Emergent Equivariance in Deep Ensembles

Jan E. Gerken

Talk at the
Workshop in statistical aspects related to machine learning
Fredrikstad, March 2024

Based on joint work with
Pan Kessel

Motivation

Symmetries in ML

Many learning problems are symmetric w.r.t. transformations by a symmetry group G

- G acts with some representation $\rho_X : G \to \mathrm{GL}(X)$ on the inputs $x_i \in X$
- G acts with some representation $\rho_Y : G \to \mathrm{GL}(Y)$ on the outputs $y_i \in Y$

Symmetries in ML

Many learning problems are symmetric w.r.t. transformations by a symmetry group G

- G acts with some representation $\rho_X : G \to \mathrm{GL}(X)$ on the inputs $x_i \in X$
- G acts with some representation $\rho_Y : G \to \mathrm{GL}(Y)$ on the outputs $y_i \in Y$
- In a symmetric learning problem, we have

$$(x, y) \in \mathcal{D} \quad \Rightarrow \quad (\rho_X(g)x, \rho_Y(g)y) \in \mathcal{D} \quad \forall g \in G$$

- Hence, the map $f : x \mapsto y$ satisfies

$$f(\rho_X(g)x) = \rho_Y(g)f(x) \qquad \forall g \in G$$

Many learning problems are symmetric w.r.t. transformations by a symmetry group G

- G acts with some representation $\rho_X : G \to \mathrm{GL}(X)$ on the inputs $x_i \in X$
- G acts with some representation $\rho_Y : G \to \mathrm{GL}(Y)$ on the outputs $y_i \in Y$
- In a symmetric learning problem, we have

$$(x, y) \in \mathcal{D} \quad \Rightarrow \quad (\rho_X(g)x, \rho_Y(g)y) \in \mathcal{D} \quad \forall g \in G$$

- Hence, the map $f : x \mapsto y$ satisfies

$$f(\rho_X(g)x) = \rho_Y(g)f(x) \qquad \forall g \in G$$

- If $\rho_Y \equiv \mathbb{1}$ then we call f ***invariant***, otherwise ***equivariant***

Symmetries in ML

Many learning problems are symmetric w.r.t. transformations by a symmetry group G

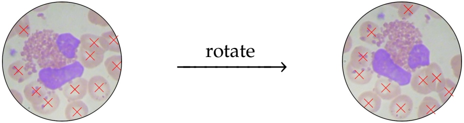
- G acts with some representation $\rho_X : G \to \mathrm{GL}(X)$ on the inputs $x_i \in X$
- G acts with some representation $\rho_Y : G \to \mathrm{GL}(Y)$ on the outputs $y_i \in Y$
- In a symmetric learning problem, we have

$$(x, y) \in \mathcal{D} \quad \Rightarrow \quad (\rho_X(g)x, \rho_Y(g)y) \in \mathcal{D} \quad \forall g \in G$$

- Hence, the map $f : x \mapsto y$ satisfies

$$f(\rho_X(g)x) = \rho_Y(g)f(x) \qquad \forall g \in G$$

- If $\rho_Y \equiv \mathbb{1}$ then we call f ***invariant***, otherwise ***equivariant***



Data augmentation

In ***data augmentation***, we train on an enlarged training dataset:

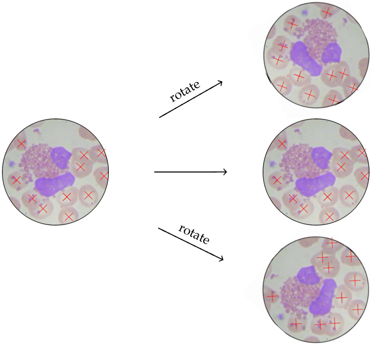
$$\mathcal{T} = \{(x_i, y_i) \mid i = 1, \ldots, N\} \qquad \rightarrow \qquad \mathcal{T} = \bigcup_{g \in G} \{(\rho_X(g) x_i, \rho_Y(g) y_i) \mid i = 1, \ldots, N\}$$

