Equivariant Neural Tangent Kernels

Philipp Misof [a]   Pan Kessel [b]   Jan E. Gerken [a]

Abstract

Little is known about the training dynamics of equivariant neural networks, in particular how it compares to data augmented training of their non-equivariant counterparts. Recently, neural tangent kernels (NTKs) have emerged as a powerful tool to analytically study the training dynamics of wide neural networks. In this work, we take an important step towards a theoretical understanding of training dynamics of equivariant models by deriving neural tangent kernels for a broad class of equivariant architectures based on group convolutions. As a demonstration of the capabilities of our framework, we show an interesting relationship between data augmentation and group convolutional networks. Specifically, we prove that they share the same expected prediction at all training times and even off-manifold. In this sense, they have the same training dynamics. We demonstrate in numerical experiments that this still holds approximately for finite-width ensembles. By implementing equivariant NTKs for roto-translations in the plane ($G = C_n \ltimes \mathbb{R}^2$) and 3d rotations ($G = \mathrm{SO}(3)$), we show that equivariant NTKs outperform their non-equivariant counterparts as kernel predictors for histological image classification and quantum mechanical property prediction.

1 Introduction

Equivariant neural networks [1, 2] are widely used in many applications of great practical importance, for example in medical image analysis in two and three dimensions [3, 4, 5, 6] and in quantum chemistry [7, 8, 9, 10]. Other application areas include particle physics [11], cosmology [12] and even fairness in large language models [13].

Recently, there has been a number of works which avoid equivariant architectures but rely on data augmentation to approximately learn equivariance, most notably AlphaFold3 [14]. This has the potential advantage that non-equivariant architectures may offer better training dynamics, for example favorable scaling capabilities. There has been a vigorous debate on this subject with some empirical works claiming superiority of equivariant architectures [15, 16] while others suggest the opposite [17, 14]. One challenging aspect to conclusively settle the matter is that there is no good theoretical understanding of how the equivariant and the purely augmentation-based training dynamics compare.

Motivated by this observation, this paper derives equivariant *neural tangent kernel (NTK)* theory [18] for group convolutional architectures. The NTK provides a powerful tool to analytically study the training dynamics of neural networks in the large width limit by analyzing the behavior of the kernel, in particular its trace, eigenvalues and other properties [19, 20, 21, 22]. A particularly important feature of the NTK is the fact that in the infinite width limit, it becomes constant throughout training [18]. Furthermore, at infinite width, the NTK can be computed by layer-wise recursion relations. These simplifications allow for complete analytic control over the training dynamics. In particular, the network output of an

[a] Department of Mathematical Sciences, Chalmers University of Technology and the University of Gothenburg, SE-412 96 Gothenburg, Sweden.

[b] Prescient Design, Genentech Roche, Basel, Switzerland.

Emails: misof@chalmers.se, pan.kessel@roche.com, gerken@chalmers.se

arbitrary input and at an arbitrary point in training time converges to a Gaussian process over initializations whose mean- and covariance functions can be computed analytically. This result has led to a number of theoretical and practical insights [19, 23, 24, 25, 26, 27] into the initialization and training of neural networks.

We derive recursive relations which determine the NTK of an equivariant neural networks for the first time. In particular, we study the NTK of group convolutional layers [28]. These layers are in some sense universal. Specifically, they have the unique property that they arise from imposing an equivariance constraint on dense fully-connected layers and are therefore the most general linear, equivariant transformations [29] and have been used in a wide array of applications [30, 31, 32].

These NTK recursion allows us to clarify the relation between the training dynamics of pure data augmentation and equivariant architectures in the large width limit. Specifically, non-equivariant architectures trained with full data augmentation converge to certain group convolutional architectures in the infinite width limit. This results holds for any input, in particular off-manifold, and at any training time. Thus, at least in the infinite width limit, the expected training dynamics of data augmentation is identical to the one of certain group convolutional architectures.

NTKs have also been shown to be interesting kernel functions in their own right. Since they are induced by neural network architectures, they allow to transfer the intuition gained in the extensive literature on the design of neural networks to kernel machines and have shown to outperform more traditional kernel functions [33, 34, 35]. In our experiments, we show that group equivariant kernels outperform their non-equivariant counterparts for both regression and classification as well as for discrete roto-translations and continuous rotations.

In summary, our main contributions are

- We derive layer-wise recursive relations for the neural tangent kernel and neural network Gaussian process kernel of group convolutional layers, the corresponding lifting layers, point-wise nonlinearities and group-pooling layers.
- We specialize our general results to the case of roto-translations in the plane as well as the three-dimensional rotation group $\mathrm{SO}(3)$. We derive and implement the kernel relations for these cases, allowing for efficient computations.
- We proof that in the infinite width limit, a standard convolutional or fully connected network trained with full data augmentation yields the same network function as a corresponding group convolutional network trained without data augmentation. This result holds for all training times as well as off manifold. We show empirically that this holds approximately at finite width.
- We verify experimentally that the NTKs of finite-width equivariant networks converge to our equivariant NTKs as width grows to infinity. Furthermore, we demonstrate the superior performance of equivariant NTKs over other kernels for medical image classification and quantum mechanical property prediction.

2 Related Work

Neural Tangent Kernel. Gaussian processes can be viewed as Bayesian neural networks as first pointed out by [36] and this relation extends to deep neural networks as shown in [37]. Neural tangent kernels allow description of training dynamics, see the seminal reference [18] and [38] for an accessible review. In [39], NTK theory was used to show that wide neural networks trained with gradient descent become Gaussian processes and generalized in a more rigorous and systematic manner by [40]. NTKs can be used to derive parametrizations that allow scaling networks to large width [24]. They can also be used to theoretically analyze GANs [26], PINNs [41], backdoor attacks [42], and pruning [43]. Recently, corrections to infinite width limit have been studied by [44, 45, 46, 47] using techniques inspired by perturbative quantum field theory. The NTK kernel for convolutional architectures was derived in [33]. Our results can be thought as a generalization thereof to general group convolutions. In [48], the effect of data augmentation on infinitely wide neural networks was studied. The authors found that the resulting Gaussian process is equivariant at all training times and even off the data manifold. In contrast, our results to not require data augmentation but derive an NTK for manifestly equivariant group convolution layers.

Equivariant Neural Networks. Equivariance has been an important theme of deep learning research over the last years, see [2] for an accessible review. Equivariant deep learning is part of the larger area of geometric deep learning [49], in which more general geometric properties of the different parts of the learning problem (e.g. the data [50], model [1] and optimization procedure [51]) are studied. Herein, we focus on group convolutional layers [28] which are the unique linear equivariant layers. They have found wide-spread application in computer vision [30, 31, 32], medical applications [3, 30, 6] as well as natural science use cases [52, 53].

3 Background

This section gives a brief overview of NTK theory as well as of equivariant neural networks with a particular emphasis on group convolutional neural networks (GCNNs).

