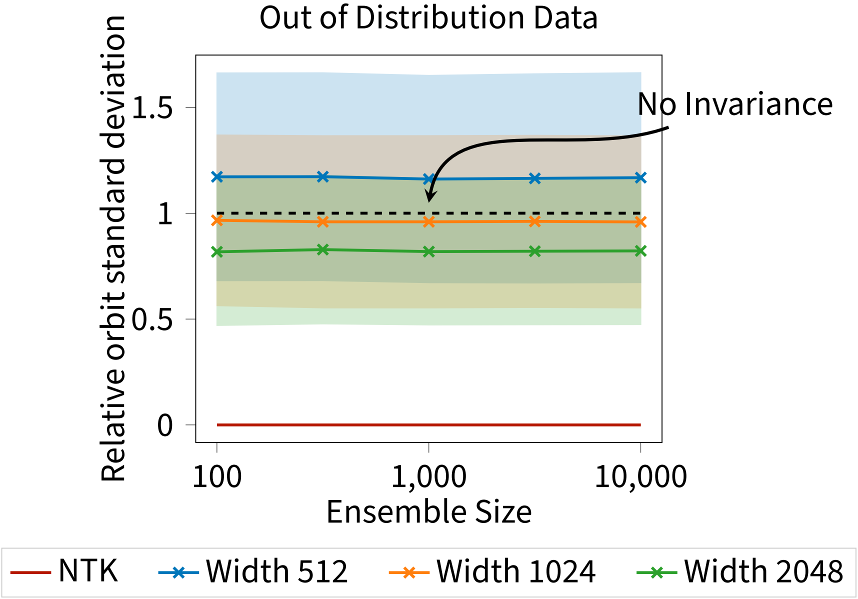


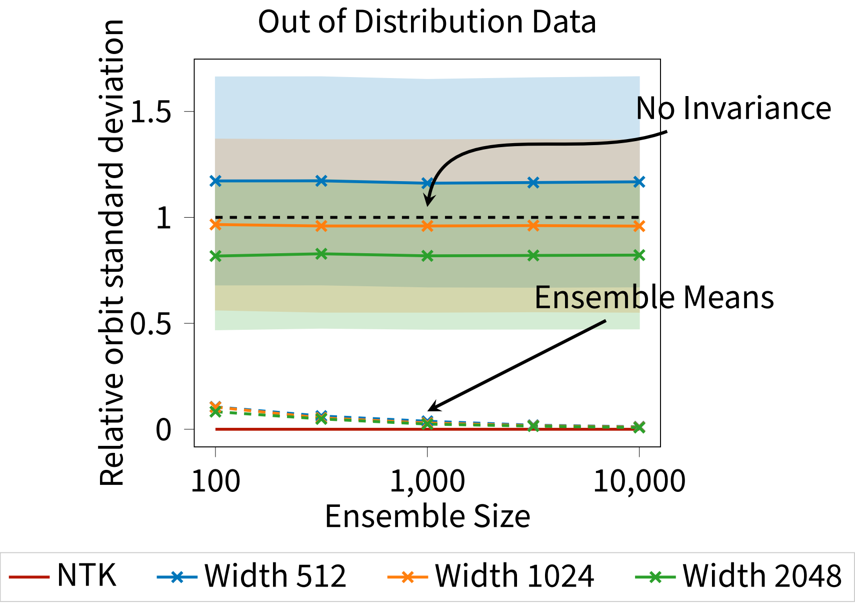
Relative Standard Deviation

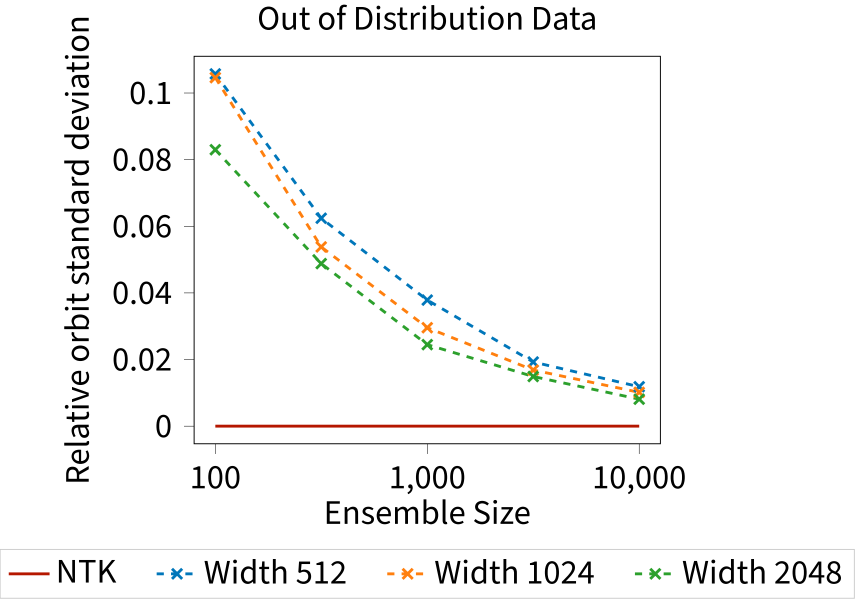
Out of Distribution Data



No Invariance

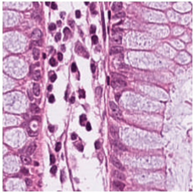

Out of Distribution Data
Relative orbit standard deviation
1.5
1
0.5
0
100
1,000
10,000
Ensemble Size
No Invariance
NTK
Width 512
Width 1024
Width 2048


Out of Distribution Data
Relative orbit standard deviation