Data augmentation

In ***data augmentation***, we train on an enlarged training dataset:

$$\mathcal{T} = \{(x_i, y_i) \mid i = 1, \ldots, N\} \qquad \longrightarrow \qquad \mathcal{T} = \bigcup_{g \in G} \{(\rho_X(g)x_i, \rho_Y(g)y_i) \mid i = 1, \ldots, N\}$$

For instance for blood cells:

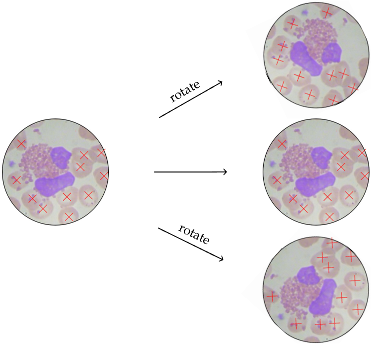


Data augmentation

In ***data augmentation***, we train on an enlarged training dataset:

$$\mathcal{T} = \{(x_i, y_i) \mid i = 1, \dots, N\} \qquad \rightarrow \qquad \mathcal{T} = \bigcup_{g \in G} \{(\rho_X(g)x_i, \rho_Y(g)y_i) \mid i = 1, \dots, N\}$$

For instance for blood cells:



Goal: Investigate data augmentation theoretically.
What are the symmetry properties of neural networks trained with augmentation?

Strategy

Consider infinitely wide neural networks

Consider infinitely wide neural networks

- In this limit, the mean output μ_t after training time t can be computed analytically

neural tangent kernel

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

learning rate

test point

train data

train labels

Strategy

Consider infinitely wide neural networks

- In this limit, the mean output μ_t after training time t can be computed analytically
- Training data explicit $\Rightarrow$ can argue about training with augmented data

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

augmented data

augmented labels

Strategy

Consider infinitely wide neural networks

- In this limit, the mean output μ_t after training time t can be computed analytically
- Training data explicit $\Rightarrow$ can argue about training with augmented data
- Compute the mean output on a transformed sample

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

augmented data

augmented labels

Strategy

Consider infinitely wide neural networks

- In this limit, the mean output μ_t after training time t can be computed analytically
- Training data explicit $\Rightarrow$ can argue about training with augmented data
- Compute the mean output on a transformed sample

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

augmented data

augmented labels

- The mean prediction corresponds to an ensemble prediction

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

Background: Neural Tangent Kernels and Wide Neural Networks

Empirical NTK

- Consider continuous gradient descent

$$\frac{\mathrm{d}\theta_\mu}{\mathrm{d}t} = -\eta \frac{\partial \mathcal{L}(f_\theta, \mathcal{D})}{\partial \theta_\mu}$$

Empirical NTK

- Consider continuous gradient descent

$$\frac{\mathrm{d}\theta_\mu}{\mathrm{d}t} = -\eta \frac{\partial \mathcal{L}(f_\theta, \mathcal{D})}{\partial \theta_\mu}$$

- Then, the network evolves according to

$$\frac{\mathrm{d}f_\theta(x)}{\mathrm{d}t} = -\frac{\eta}{N} \sum_{i=1}^{N} \sum_{\mu} \frac{\partial f(x)}{\partial \theta_\mu} \frac{\partial f(x_i)}{\partial \theta_\mu} \frac{\partial l(f_\theta(x_i), y_i)}{\partial f}$$

Empirical NTK

- Consider continuous gradient descent

$$\frac{\mathrm{d}\theta_\mu}{\mathrm{d}t} = -\eta \frac{\partial \mathcal{L}(f_\theta, \mathcal{D})}{\partial \theta_\mu}$$

- Then, the network evolves according to

$$\frac{\mathrm{d}f_\theta(x)}{\mathrm{d}t} = -\frac{\eta}{N} \sum_{i=1}^{N} \sum_{\mu} \frac{\partial f(x)}{\partial \theta_\mu} \frac{\partial f(x_i)}{\partial \theta_\mu} \frac{\partial l(f_\theta(x_i), y_i)}{\partial f}$$