Neural tangent kernels. The frozen NTK (see Appendix A for an introduction) can be computed analytically by layer-wise recursive relations [18] starting from the definition

$$\Theta^{(\ell)}(x,x') = \mathbb{E}\left[\sum_{\ell'=1}^{\ell} \frac{\partial \mathcal{N}^{(\ell)}(x)}{\partial \theta^{(\ell')}} \left(\frac{\partial \mathcal{N}^{(\ell)}(x')}{\partial \theta^{(\ell')}}\right)^{\top}\right] \tag{1}$$

of the layer-ℓ NTK. The NTK of the full network is given by $\Theta(x,x') = \Theta^{(L)}(x,x')$ for a network depth L. Here, $\theta^{(\ell)}$ are the parameters of the layer ℓ and we adopt the convention that expectation values are over the initialization distribution unless otherwise stated. As customary in the NTK literature, we treat activations and preactivations as distinct layers and refer to $\mathcal{N}^{(\ell)}$ as the layer-ℓ features with $\mathcal{N}(x) = \mathcal{N}^{(L)}(x)$. This allows us to treat linear- and nonlinear layers on an equal footing. Since (1) is proportional to the unit matrix, we can treat it as a scalar.

We can find a recursion relation between $\Theta^{(\ell+1)}$ and $\Theta^{(\ell)}$ by separating the $\ell' = \ell + 1$ contribution from the sum and computing the $\ell' \leq \ell$ contributions in terms of derivatives through the layer $\ell + 1$ using the chain rule,

$$\begin{aligned}\Theta^{(\ell+1)}(x,x') = \mathbb{E}&\left[\frac{\partial \mathcal{N}^{(\ell+1)}(x)}{\partial \theta^{(\ell+1)}} \left(\frac{\partial \mathcal{N}^{(\ell+1)}(x')}{\partial \theta^{(\ell+1)}}\right)^{\top}\right] \\ &+ \mathbb{E}\left[\frac{\partial \mathcal{N}^{(\ell+1)}(x)}{\partial \mathcal{N}^{(\ell)}(x)} \underbrace{\left(\sum_{\ell'=1}^{\ell} \frac{\partial \mathcal{N}^{(\ell)}(x)}{\partial \theta^{(\ell')}} \left(\frac{\partial \mathcal{N}^{(\ell)}(x')}{\partial \theta^{(\ell')}}\right)^{\top}\right)}_{\Theta^{(\ell)}(x,x')} \times \left(\frac{\partial \mathcal{N}^{(\ell+1)}(x)}{\partial \mathcal{N}^{(\ell)}(x)}\right)^{\top}\right].\end{aligned} \tag{2}$$

Note that according to the NTK's definition (1), it holds that $\Theta^{(0)} = 0$. The recursions (2) have been computed explicitly for a number of layers, e.g. fully connected [18], nonlinear [18], convolution [33], and graph convolution [54]. An efficient implementation for many layers is available in the Jax-based Python package `neural-tangents` [55].

For evaluating the expectation values in (2), it is convenient to introduce the *neural network Gaussian process (NNGP)* kernel

$$K^{(\ell)}(x,x') = \mathbb{E}\left[\mathcal{N}^{(\ell)}(x)\,\left(\mathcal{N}^{(\ell)}(x')\right)^{\top}\right], \tag{3}$$

whose name originates in the fact that at initialization, the neural network converges in the infinite width limit to a zero-mean Gaussian process with covariance function $K^{(L)}(x,x')$ [36, 37]. In the infinite width limit, K is proportional to the unit matrix, so we will treat it as a scalar as well. The NNGP can also be computed recursively layer-by-layer. For the $\ell = 0$, the NNGP is the covariance matrix of the input features $K^{(0)}(x,x') = x\,x'^{\top}$.

Using the definition (3) of the NNGP, we can determine the structure of the NTK recursive relations from (2). For linear layers, the first expectation value will evaluate to the NNGP, while the second expectation value will be proportional to the unit matrix due to the initialization with independent normally distributed parameters. For nonlinear layers, the first expectation value vanishes and the second expectation values will depend on the derivative of the nonlinearity.

Group convolutions. Group convolutions [28] act on feature maps $f : G \longrightarrow \mathbb{R}^{n_{\text{in}}}$ where n_{in} denotes the number of input features to the network. In the example of image inputs, this feature map would be $f : \mathbb{Z}^2 \longrightarrow \mathbb{R}^3$ where $\mathbb{Z}^2$ is the pixel grid, $\mathbb{R}^3$ is the space of RGB colors, and the feature map f is supported on $[0,h] \times [0,w]$ for imagesize $h \times w$. Let $L^2(X,Y)$ denote the set of square integrable functions from X to Y. The ℓ-th neural network layer $\mathcal{N}^{(\ell)} : L^2(G,\mathbb{R}^{n_{\text{in}}}) \longrightarrow L^2(G,\mathbb{R}^{n_\ell})$ maps an input feature map $f : G \longrightarrow \mathbb{R}^{n_{\text{in}}}$ to an output feature map $\mathcal{N}^{(\ell)}(f) : G \longrightarrow \mathbb{R}^{n_\ell}$. A particular instance of such a layer is the group convolution layer which in NTK representation is given by

$$[\mathcal{N}^{(\ell+1)}(f)](g) = \frac{1}{\sqrt{n_\ell |S_\kappa|}} \int_G \mathrm{d}h\; \kappa\big(g^{-1}h\big)[\mathcal{N}^{(\ell)}(f)](h)\,, \tag{4}$$

with filter $\kappa : G \longrightarrow \mathbb{R}^{n_\ell, n_{\ell+1}}$ with support $S_\kappa \subset G$. Here, we integrate over the group with respect to the Haar measure. For finite groups, the integral becomes a sum over group elements. Due to the invariance of the Haar measure, the layers (4) are equivariant with respect to the regular representation

$$(\rho_{\text{reg}}(g)f)(h) = f(g^{-1}h) \qquad g,h \in G\,. \tag{5}$$

Since the input features typically have domain $X \subseteq \mathbb{R}^{n_{\text{in}}}$ which is not the symmetry group G, the first layer of a *group convolutional neural network (GCNN)* is a *lifting layer* which maps a feature map with domain X equivariantly into a feature map with domain G [28]

$$[\mathcal{N}^{(1)}(f)](g) = \frac{1}{\sqrt{n_{\text{in}} |S_\kappa|}} \int_X \mathrm{d}x\; \kappa\big(\rho(g^{-1})x\big) f(x)\,, \tag{6}$$

where ρ is a representation of G on X. We assume here and in the following that X is a homogeneous space of G, i.e. that any two elements of X are connected by a group transformation.

As is common for other network types as well, the nonlinearities in group convolutional networks are applied component-wise across the different group elements,

$$[\mathcal{N}^{(\ell+1)}(f)](g) = \sigma\big([\mathcal{N}^{(\ell)}(f)](g)\big) \tag{7}$$

for nonlinearity σ. Due to this component-wise structure, the layers (7) are equivariant with respect to the regular representation (5) as well.