1.5
1
0.5
0
100
1,000
10,000
Ensemble Size
No Invariance
Ensemble Means
NTK
Width 512
Width 1024
Width 2048


Out of Distribution Data
Relative orbit standard deviation
0.1
0.08
0.06
0.04
0.02
0
100
1,000
10,000
Ensemble Size
NTK
Width 512
Width 1024
Width 2048

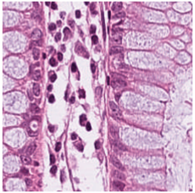
Histological slices

[Kather et al. 2018]



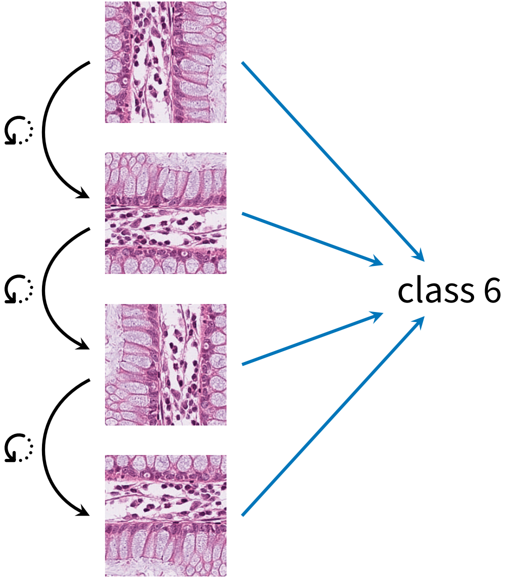
Histological slices

[Kather et al. 2018]