- Hence, the training is driven by the ***empirical neural tangent kernel (NTK)***

$$\Theta^{\theta}_{ij}(x, x') = \sum_{\mu} \frac{\partial f_i(x)}{\partial \theta_\mu} \frac{\partial f_j(x')}{\partial \theta_\mu}$$

NTK at initialization

Deterministic NTK [Jacot et al. 2020]

When taking the layer widths to infinity sequentially, the empirical NTK $\Theta^{\theta}_{ij}(x, x')$ at initialization converges in probability to a deterministic kernel $\Theta(x, x')\delta_{ij}$

NTK at initialization

Deterministic NTK [Jacot et al. 2020]

When taking the layer widths to infinity sequentially, the empirical NTK $\Theta_{ij}^{\theta}(x, x')$ at initialization converges in probability to a deterministic kernel $\Theta(x, x')\delta_{ij}$

- Due to law of large numbers

NTK at initialization

Deterministic NTK [Jacot et al. 2020]

When taking the layer widths to infinity sequentially, the empirical NTK $\Theta^{\theta}_{ij}(x, x')$ at initialization converges in probability to a deterministic kernel $\Theta(x, x')\delta_{ij}$

- Due to law of large numbers
- The deterministic kernel is given in terms of a recursion over layers

NTK at initialization

Deterministic NTK [Jacot et al. 2020]

When taking the layer widths to infinity sequentially, the empirical NTK $\Theta^{\theta}_{ij}(x, x')$ at initialization converges in probability to a deterministic kernel $\Theta(x, x')\delta_{ij}$

- Due to law of large numbers
- The deterministic kernel is given in terms of a recursion over layers
- For most common architectures, this recursion can be performed explicitly, e.g. using `neural-tangents` Python package [Novak et al. 2020]

Frozen NTK

In NTK parametrization:

Freezing of NTK [Jacot et al. 2020]

For a nonlinearity which is Lipschitz, twice differentiable and has bounded second derivative,

$$\Theta_{ij}^{\theta_t}(x, x') \to \Theta(x, x')\delta_{ij}$$

uniformly in t as the layer widths go to infinity sequentially.

Frozen NTK

In NTK parametrization:

Freezing of NTK [Jacot et al. 2020]

For a nonlinearity which is Lipschitz, twice differentiable and has bounded second derivative,

$$\Theta_{ij}^{\theta_t}(x, x') \to \Theta(x, x')\delta_{ij}$$

uniformly in t as the layer widths go to infinity sequentially.

- Intuitively, this happens because the weight updates vanish in the limit $n \to \infty$

Frozen NTK

In NTK parametrization:

Freezing of NTK [Jacot et al. 2020]

For a nonlinearity which is Lipschitz, twice differentiable and has bounded second derivative,

$$\Theta_{ij}^{\theta_t}(x, x') \to \Theta(x, x')\delta_{ij}$$

uniformly in t as the layer widths go to infinity sequentially.

- Intuitively, this happens because the weight updates vanish in the limit $n \to \infty$
- However, the network still learns because the number of neurons grows, leading to a non-zero collective effect

Gradient descent at inifinite width

[Jacot et al. 2020]

- At infinite width, continuous gradient descent training under the MSE loss is given by

$$\frac{\mathrm{d}f_{\theta_t}(x)}{\mathrm{d}t} = -\eta \sum_{i=1}^{N} \Theta(x, x_i)(f_\theta(x_i) - y_i)$$

Gradient descent at inifinite width

[Jacot et al. 2020]

- At infinite width, continuous gradient descent training under the MSE loss is given by

$$\frac{\mathrm{d}f_{\theta_t}(x)}{\mathrm{d}t} = -\eta \sum_{i=1}^{N} \Theta(x, x_i)(f_\theta(x_i) - y_i)$$

- This ODE can be solved analytically, resulting in

$$f_{\theta_t}(x) = \Theta(x, X)\Theta(X, X)^{-1}(e^{-\eta\Theta(X,X)t} - \mathbb{1})(f_{\theta_0}(X) - Y) + f_{\theta_0}(x)$$