By combining lifting- and group-convolution layers with nonlinearities, one can construct expressive architectures which are equivariant with respect to the regular representation, i.e. which satisfy

$$\mathcal{N}(\rho_{\mathrm{reg}}(g)f) = \rho_{\mathrm{reg}}(g)\mathcal{N}(f)\,, \qquad g \in G\,. \tag{8}$$

Many practical applications necessitate an invariant network

$$\mathcal{N}(\rho_{\mathrm{reg}}(g)f) = \mathcal{N}(f)\,, \qquad g \in G\,. \tag{9}$$

Such a transformation property can be achieved by appending a *group pooling layer* to a GCNN,

$$\mathcal{N}^{(\ell+1)}(f) = \frac{1}{\mathrm{vol}(G)} \int_G \mathrm{d}g\; [\mathcal{N}^{(\ell)}(f)](g)\,. \tag{10}$$

Using these layers, a wide variety of equivariant- and invariant networks with respect to a general symmetry group G can be easily constructed.

4 Equivariant neural tangent kernels

This section presents our recursive relations for the NTK and the NNGP for group convolutional layers. These recursions allow for efficient calculation of these kernels for arbitrary group convolutional architectures and thus provide the necessary tools to analytically study their training dynamics in the large width limit. Specifically, we derive recursion relations for group convolutions (4), lifting layers (6) and group pooling layers (10) by evaluating the derivatives and expectation values in (2).

4.1 Equivariant NTK for group convolutions

Since the domain of the feature maps in GCNNs is the symmetry group G, the layer-ℓ NNGP and NTK kernels do not only depend on the input feature maps f and f' but also on the group elements g, g' at which the feature maps are evaluated, i.e.,

$$K^{(\ell)}_{g,g'}(f,f') = \mathbb{E}\left[[\mathcal{N}^{(\ell)}(f)](g) \left([\mathcal{N}^{(\ell)}(f')](g') \right)^{\top} \right], \tag{11}$$

$$\Theta^{(\ell)}_{g,g'}(f,f') = \mathbb{E}\left[\sum_{\ell'=1}^{\ell} \frac{\partial [\mathcal{N}^{(\ell)}(f)](g)}{\partial \theta^{(\ell')}} \left(\frac{\partial [\mathcal{N}^{(\ell)}(f')](g')}{\partial \theta^{(\ell')}} \right)^{\top} \right]. \tag{12}$$

For these kernels, we derive the following recursion relation:

Theorem 1 (Kernel recursions for group convolutional layers). *The layer-wise recursive relations for the NNGP and NTK of the group convolutional layer* (4) *are given by*

$$K^{(\ell+1)}_{g,g'}(f,f') = \frac{1}{|S_\kappa|} \int_{S_\kappa} \mathrm{d}h\; K^{(\ell)}_{gh,g'h}(f,f') \tag{13}$$

$$\Theta^{(\ell+1)}_{g,g'}(f,f') = K^{(\ell+1)}_{g,g'}(f,f') + \frac{1}{|S_\kappa|} \int_{S_\kappa} \mathrm{d}h\, \Theta^{(\ell)}_{gh,g'h}(f,f')\,. \tag{14}$$

Proof. See Appendix B.

Given G-invariant filter supports S_κ, these recursive definitions imply an invariance of the kernels in their group-indices under right-multiplication by the same group element $h \in G$,

$$K^{(\ell+1)}_{gh,g'h}(f,f') = K^{(\ell+1)}_{g,g'}(f,f') \tag{15}$$

$$\Theta^{(\ell+1)}_{gh,g'h}(f,f') = \Theta^{(\ell+1)}_{g,g'}(f,f')\,. \tag{16}$$

While the kernels of feature maps on the group carry g, g'-indices, the kernels of the input features carry x, x'-indices,

$$K^{(0)}_{x,x'}(f, f') = f(x)f'(x')\,, \qquad \Theta^{(0)}_{x,x'}(f, f') = 0\,. \tag{17}$$

Using this, we also derive the following recursion relations:

Theorem 2 (Kernel recursions for the lifting layer). *The layer-wise recursive relations for the NNGP and NTK of the lifting layer* (6) *are given by*

$$K^{(\ell+1)}_{g,g'}(f, f') = \frac{1}{|S_\kappa|}\int_{S_\kappa} \mathrm{d}x\, K^{(\ell)}_{\rho(g)x,\rho(g')x}(f, f')\,, \tag{18}$$

$$\Theta^{(\ell+1)}_{g,g'}(f, f') = \frac{1}{|S_\kappa|}\int_{S_\kappa} \mathrm{d}x\ \Theta^{(\ell)}_{\rho(g)x,\rho(g')x}(f, f') + K^{(\ell+1)}_{g,g'}(f, f')\,, \tag{19}$$

where the regular representation ρ_{reg} is defined in (5).

Proof. See Appendix B.

The group pooling layer (10) maps feature maps on G onto channel-vectors. Therefore, the kernels lose their g, g'-indices in this layer, as is reflected in the following result:

Theorem 3 (Kernel recursions for group pooling layer). *The layer-wise recursive relations for the NNGP and NTK of the group pooling layer* (10) *are given by*

$$K^{(\ell+1)}(f, f') = \frac{1}{(\mathrm{vol}(G))^2}\int_G \mathrm{d}g \int_G \mathrm{d}g'\ K^{(\ell)}_{g,g'}(f, f') \tag{20}$$

$$\Theta^{(\ell+1)}(f, f') = \frac{1}{(\mathrm{vol}(G))^2}\int_G \mathrm{d}g \int_G \mathrm{d}g'\ \Theta^{(\ell)}_{g,g'}(f, f')\,. \tag{21}$$

Proof. See Appendix B.

The final layer necessary to compute kernels of GCNNs are the nonlinearities (7). Since these act pointwise on the feature maps, the recursive relations are the same as those for nonlinearities in MLPs [18]:

Corollary 4 (Kernel recursions for nonlinearities). *The layer-wise recursive relations for the NNGP and NTK of the nonlinear layer* (7) *are given by*

$$\Lambda^{(\ell)}_{g,g'}(f, f') = \begin{pmatrix} K^{(\ell)}_{g,g}(f, f) & K^{(\ell)}_{g,g'}(f, f') \\ K^{(\ell)}_{g',g}(f', f) & K^{(\ell)}_{g',g'}(f', f') \end{pmatrix} \tag{22}$$

$$K^{(\ell+1)}_{g,g'}(f, f') = \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\Lambda^{(\ell)}_{g,g'}(f,f'))}[\sigma(u)\sigma(v)] \tag{23}$$

$$\dot{K}^{(\ell+1)}_{g,g'}(f, f') = \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\Lambda^{(\ell)}_{g,g'}(f,f'))}[\sigma'(u)\sigma'(v)] \tag{24}$$

$$\Theta^{(\ell+1)}_{g,g'}(f, f') = \dot{K}^{(\ell+1)}_{g,g'}(f, f')\Theta^{(\ell)}_{g,g'}(f, f')\,. \tag{25}$$