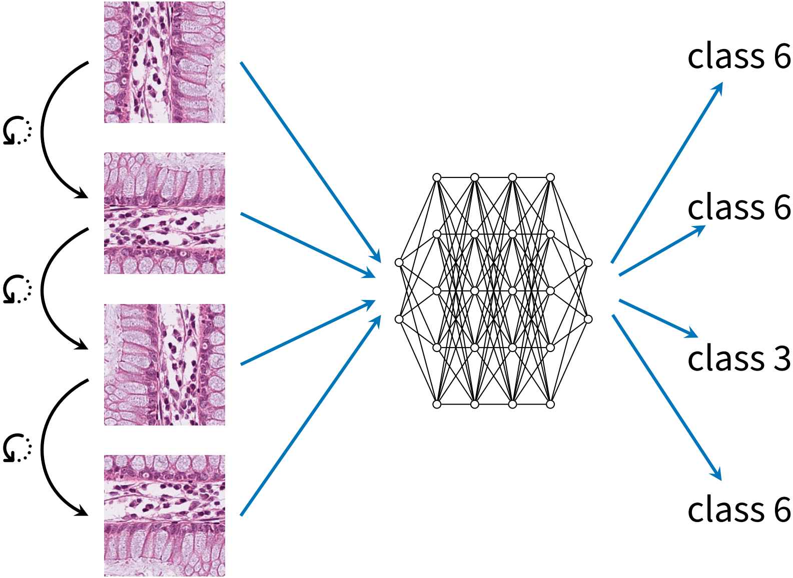


class 6

Histological slices



Histological slices



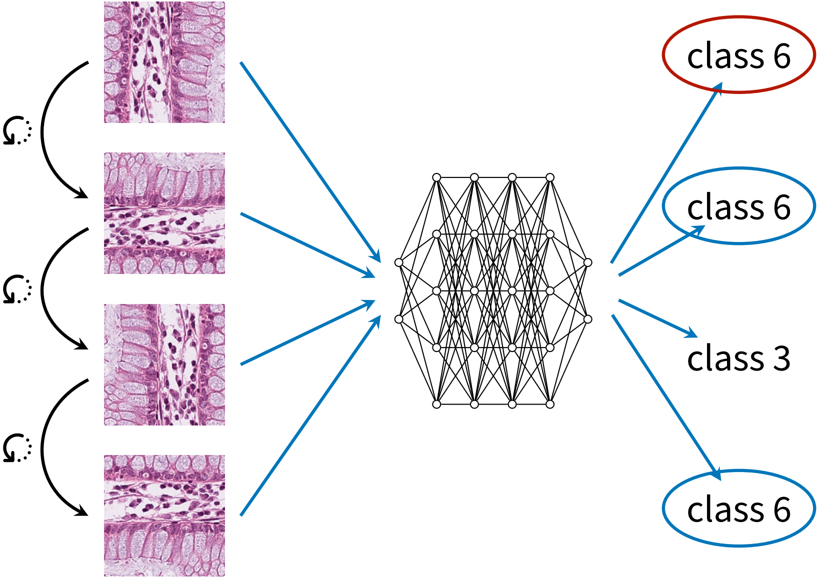
class 6

class 6

class 3

class 6

Histological slices



class 6

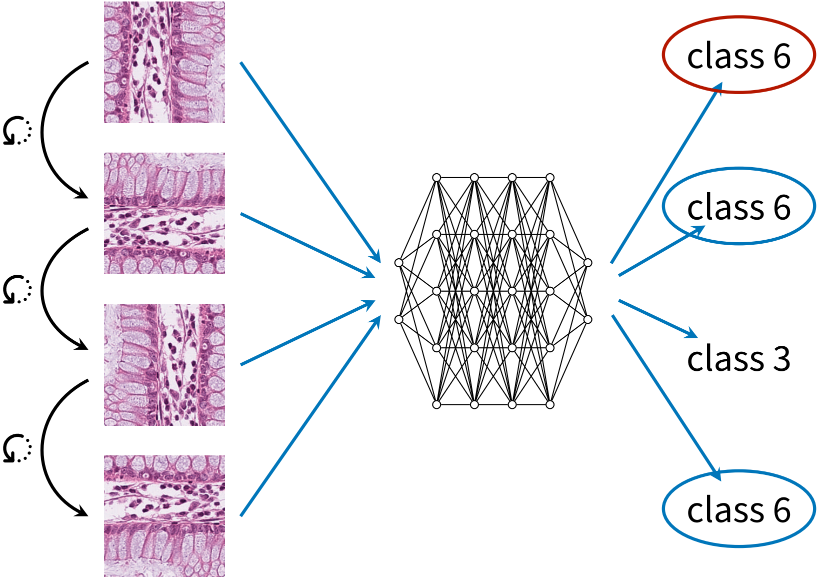
class 6

class 3

class 6

Histological slices

Orbit Same Predictions = 3



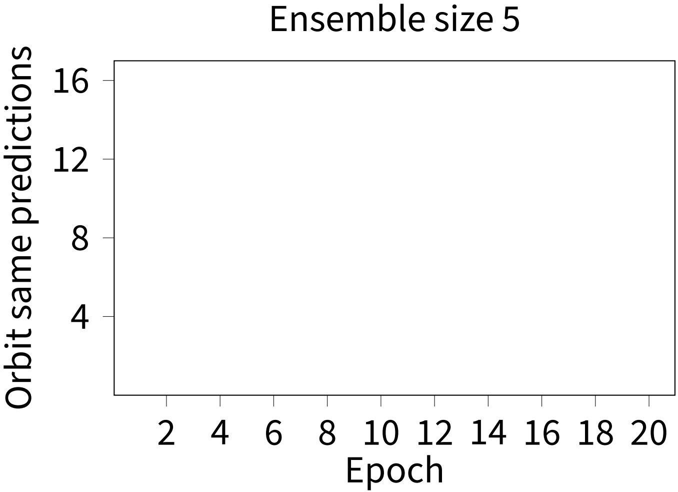
class 6

class 6

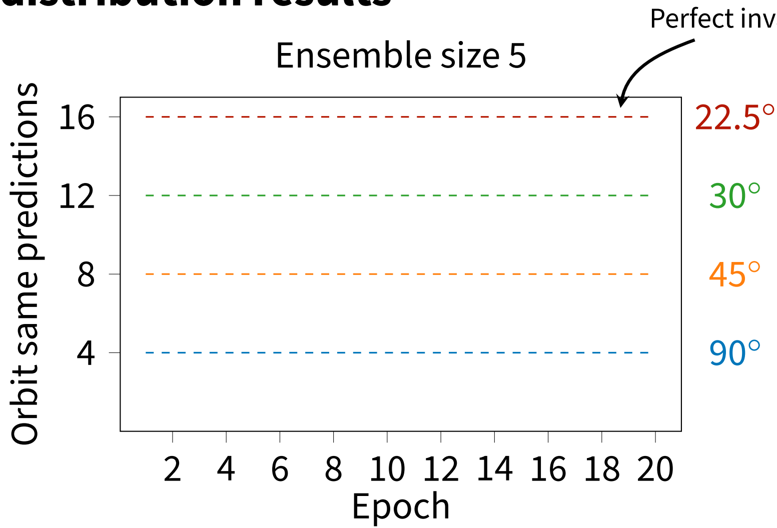