Gradient descent at inifinite width

[Jacot et al. 2020]

- At infinite width, continuous gradient descent training under the MSE loss is given by

$$\frac{\mathrm{d} f_{\theta_t}(x)}{\mathrm{d}t} = -\eta \sum_{i=1}^{N} \Theta(x, x_i)(f_\theta(x_i) - y_i)$$

- This ODE can be solved analytically, resulting in

$$f_{\theta_t}(x) = \Theta(x, X)\Theta(X, X)^{-1}(e^{-\eta\Theta(X,X)t} - \mathbb{1})(f_{\theta_0}(X) - Y) + f_{\theta_0}(x)$$

- At initialization, infinitely wide neural networks f_{θ_0} are zero-mean GPs with covariance function $K(x, x')$ (NNGP)

[Neal 1995
Lee et al. 2018
Matthews et al. 2018]

Gradient descent at inifinite width

[Jacot et al. 2020]

- At infinite width, continuous gradient descent training under the MSE loss is given by

$$\frac{\mathrm{d} f_{\theta_t}(x)}{\mathrm{d}t} = -\eta \sum_{i=1}^{N} \Theta(x, x_i)(f_\theta(x_i) - y_i)$$

- This ODE can be solved analytically, resulting in

$$f_{\theta_t}(x) = \Theta(x, X)\Theta(X, X)^{-1}(e^{-\eta\Theta(X,X)t} - \mathbb{1})(f_{\theta_0}(X) - Y) + f_{\theta_0}(x)$$

- At initialization, infinitely wide neural networks f_{θ_0} are zero-mean GPs with covariance function $K(x, x')$ (NNGP)

[Neal 1995
Lee et al. 2018
Matthews et al. 2018]

- As a linear combination of the GPs f_{θ_0}, the prediction $f_{\theta(t)}(x)$ is a GP

Gradient descent at inifinite width

[Jacot et al. 2020]

- At infinite width, continuous gradient descent training under the MSE loss is given by

$$\frac{\mathrm{d} f_{\theta_t}(x)}{\mathrm{d}t} = -\eta \sum_{i=1}^{N} \Theta(x, x_i)(f_\theta(x_i) - y_i)$$

- This ODE can be solved analytically, resulting in

$$f_{\theta_t}(x) = \Theta(x, X)\Theta(X, X)^{-1}(e^{-\eta\Theta(X,X)t} - \mathbb{1})(f_{\theta_0}(X) - Y) + f_{\theta_0}(x)$$

- At initialization, infinitely wide neural networks f_{θ_0} are zero-mean GPs with covariance function $K(x, x')$ (NNGP)

[Neal 1995
Lee et al. 2018
Matthews et al. 2018]

- As a linear combination of the GPs f_{θ_0}, the prediction $f_{\theta(t)}(x)$ is a GP
- The mean function is given by

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

Gradient descent at inifinite width

[Jacot et al. 2020]

- At infinite width, continuous gradient descent training under the MSE loss is given by

$$\frac{\mathrm{d} f_{\theta_t}(x)}{\mathrm{d}t} = -\eta \sum_{i=1}^{N} \Theta(x, x_i)(f_\theta(x_i) - y_i)$$

- This ODE can be solved analytically, resulting in

$$f_{\theta_t}(x) = \Theta(x, X)\Theta(X, X)^{-1}(e^{-\eta\Theta(X,X)t} - \mathbb{1})(f_{\theta_0}(X) - Y) + f_{\theta_0}(x)$$

- At initialization, infinitely wide neural networks f_{θ_0} are zero-mean GPs with covariance function $K(x, x')$ (NNGP)

[Neal 1995 Lee et al. 2018 Matthews et al. 2018]

- As a linear combination of the GPs f_{θ_0}, the prediction $f_{\theta(t)}(x)$ is a GP
- The mean function is given by