Using these results, the NTK and NNGP can be straightforwardly computed for any GCNN architecture. In particular, consider the transformation of the kernels under transformations of the inputs, i.e. consider $K(\rho_{\text{reg}}(h)f, \rho_{\text{reg}}(h')f')$ and $\Theta(\rho_{\text{reg}}(h)f, \rho_{\text{reg}}(h')f')$. From the recursion relations for the lifting layer in Theorem 2, we have the transformation property

$$K^{(1)}_{g,g'}(\rho_{\text{reg}}(h)f, \rho_{\text{reg}}(h')f') = K^{(1)}_{h^{-1}g,h'^{-1}g'}(f, f') \tag{26}$$

$$\Theta^{(1)}_{g,g'}(\rho_{\text{reg}}(h)f, \rho_{\text{reg}}(h')f') = \Theta^{(1)}_{h^{-1}g,h'^{-1}g'}(f, f')\,. \tag{27}$$

This left-multiplication is preserved by the recursions of both the group convolutions in Theorem 1 and the nonlinearities in Corollary 4. Therefore, before any pooling layer, we have

$$K^{(\ell)}_{g,g'}(\rho_{\text{reg}}(h)f, \rho_{\text{reg}}(h')f') = K^{(\ell)}_{h^{-1}g,h'^{-1}g'}(f, f') \tag{28}$$

$$\Theta^{(\ell)}_{g,g'}(\rho_{\text{reg}}(h)f, \rho_{\text{reg}}(h')f') = \Theta^{(\ell)}_{h^{-1}g,h'^{-1}g'}(f, f')\,, \tag{29}$$

reflecting the equivariance of the network. The recursions of the group-pooling layer in Theorem 3 average over the group and the kernels become invariant after the group pooling layer

$$K^{(\ell)}(\rho_{\text{reg}}(h)f, \rho_{\text{reg}}(h')f') = K^{(\ell)}(f, f') \tag{30}$$

$$\Theta^{(\ell)}(\rho_{\text{reg}}(h)f, \rho_{\text{reg}}(h')f') = \Theta^{(\ell)}(f, f')\,, \tag{31}$$

as expected from an invariant network. Note that these transformation properties of the kernels are independent for both arguments.

4.2 Roto-translations in the plane

The kernel recursions provided in the previous section are valid for general symmetry groups G. In this section, we will specialize these expressions to the case of roto-translations in the plane with rotations by $(360/n)°$. In this case, $G = C_n \ltimes \mathbb{R}^2$ where G is the semidirect product of the cyclic group C_n and the translation group in two dimensions $\mathbb{R}^2$. It was shown that adding this rotational symmetry to conventional CNNs boosts performance considerably for important applications such as medical image analysis [30, 3, 6]. Due to the semidirect product nature of the symmetry group, the group convolutional layers can be written as a stack of n conventional convolutions which are summed over the rotation group. Details and explicit expressions for the lifting-, group convolutional- and group pooling layers in this case can be found in Appendix C.1.

The kernel recursion of ordinary CNN-layers can be written in terms of the operator [56]

$$[\mathcal{A}_{S_\kappa}(K)](t, t') = \frac{1}{|S_\kappa|}\int_{S_\kappa} \mathrm{d}\tilde{t}\; K(t + \tilde{t}, t' + \tilde{t})\,, \tag{32}$$

for which efficient implementations in terms of convolutions are available in [55]. In Appendix C.2, we present explicit expressions for the NNGP and NTK recursions of roto-translation equivariant convolutions in terms of $\mathcal{A}$, retaining the efficiency of the non-rotation-equivariant kernel computations. We provide implementations of these recursions for $n = 4$ as new layers based on the `neural-tangents` package.

4.3 Rotations in 3d

Spherical signals subject to rotations in 3d are a further important use case with numerous applications in quantum chemistry [7], weather prediction [57] and 3d shape recognition [58]. The group convolutions for the corresponding symmetry group SO(3) can be computed efficiently in the Fourier domain, in terms of coefficients in a steerable basis of spherical harmonics Y^l_m or Wigner matrices $\mathcal{D}^l_{mn}$, respectively [59, 60]. Due to the continuous nature of the SO(3) group, comprehensive data augmentation is not feasible, thus making group convolutional networks the natural choice to incorporate such symmetries. In Appendix D.1, we provide a summary of the necessary Fourier space relations for SO(3)-equivariant networks.

The kernel forward equations (13), (14) simplify to purely algebraic equations in terms of Fourier coefficients

$$\left[\widehat{K^{(\ell)}(f, f')}\right]^{l,l'}_{mn,m'n'} = \int \mathrm{d}R \int \mathrm{d}R'\; K^{(\ell)}_{R,R'}(f, f')\mathcal{D}^l_{mn}(R)\mathcal{D}^{l'}_{m'n'}(R')\,, \tag{33}$$

and analogously for the NTK. Note that the kernels have two group indices, thus necessitating a double Fourier transform. Detailed relations for lifting-, group convolutional- and group pooling layer in the Fourier space are provided in Appendix D.2. Again, these layers are implemented in the `neural-tangents` package and the necessary generalized FFTs are provided by the JAX-based package `s2fft` [61].

5 Data augmentation versus group convolutions at infinite width

The recursive relations presented in the previous sections give analytical access to the training dynamics of equivariant neural networks. In particular, they allow for a more in-depth theoretical understanding of the similarities and differences of data augmentation and manifest equivariance than previously possible.

It is known that ensembles of neural networks trained with data augmentation yield equivariant mean predictions [48, 62]. It is however unclear how these equivariant functions relate to trained manifestly equivariant networks. Using the recursive relations from Section 4.1, it is possible to show that non-equivariant networks trained with data augmentation in fact converge to group convolutional networks in the ensemble mean.

5.1 Data augmentation at infinite width

In the infinite width limit, the training dynamics under gradient descent can be solved exactly [18]. This enables us to explicitly study data augmentation, showing that data augmentation and kernel averaging yield the same mean predictions, as detailed in the following

Theorem 5. *Let μ_t^{aug} and μ_t be the mean predictions after t training steps of infinite ensembles of two neural network architectures $\mathcal{N}^{\text{aug}}$ and $\mathcal{N}$. Let $\mathcal{N}^{\text{aug}}$ be trained on the fully G-augmented training data of $\mathcal{N}$ and assume that the NTKs of the two architectures are related by*

$$\Theta(f, f') = \frac{1}{\text{vol}(G)} \int_G \text{d}g\ \Theta^{\text{aug}}(f, \rho_{\text{reg}}(g) f') \,. \tag{34}$$

Then, μ_t^{aug} and μ_t converge in the infinite width limit to the same function for all t for quadratic losses, up to quadratic corrections in the learning rate.

Proof. See Appendix E.

The proof of Theorem 5 proceeds inductively over training steps. At initialization, both mean functions are identically zero [36, 37]. In the infinite width limit, the training updates can be written in terms of the NTK. The updates for the two networks can then be shown to agree by splitting the sum over augmented training data into a sum over samples and a sum over G and using the assumption (34).