class 3

class 6

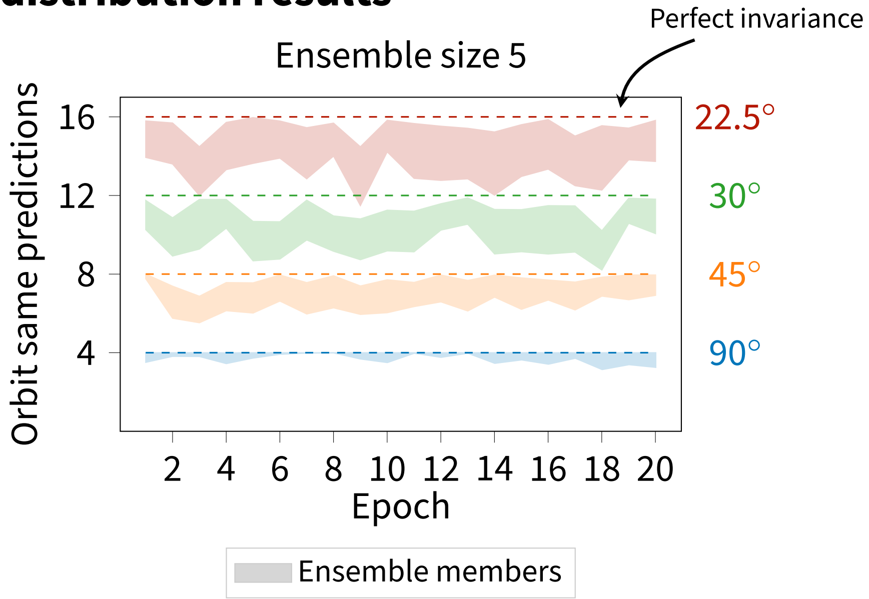
Out of distribution results



Out of distribution results

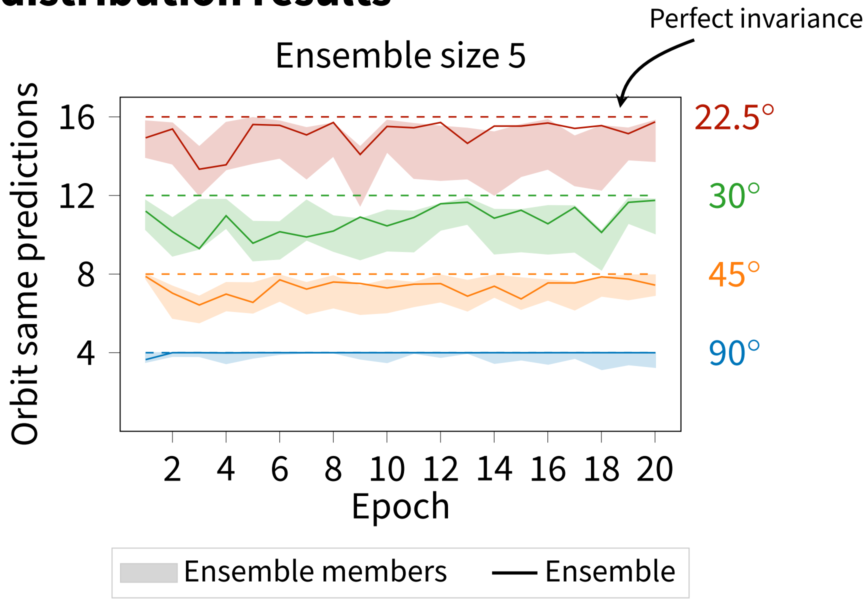


Out of distribution results

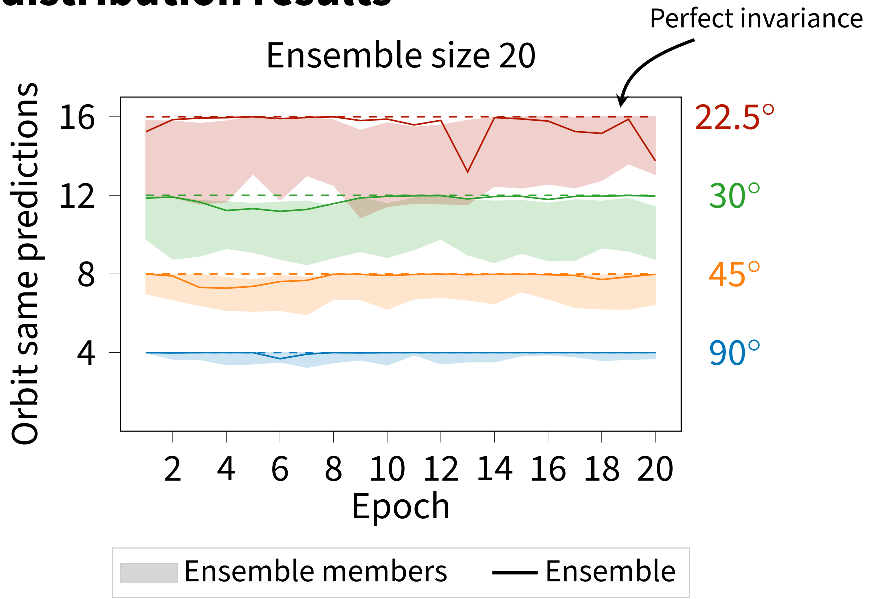


Perfect invariance

Out of distribution results


Perfect invariance

Out of distribution results



Ensemble size 20

Perfect invariance

Further experimental results

Further experimental results

✓ Emergent invariance for rotated FashionMNIST

Further experimental results

- ✓ Emergent invariance for rotated FashionMNIST
- ✓ Partial augmentation for continuous symmetries

Further experimental results

- ✓ Emergent invariance for rotated FashionMNIST
- ✓ Partial augmentation for continuous symmetries
- ✓ Emergent equivariance (as opposed to invariance)