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

- The covariance function is given by

$$\Sigma_t(x, x') = K(x, x') + \Theta(x, X)\,\Theta^{-1}\,(\mathbb{1} - e^{-\eta\Theta t})\,K\,(\mathbb{1} - e^{-\eta\Theta t})\,\Theta^{-1}\,\Theta(X, x')$$
$$- \left(\Theta(x, X)\,\Theta^{-1}\,(\mathbb{1} - e^{-\eta\Theta t})\,K(X, x') + \text{h.c.}\right)$$

Emergent Equivariance for Large-Width Deep Ensembles

Kernel transformation

Consider the transformation of the kernels on arbitrary inputs

$$K(x, x') \rightarrow K(\rho_X(g)x, \rho_X(g)x')$$
$$\Theta(x, x') \rightarrow \Theta(\rho_X(g)x, \rho_X(g)x')$$

Kernel transformation

Consider the transformation of the kernels on arbitrary inputs

$$K(x, x') \to K(\rho_X(g)x, \rho_X(g)x')$$
$$\Theta(x, x') \to \Theta(\rho_X(g)x, \rho_X(g)x')$$

Kernel transformation

The neural tangent kernel Θ as well as the NNGP kernel K transform according to

$$\Theta(\rho_X(g)x, \rho_X(g)x') = \rho_K(g)\Theta(x, x')\rho_K^\top(g)\,,$$
$$K(\rho_X(g)x, \rho_X(g)x') = \rho_K(g)K(x, x')\rho_K^\top(g)\,,$$

for all $g \in G$ and $x, x' \in X$, where ρ_K is a transformation acting on the spatial dimensions of the kernels. If the kernels do not have spatial axes, $\rho_K = \mathbb{1}$.

Kernel transformation

Consider the transformation of the kernels on arbitrary inputs

$$K(x, x') \longrightarrow K(\rho_X(g)x, \rho_X(g)x')$$
$$\Theta(x, x') \longrightarrow \Theta(\rho_X(g)x, \rho_X(g)x')$$

Kernel transformation

The neural tangent kernel Θ as well as the NNGP kernel K transform according to

$$\Theta(\rho_X(g)x, \rho_X(g)x') = \rho_K(g)\Theta(x, x')\rho_K^\top(g)\,,$$
$$K(\rho_X(g)x, \rho_X(g)x') = \rho_K(g)K(x, x')\rho_K^\top(g)\,,$$

for all $g \in G$ and $x, x' \in X$, where ρ_K is a transformation acting on the spatial dimensions of the kernels. If the kernels do not have spatial axes, $\rho_K = \mathbb{1}$.

- Prove inductively over layers
- For nonlinearities, CNN-, fully-connected- and flattening layers

Permutation shift

- Under data augmentation

$$\rho_X(g)x_i = x_{\pi_g(i)}\,, \qquad \rho_Y(g)y_i = y_{\pi_g(i)}\,, \qquad \pi \in S_N$$

Permutation shift

- Under data augmentation

$$\rho_X(g)x_i = x_{\pi_g(i)}\,, \qquad \rho_Y(g)y_i = y_{\pi_g(i)}\,, \qquad \pi \in S_N$$

Permutation shift

Data augmentation implies that the permutation group action Π commutes with any matrix-valued analytical function F involving the Gram matrices of the NNGP and NTK as well as their inverses:

$$\begin{aligned}&\Pi(g)F(\Theta,\Theta^{-1},K,K^{-1})\\ &=\rho_K(g)F(\Theta,\Theta^{-1},K,K^{-1})\Pi(g)\rho_K^\top(g)\,.\end{aligned}$$

Permutation shift

- Under data augmentation

$$\rho_X(g)x_i = x_{\pi_g(i)}\,, \qquad \rho_Y(g)y_i = y_{\pi_g(i)}\,, \qquad \pi \in S_N$$

Permutation shift

Data augmentation implies that the permutation group action Π commutes with any matrix-valued analytical function F involving the Gram matrices of the NNGP and NTK as well as their inverses:

$$\begin{aligned}&\Pi(g)F(\Theta,\Theta^{-1},K,K^{-1})\\ &=\rho_K(g)F(\Theta,\Theta^{-1},K,K^{-1})\Pi(g)\rho_K^\top(g)\,.\end{aligned}$$