5.2 Kernel averaging yields GCNN-kernels

Theorem 5 shows equivalence of augmented and non-augmented networks if the NTKs of both architectures are related by group-averaging. Consider the case of training an MLP on augmented data. Then, (34) prompts us to consider the group-average of its NTK to find the architecture which results in the same mean predictions if trained without data augmentation. By iterating the recursive kernel-relations found in the previous section, one can in fact show that this architecture is a GCNN, as detailed in the following

Theorem 6. *Let $\mathcal{N}^{\mathrm{FC}}$ be an MLP acting on feature maps with output in $\mathbb{R}$ and architecture*

$$\mathcal{N}^{\mathrm{FC}} = \mathrm{FC}^{(L)} \circ \sigma \circ \cdots \circ \mathrm{FC}^{(3)} \circ \sigma \circ \mathrm{FC}^{(1)}\,, \tag{35}$$

where FC denotes a dense MLP layer and σ a point-wise nonlinearity. Let $\mathcal{N}^{\mathrm{GC}}$ be a G-invariant GCNN with architecture

$$\mathcal{N}^{\mathrm{GC}} = \mathrm{GPool} \circ \mathrm{GConv}(S_\kappa^L) \circ \sigma \circ \mathrm{GConv}(S_\kappa^{L-2}) \circ \sigma \circ \cdots \circ \mathrm{GConv}(S_\kappa^3) \circ \sigma \circ \mathrm{Lifting}(S_\kappa^1)\,, \tag{36}$$

where S_κ^ℓ are the supports of the convolutional filters with $S_\kappa^1 = X$, the domain of the input feature maps, and the other S_κ^ℓ are invariant under G. Then, the G-averages of the kernels of the MLP are given by the kernels of the GCNN,

$$K^{\mathrm{GC}}(f, f') = \frac{1}{\mathrm{vol}(G)} \int \mathrm{d}g\ K^{\mathrm{FC}}(f, \rho_{\mathrm{reg}}(g) f') \tag{37}$$

$$\Theta^{\mathrm{GC}}(f, f') = \frac{1}{\mathrm{vol}(G)} \int \mathrm{d}g\ \Theta^{\mathrm{FC}}(f, \rho_{\mathrm{reg}}(g) f')\,. \tag{38}$$

Proof. See Appendix E.

Together with Theorem 5, this theorem shows that by augmenting an MLP at infinite width, one obtains a specific equivariant architecture, namely a GCNN with the same depth and an additional group-pooling layer. This result singles out group convolutional layers among other equivariant layers and mirrors the fact that group convolutions are the unique linear equivariant layers under the regular representation. Note that according to Theorem 5 the equivalence between augmented and equivariant networks holds throughout training and even out of distribution.

5.3 Augmenting a CNN

Consider a generalization of the roto-translation symmetry discussed in Section 4.2, namely a general semidirect product group, $G = K \ltimes N$ with N a normal subgroup of G. For N a translation group, this covers cases such as CNNs in two and three dimensions with additional rotation or reflection symmetry [63].

The semidirect product structure of G allows a splitting of the full equivariance, namely training a $K \ltimes N$-invariant GCNN is equivalent to training an N-invariant GCNN on K-augmented data. In order to see this, we show the corresponding kernel averages for Theorem 5 to hold:

Theorem 7. *Let $\mathcal{N}^{K\ltimes N}$ be the $K \ltimes N$-invariant GCNN with architecture (36) and K-invariant filter supports S_κ^ℓ which for the GConv-layers decompose as $S_\kappa^\ell = K_\kappa^\ell \times N_\kappa^\ell$, $K_\kappa^\ell \subseteq K$, $N_\kappa^\ell \subseteq N$. Let $\mathcal{N}^N$ be the N-invariant GCNN with architecture (36) and filter supports $N_\kappa^L, \ldots, N_\kappa^3$ and S_κ^1. Then, the NNGPs and NTKs of these networks are related by*

$$K^{K\ltimes N}(f, f') = \frac{1}{\mathrm{vol}(K)} \int_K \mathrm{d}k\ K^N(f, \rho_{\mathrm{reg}}(k) f') \tag{39}$$

$$\Theta^{K\ltimes N}(f, f') = \frac{1}{\mathrm{vol}(K)} \int_K \mathrm{d}k\ \Theta^N(f, \rho_{\mathrm{reg}}(k) f')\,. \tag{40}$$

Proof. See Appendix E.

Remark 8. For $K = C_n$ and $N = \mathbb{R}^2$, i.e. roto-translations in the plane, $\mathcal{N}^N$ becomes an ordinary CNN and $\mathcal{N}^{K \ltimes N}$ is of the form discussed in Section 4.2. According to Theorem 7, the kernels of the rotation-equivariant network are then given by averaging the kernels of the CNN,

$$K^{C_n \ltimes \mathbb{R}^2}(f, f') = \frac{1}{n} \sum_{r \in C_n} K^{\mathrm{CNN}}(f, \rho_{\mathrm{reg}}(r) f') \tag{41}$$

$$\Theta^{C_n \ltimes \mathbb{R}^2}(f, f') = \frac{1}{n} \sum_{r \in C_n} \Theta^{\mathrm{CNN}}(f, \rho_{\mathrm{reg}}(r) f') \tag{42}$$

if the spatial filter shapes agree for both networks and are rotation-invariant.

Taken togenther with Theorem 5, this shows that training an N-invariant GCNN on K-augmented data results in a $K \ltimes N$-invariant GCNN. For the special case of N a translation group and K a rotation group, this means that training a CNN on rotation-augmented data is equivalent to training a roto-translation equivariant GCNN on unaugmented data. In Section 6, we will show that this still holds approximately for finite-width networks and ensembles.

5.4 Distribution of ensemble members

Consider training two ensembles of networks with (a) data augmentation on a non-equivariant architecture and (b) no data augmentation on an equivariant architecture. Then, the distributions of the individual networks in these ensembles do not agree since most of the augmented networks will not be equivariant. However, our results show that the ensemble mean of (a) converges to the ensemble mean of (b) with a specific GCNN architecture. This establishes a highly non-trivial relation between data augmentation and GCNNs.

6 Experiments

In the following, we validate the theoretical results of the preceding sections experimentally for various datasets (Cifar10, QM9, MNIST, and histological data), tasks (regression and classification), and groups (SO(3) and $C_4 \ltimes \mathbb{R}^2$).

Kernel convergence for $C_4 \ltimes \mathbb{R}^2$. Figure 1 confirms that the Monte-Carlo estimate of the NTK converges to our analytical expression as the width increases. Our MC estimates are obtained by replacing the expectation values in (1) by the sample mean of 1000 initializations of the network. We considered GCNNs with one lifting- and four group-convolution layers interspersed with ReLU nonlinearites, followed by a group-pooling layer. The convergence of the NNGP is shown in a similar plot in Appendix F.1.