Comparison to other methods

Comparison to other methods

- ➪ Models trained on rotated FashionMNIST

Comparison to other methods

➪ Models trained on rotated FashionMNIST

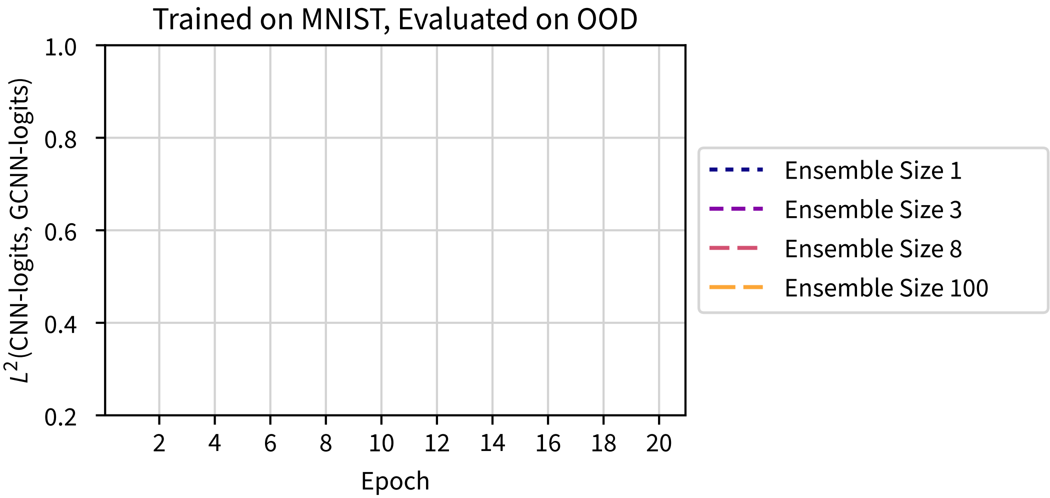
Orbit same predictions out of distribution:

	C_4	C_8	C_{16}
DeepEns+DA	3.85±0.12	**7.72±0.34**	**15.24±0.69**
only DA	3.41±0.18	6.73±0.24	12.77±0.71
E2CNN[1]	**4±0.0**	**7.71±0.21**	**15.08±0.34**
Canon[2]	**4±0.0**	**7.45±0.14**	12.41±0.85

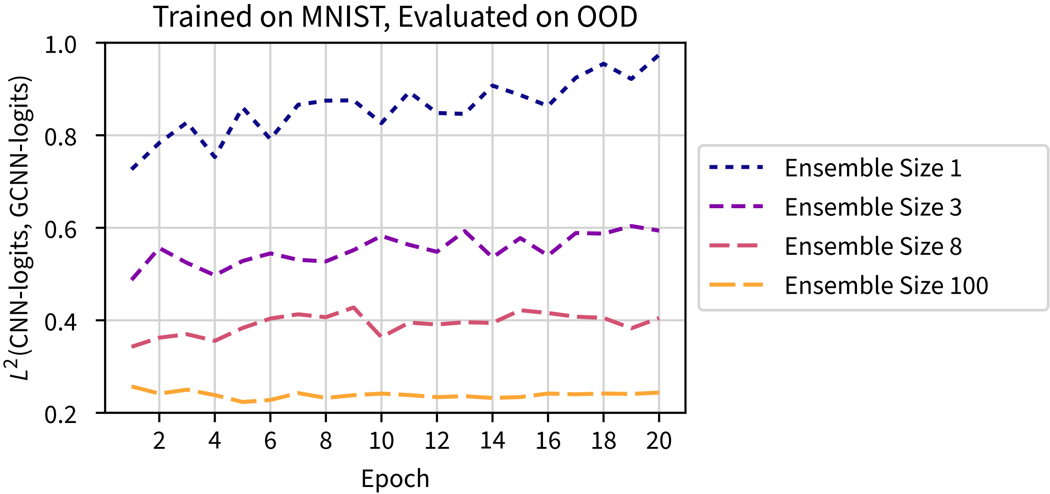
[1] [Weiler et al. 2019], [2] [Kaba et al. 2022]