- Proof permutation shift separately for Θ, Θ^{-1}, K, K^{-1} and all powers of these

Invariance of ensemble of MLPs

- Consider an ensemble of MLPs trained with data augmentation towards invariance

Invariance of ensemble of MLPs

- Consider an ensemble of MLPs trained with data augmentation towards invariance
- The mean prediction on a transformed test sample is given by

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

Invariance of ensemble of MLPs

- Consider an ensemble of MLPs trained with data augmentation towards invariance
- The mean prediction on a transformed test sample is given by

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

- By kernel transformation in the MLP case

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

Invariance of ensemble of MLPs

- Consider an ensemble of MLPs trained with data augmentation towards invariance
- The mean prediction on a transformed test sample is given by

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

- By kernel transformation in the MLP case

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

- By permutation shift

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

Invariance of ensemble of MLPs

- Consider an ensemble of MLPs trained with data augmentation towards invariance
- The mean prediction on a transformed test sample is given by

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

- By kernel transformation in the MLP case

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

- By permutation shift

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

- Due to data augmentation, the labels are invariant under group-permutations

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

Emergent equivariance of deep ensembles

Emergent equivariance of deep ensembles

The distribution of large-width ensemble members $f_\theta : X \to Y$ *is equivariant* with respect to the representations ρ_X and ρ_Y of the group G *if data augmentation is applied*. In particular, the ensemble prediction

$$\bar{f}_t(x) = \mathbb{E}_{\text{initializations}}[f_\theta(x)]$$

is equivariant,

$$\bar{f}_t(\rho_X(g)\,x) = \rho_Y(g)\,\bar{f}_t(x)\,,$$

for all $g \in G$. This result holds

1. at any training time t,
2. for any element of the input space $x \in X$.

- Prove by showing equivariance of μ_t and Σ_t

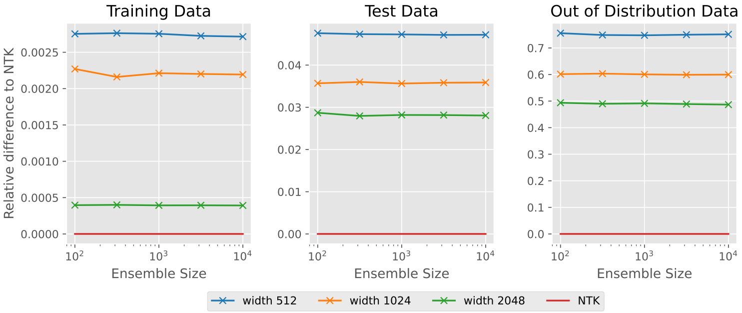
Experiments

Ising model: convergence to the NTK

- Consider the 2d Ising model
- Symmetry: Energy is invariant under C_4 lattice rotations
- Train MLP ensembles with data augmentation and compute NTK exactly

Ising model: convergence to the NTK

- Consider the 2d Ising model
- Symmetry: Energy is invariant under C_4 lattice rotations
- Train MLP ensembles with data augmentation and compute NTK exactly
- For growing width, the MLP ensemble-predictions converge to the NTK predictions