Medical image classification with $C_4 \ltimes \mathbb{R}^2$. We show that rotation-equivariant NTK-predictors outperform non-equivariant NTK-predictors on a dataset of histological images [64] containing nine distinct classes of tissues. Specifically, we compare a CNN architecture with the corresponding rotation-invariant GCNN architecture, in which we replace each of the five convolutional layers with a group-convolutional- or lifting layer, respectively and used a group-pooling layer instead of a `SumPool` layer. Figure 3 shows the improved scaling behavior of the equivariant kernel with training set size upon using the infinite-time solution of the NTK-dynamics under MSE loss,

$$\mu(x) = \Theta(x, \mathcal{X})\Theta(\mathcal{X}, \mathcal{X})^{-1}\mathcal{Y}\,, \tag{43}$$

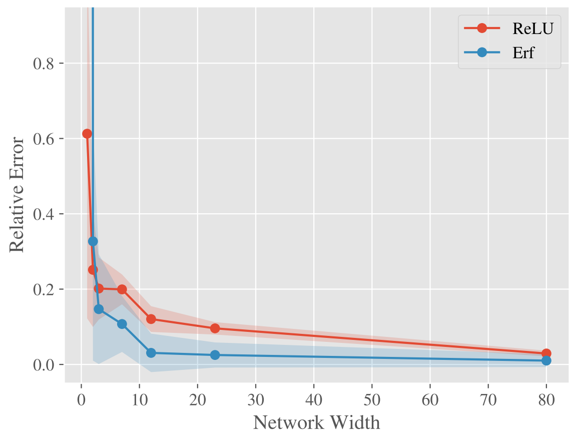

Figure 1: **Convergence of the Monte-Carlo estimates of the NTK to their infinite-width limits for $G = C_4 \ltimes \mathbb{R}^2$.** Plotted is the relative error averaged over the components of a 3×3 Gram matrix for networks with a ReLU or an error function nonlinearity. Bands show ± one standard deviation of the estimator.

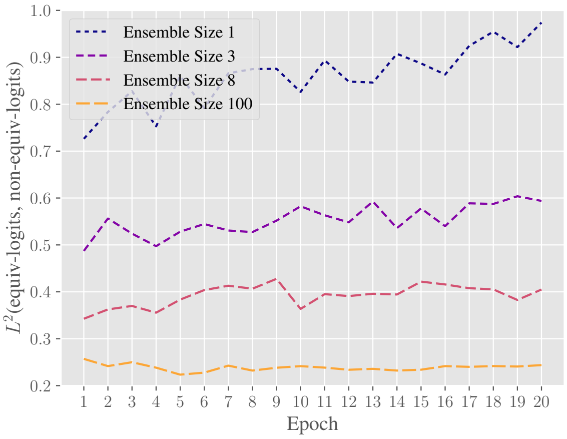

Figure 2: **Convergence of finite-width ensembles trained with data augmentation to ensembles of GCNNs on MNIST.** L^2-distance between the logits of the equivariant- and non-equivariant ensemble trained with data augmentation for different ensemble sizes on out of distribution data. For larger ensembles, the distance decreases.

where $\mathcal{X}$ represents the training images, scaled down to 32×32 pixels, $\mathcal{Y}$ are the training labels, given by $\boldsymbol{e}_c - \frac{1}{9}\mathbf{1}$ for class c, and $\Theta(\mathcal{X}, \mathcal{X})$ is the Gram matrix of the NTK. We refer to Appendix F.2 for more details.

Molecular energy regression with $\mathrm{SO}(3)$**.** We benchmark the NTK-predictor resulting from an $\mathrm{SO}(3)$-invariant network on the *QM9* dataset [65] by predicting molecular energies U_0 from atom configurations utilizing (43). Comparing this to the corresponding MLP kernel, we observe a considerable performance boost for the invariant kernel over a range of training set sizes, as shown in Figure 4. For preprocessing, we construct spherical signals from the atom configurations as described in [66]. For each atom i of the at most 29 atoms, the environment is represented by pairwise Gaussian smearing over atoms with the same atomic number z

$$f_{i,z,p}(x) = \sum_{j: z_j = z} \frac{z_i z}{\|r_{ij}\|^p} e^{-\frac{1}{\beta}\left(\frac{r_{ij}}{\|r_{ij}\|} \cdot x - 1\right)^2} . \tag{44}$$

Choosing $p \in \{2, 6\}$ and considering all of QM9's five atom types leads to 29 spherical per-atom signals with 5×2 channels each. Each of those per-atom signals are then either processed by a two layer $\mathrm{SO}(3)$-equivariant network with group pooling on top or by a two-layer MLP. The per-atom outputs are eventually summed and fed into a final fully-connected layer similarly as in [59]. The input signals are constructed at a resolution of 12×11 on the sphere, corresponding to a bandlimit of $L = 6$, which are then downsampled to a bandlimit $L = 3$ for the group layer. We provide further details in Appendix F.3.

Data augmentation versus group convolutions at finite width. In Section 5, we proved that networks trained with data augmentation converge to group convolutional networks at infinite width. We verify that this also holds approximately at finite width. To this end, we train ensembles of CNNs and GCNNs with symmetry group $C_4 \ltimes \mathbb{R}^2$ as discussed in Remark 8 on CIFAR10 and MNIST using the MSE-loss against smoothed one-hot labels as for the medical images above. For implementing the GCNNs, we used the `escnn`-package [67]. As shown in Figure 2 for MNIST, the outputs of both ensembles converge to the same vector for large ensemble sizes throughout training and even out of distribution. For further details on the model architectures, out of distribution data, as well as results on CIFAR10 see Appendix F.4.

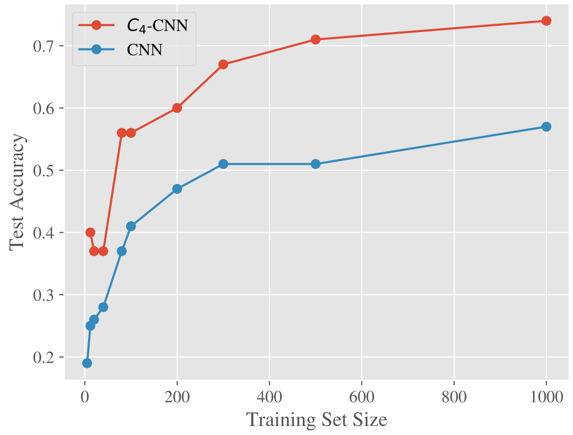

Figure 3: **NTK for image classification.** Test accuracy of the arising NTK kernel methods in the infinite width and infinite training time limit for different training set sizes. The results for both a conventional CNN and a $C_4 \ltimes \mathbb{R}^2$-invariant GCNN are shown.

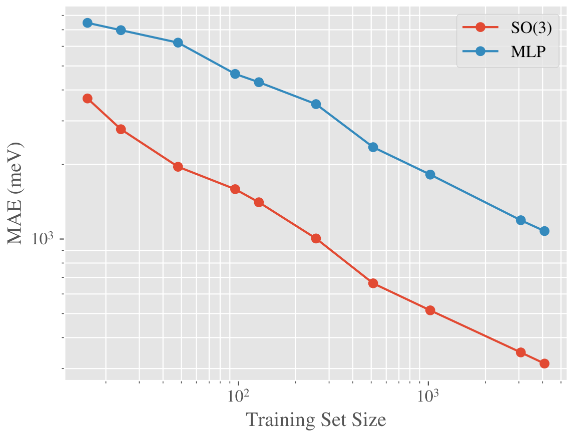

Figure 4: **NTK for molecular energy prediction.** Molecular energy MAEs of the NTK kernel methods in the infinite width and training time limit for different training set sizes. The results are for both a conventional MLP and a SO(3)-invariant GCNN.