Convergence of augmented CNNs to GCNNs

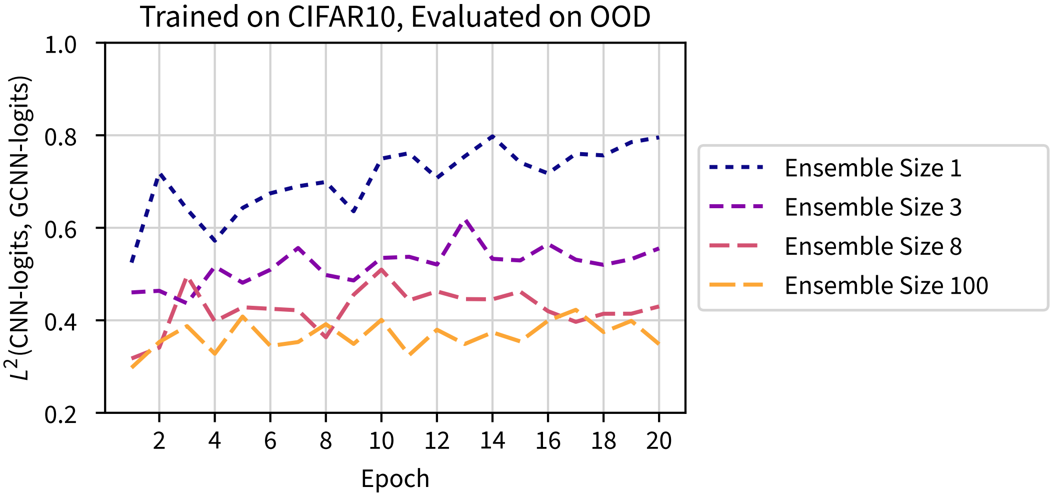
Convergence of augmented CNNs to GCNNs



Convergence of augmented CNNs to GCNNs



Convergence of augmented CNNs to GCNNs



Key takeaways

Key takeaways

If you need ensembles

👍 use data augmentation to obtain an equivariant model.

Key takeaways

If you need ensembles

👍 use data augmentation to obtain an equivariant model.

If you need data augmentation

👍 use an ensemble to boost the equivariance.

Key takeaways

If you need ensembles

👍 use data augmentation to obtain an equivariant model.

If you need data augmentation

👍 use an ensemble to boost the equivariance.

Analysis of neural tangent kernel can lead to powerful practical insights!

Papers

- Emergent Equivariance in Deep Ensembles
  Jan E. Gerken*, Pan Kessel*

  ICML 2024 (Oral)

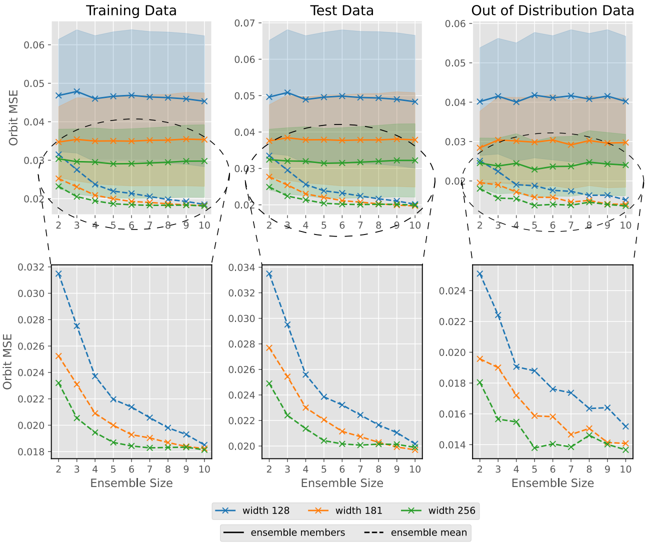
  * Equal contribution
- Equivariant Neural Tangent Kernels
  Philipp Misof, Pan Kessel, Jan E. Gerken