Ising model: emergent invariance

- Measure ***relative orbit standard deviation***

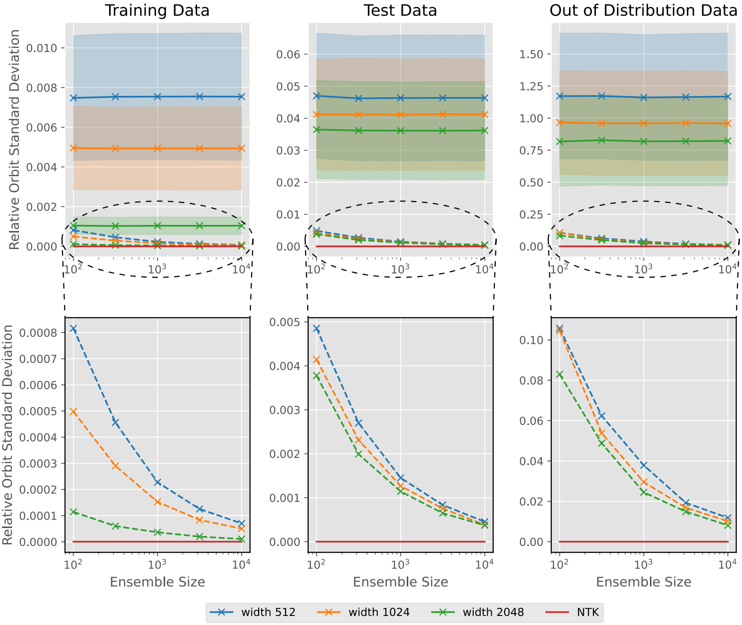
$$\frac{\mathrm{std}_{g \in C_4} \mathcal{E}(\{s_{\rho(g)i}\})}{\mathrm{mean}_{g \in C_4} \mathcal{E}(\{s_{\rho(g)i}\})}$$

Ising model: emergent invariance

- Measure ***relative orbit standard deviation***

$$\frac{\mathrm{std}_{g\in C_4}\mathcal{E}(\{s_{\rho(g)i}\})}{\mathrm{mean}_{g\in C_4}\mathcal{E}(\{s_{\rho(g)i}\})}$$

Ising model: emergent invariance



FashionMNIST: emergent invariance

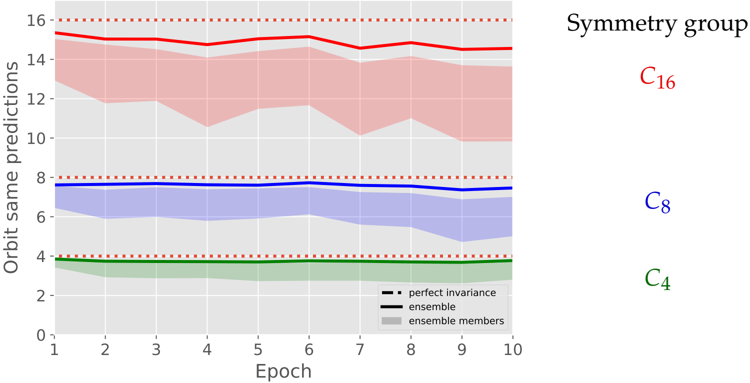
- Train ensembles of CNNs on FashionMNIST augmented by C_k (multiples of $360^\circ/k$) with $k = 4, 8, 16$

FashionMNIST: emergent invariance

- Train ensembles of CNNs on FashionMNIST augmented by C_k (multiples of $360^\circ/k$) with $k = 4, 8, 16$
- Measure invariance using ***orbit same predictions***: number of predictions in the orbit which agree with the prediction on untransformed sample

FashionMNIST: emergent invariance

- Train ensembles of CNNs on FashionMNIST augmented by C_k (multiples of $360°/k$) with $k = 4, 8, 16$
- Measure invariance using ***orbit same predictions***: number of predictions in the orbit which agree with the prediction on untransformed sample
- Throughout training, the ensemble predictions are more invariant than the predictions of the ensemble members, even out of distribution:



Conclusion

Conclusions

Summary

- Under data augmentation, ensemble predictions become exactly equivariant in the large width limit
- This equivariance holds even out of distribution and at any training time
- We show this by explicitly computing the transformation properties of the neural tangent kernel under data augmentation

Conclusions

Summary

- Under data augmentation, ensemble predictions become exactly equivariant in the large width limit
- This equivariance holds even out of distribution and at any training time
- We show this by explicitly computing the transformation properties of the neural tangent kernel under data augmentation

Application

- If you need an ensemble, consider data augmentation instead of manifestly equivariant models
- If you need data augmentation, consider an ensemble to boost equivariance

Emergent Equivariance in Deep Ensembles

arXiv: 2403.03103

Thank you!