7 Conclusion

This paper provides recursive relations for the NNGP and NTK for group convolutional neural networks allowing us to theoretically establish an interesting equivalence between equivariance-based to data-augmentation-based training dynamics. We also show that equivariant kernels outperform their non-equivariant counterparts as kernel machines.

Acknowledgments

J.G. and P.M. want to thank Max Guillen for inspiring discussions and collaborations on related projects. The work of J.G. and P.M. is supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The computations were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) and the Swedish National Infrastructure for Computing (SNIC) at C3SE partially funded by the Swedish Research Council through grant agreements 2022-06725 and 2018-05973.

References

[1] Maurice Weiler et al. *Equivariant and Coordinate Independent Convolutional Networks. A Gauge Field Theory of Neural Networks*. 2023. URL: https://maurice-weiler.gitlab.io/cnn_book/EquivariantAndCoordinateIndependentCNNs.pdf.

[2] Jan E. Gerken et al. "Geometric Deep Learning and Equivariant Neural Networks". In: *Artificial Intelligence Review* (June 2023). ISSN: 1573-7462. DOI: 10.1007/s10462-023-10502-7. arXiv: 2105.13926.

[3] Erik J. Bekkers et al. "Roto-Translation Covariant Convolutional Networks for Medical Image Analysis". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2018*. Ed. by Alejandro F. Frangi et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 440–448. ISBN: 978-3-030-00928-1. DOI: 10.1007/978-3-030-00928-1_50.

[4] Marysia Winkels and Taco S. Cohen. “Pulmonary Nodule Detection in CT Scans with Equivariant CNNs”. In: *Medical Image Analysis* 55 (July 2019), pp. 15–26. ISSN: 1361-8415. DOI: 10.1016/j.media.2019.03.010. arXiv: 1804.04656.

[5] Philip Müller et al. *Rotation-Equivariant Deep Learning for Diffusion MRI*. Feb. 2021. arXiv: 2102.06942.

[6] Shuchao Pang et al. “Beyond CNNs: Exploiting Further Inherent Symmetries in Medical Image Segmentation”. In: *IEEE Transactions on Cybernetics* 53.11 (Nov. 2023), pp. 6776–6787. ISSN: 2168-2275. DOI: 10.1109/TCYB.2022.3195447. arXiv: 2207.14472.

[7] Alexandre Duval et al. *A Hitchhiker’s Guide to Geometric GNNs for 3D Atomic Systems*. Dec. 2023. arXiv: 2312.07511.

[8] Simon Batzner et al. “E(3)-Equivariant Graph Neural Networks for Data-Efficient and Accurate Interatomic Potentials”. In: *Nature Communications* 13.1 (May 2022), p. 2453. ISSN: 2041-1723. DOI: 10.1038/s41467-022-29939-5. arXiv: 2101.03164.

[9] Kristof Schütt, Oliver Unke, and Michael Gastegger. “Equivariant Message Passing for the Prediction of Tensorial Properties and Molecular Spectra”. In: *Proceedings of the 38th International Conference on Machine Learning*. PMLR, July 2021, pp. 9377–9388. arXiv: 2102.03150.

[10] Oliver Unke et al. “SE(3)-Equivariant Prediction of Molecular Wavefunctions and Electronic Densities”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 14434–14447. arXiv: 2106.02347.

[11] Alexander Bogatskiy et al. “Lorentz Group Equivariant Neural Network for Particle Physics”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, 2020, pp. 992–1002. arXiv: 2006.04780.

[12] Nathanaël Perraudin et al. “DeepSphere: Efficient Spherical Convolutional Neural Network with HEALPix Sampling for Cosmological Applications”. In: *Astronomy and Computing* 27 (Apr. 2019), pp. 130–146. ISSN: 2213-1337. DOI: 10.1016/j.ascom.2019.03.004. arXiv: 1810.12186.

[13] Sourya Basu et al. “Equi-Tuning: Group Equivariant Fine-Tuning of Pretrained Models”. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 37. June 2023, pp. 6788–6796. DOI: 10.1609/aaai.v37i6.25832. arXiv: 2210.06475.

[14] Josh Abramson et al. “Accurate structure prediction of biomolecular interactions with AlphaFold 3”. In: *Nature* (2024), pp. 1–3.

[15] Jan E. Gerken et al. “Equivariance versus Augmentation for Spherical Images”. In: *Proceedings of the 39th International Conference on Machine Learning*. PMLR, June 2022, pp. 7404–7421. arXiv: 2202.03990.

[16] Johann Brehmer et al. “Does equivariance matter at scale?” In: *arXiv preprint arXiv:2410.23179* (2024).

[17] Yuyang Wang et al. “Swallowing the Bitter Pill: Simplified Scalable Conformer Generation”. In: *Forty-first International Conference on Machine Learning 2024*. 2024.

[18] Arthur Jacot, Franck Gabriel, and Clement Hongler. “Neural Tangent Kernel: Convergence and Generalization in Neural Networks”. In: *Advances in Neural Information Processing Systems*. Vol. 31. Curran Associates, Inc., 2018. arXiv: 1806.07572.

[19] Mario Geiger et al. “Disentangling Feature and Lazy Training in Deep Neural Networks”. In: *Journal of Statistical Mechanics: Theory and Experiment* 2020.11 (Nov. 2020), p. 113301. ISSN: 1742-5468. DOI: 10.1088/1742-5468/abc4de. arXiv: 1906.08034.

[20] Jisoo Mok et al. “Demystifying the Neural Tangent Kernel from a Practical Perspective: Can It Be Trusted for Neural Architecture Search without Training?” In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2022, pp. 11851–11860. DOI: 10.1109/CVPR52688.2022.01156. arXiv: 2203.14577.

[21] Andrew William Engel et al. “Faithful and Efficient Explanations for Neural Networks via Neural Tangent Kernel Surrogate Models”. In: *The Twelfth International Conference on Learning Representations*. Oct. 2023. arXiv: 2305.14585.

[22] Che-Ping Tsai, Chih-Kuan Yeh, and Pradeep Ravikumar. “Sample Based Explanations via Generalized Representers”. In: *Advances in Neural Information Processing Systems*. Ed. by A. Oh et al. Vol. 36. Curran Associates, Inc., 2023, pp. 23485–23498. arXiv: 2310.18526.

[23] Arthur Jacot et al. *Order and Chaos: NTK Views on DNN Normalization, Checkerboard and Boundary Artifacts*. June 2020. arXiv: 1907.05715.