  arXiv: 2406.06504

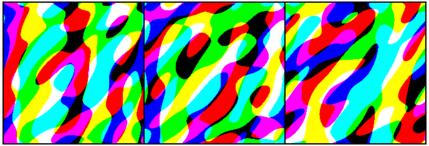
Thank you

Backup

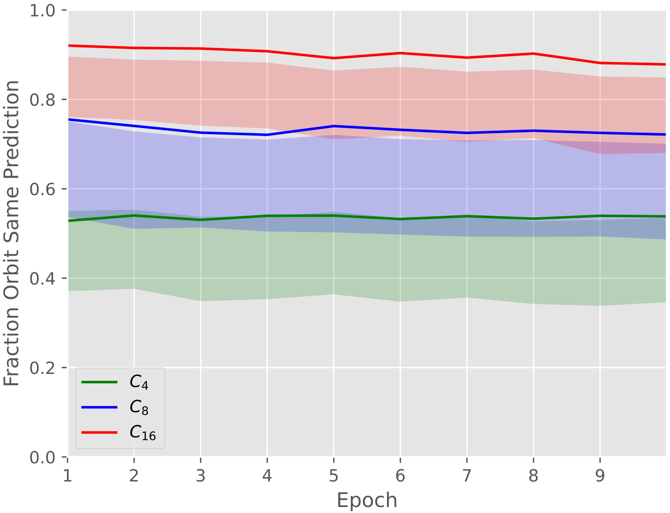
Emergent equivariance of cross products



Histological Data – OOD samples

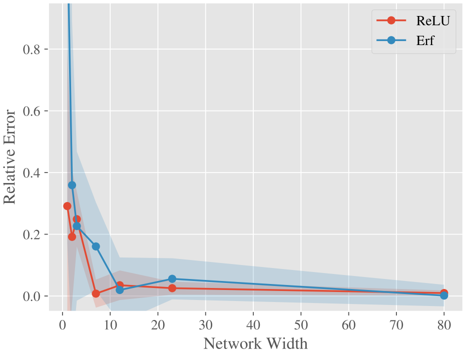


Emergent continuous symmetry on FashionMNIST

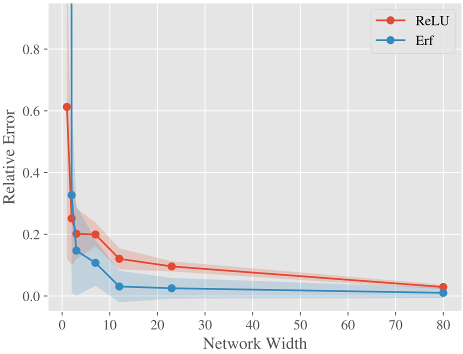


Kernel convergence

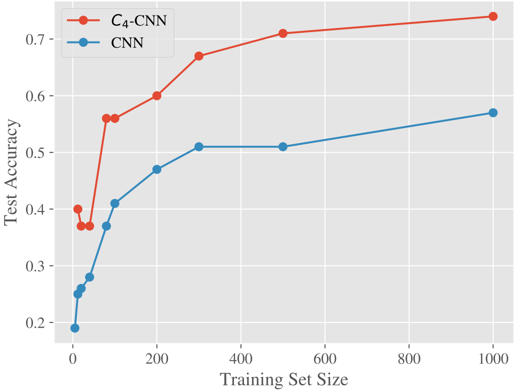
NNGP



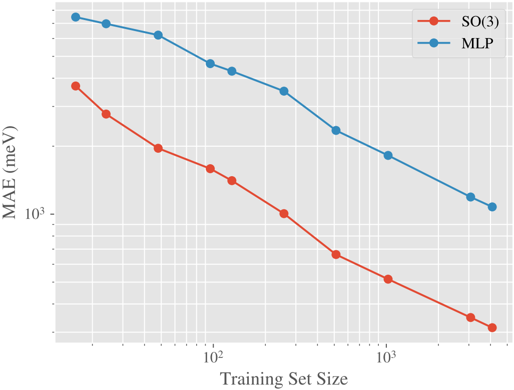
NTK



Equivariant NTKs for medical image classification

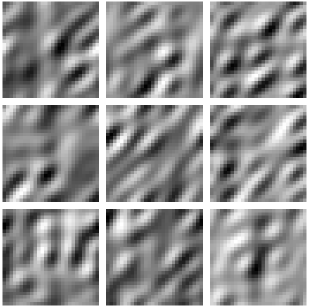


Equivariant NTKs for molecular property regression



OOD samples for CNN to GCNN convergence

MNIST



CIFAR10