[24] Greg Yang and Edward J. Hu. “Tensor Programs IV: Feature Learning in Infinite-Width Neural Networks”. In: *Proceedings of the 38th International Conference on Machine Learning*. Ed. by Marina Meila and Tong Zhang. Vol. 139. Proceedings of Machine Learning Research. PMLR, 2021, pp. 11727–11737. arXiv: 2011.14522.

[25] Ge Yang et al. “Tuning Large Neural Networks via Zero-Shot Hyperparameter Transfer”. In: *Advances in Neural Information Processing Systems*. Vol. 34. Curran Associates, Inc., 2021, pp. 17084–17097. arXiv: 2203.03466.

[26] Jean-Yves Franceschi et al. "A Neural Tangent Kernel Perspective of GANs". In: *Proceedings of the 39th International Conference on Machine Learning*. PMLR, June 2022, pp. 6660–6704. arXiv: 2106.05566.

[27] Hannah Day, Yonatan Kahn, and Daniel A. Roberts. *Feature Learning and Generalization in Deep Networks with Orthogonal Weights*. Oct. 2023. arXiv: 2310.07765.

[28] Taco Cohen and Max Welling. "Group Equivariant Convolutional Networks". In: *Proceedings of The 33rd International Conference on Machine Learning*. PMLR, June 2016, pp. 2990–2999. arXiv: 1602.07576.

[29] Risi Kondor and Shubhendu Trivedi. "On the Generalization of Equivariance and Convolution in Neural Networks to the Action of Compact Groups". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, 2018, pp. 2747–2755. arXiv: 1802.03690.

[30] Benjamin Chidester et al. "Enhanced Rotation-Equivariant U-Net for Nuclear Segmentation". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. June 2019, pp. 1097–1104. DOI: 10.1109/CVPRW.2019.00143.

[31] Elena Celledoni et al. "Equivariant Neural Networks for Inverse Problems". In: *Inverse Problems* 37.8 (July 2021), p. 085006. ISSN: 0266-5611. DOI: 10.1088/1361-6420/ac104f.

[32] Daniel Moyer et al. "Equivariant Filters for Efficient Tracking in 3D Imaging". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2021*. Ed. by Marleen de Bruijne et al. Cham: Springer International Publishing, 2021, pp. 193–202. ISBN: 978-3-030-87202-1. DOI: 10.1007/978-3-030-87202-1_19.

[33] Sanjeev Arora et al. "On Exact Computation with an Infinitely Wide Neural Net". In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. arXiv: 1904.11955.

[34] Zhiyuan Li et al. *Enhanced Convolutional Neural Tangent Kernels*. Nov. 2019. arXiv: 1911.00809.

[35] Jaehoon Lee et al. "Finite versus Infinite Neural Networks: An Empirical Study". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 15156–15172. arXiv: 2007.15801.

[36] Radford M. Neal. *Bayesian Learning for Neural Networks*. Springer Science & Business Media, 1996. ISBN: 978-1-4612-0745-0.

[37] Jaehoon Lee et al. "Deep Neural Networks as Gaussian Processes". In: *International Conference on Learning Representations*. Feb. 2018. arXiv: 1711.00165.

[38] Eugene Golikov, Eduard Pokonechnyy, and Vladimir Korviakov. "Neural Tangent Kernel: A Survey". In: arXiv:2208.13614 (Aug. 2022). arXiv: 2208.13614.

[39] Jaehoon Lee et al. "Wide Neural Networks of Any Depth Evolve as Linear Models Under Gradient Descent". In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., 2019. DOI: 10.1088/1742-5468/abc62b. arXiv: 1902.06720.

[40] Greg Yang. "Scaling Limits of Wide Neural Networks with Weight Sharing: Gaussian Process Behavior, Gradient Independence, and Neural Tangent Kernel Derivation". In: *arXiv:1902.04760 [cond-mat, physics:math-ph, stat]* (Apr. 2020). arXiv: 1902.04760.

[41] Sifan Wang, Xinling Yu, and Paris Perdikaris. "When and Why PINNs Fail to Train: A Neural Tangent Kernel Perspective". In: *Journal of Computational Physics* 449 (Jan. 2022), p. 110768. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2021.110768.

[42] Jonathan Hayase and Sewoong Oh. "Few-Shot Backdoor Attacks via Neural Tangent Kernels". In: *The Eleventh International Conference on Learning Representations*. Sept. 2022. arXiv: 2210.05929.

[43] Hongru Yang and Zhangyang Wang. "On the Neural Tangent Kernel Analysis of Randomly Pruned Neural Networks". In: *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*. PMLR, Apr. 2023, pp. 1513–1553. arXiv: 2203.14328.

[44] Jiaoyang Huang and Horng-Tzer Yau. "Dynamics of Deep Neural Networks and Neural Tangent Hierarchy". In: *Proceedings of the 37th International Conference on Machine Learning*. PMLR, Nov. 2020, pp. 4542–4551. arXiv: 1909.08156.

[45] Sho Yaida. "Non-Gaussian Processes and Neural Networks at Finite Widths". In: *Proceedings of The First Mathematical and Scientific Machine Learning Conference*. PMLR, Aug. 2020, pp. 165–192. arXiv: 1910.00019.

[46] James Halverson, Anindita Maiti, and Keegan Stoner. "Neural Networks and Quantum Field Theory". In: *Machine Learning: Science and Technology* 2.3 (Sept. 2021), p. 035002. ISSN: 2632-2153. DOI: 10.1088/2632-2153/abeca3. arXiv: 2008.08601.

[47] Harold Erbin, Vincent Lahoche, and Dine Ousmane Samary. "Nonperturbative Renormalization for the Neural Network-QFT Correspondence". In: *Machine Learning: Science and Technology* 3.1 (Mar. 2022), p. 015027. ISSN: 2632-2153. DOI: 10.1088/2632-2153/ac4f69. arXiv: 2108.01403.

[48] Jan E. Gerken and Pan Kessel. “Emergent Equivariance in Deep Ensembles”. In: *Proceedings of the 41st International Conference on Machine Learning*. PMLR, July 2024. arXiv: 2403.03103.

[49] Michael M. Bronstein et al. “Geometric Deep Learning: Going beyond Euclidean Data”. In: *IEEE Signal Processing Magazine* 34.4 (July 2017), pp. 18–42. ISSN: 1053-5888, 1558-0792. DOI: 10.1109/MSP.2017.2693418. arXiv: 1611.08097.

[50] Ann-Kathrin Dombrowski et al. “Diffeomorphic Counterfactuals With Generative Models”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 46.5 (May 2024), pp. 3257–3274. ISSN: 1939-3539. DOI: 10.1109/TPAMI.2023.3339980.

[51] Shun-ichi Amari. “Natural Gradient Works Efficiently in Learning”. In: *Neural Computation* 10.2 (Feb. 1998), pp. 251–276. ISSN: 0899-7667. DOI: 10.1162/089976698300017746.

[52] Kim A Nicoli et al. “Asymptotically unbiased estimation of physical observables with neural samplers”. In: *Physical Review E* 101.2 (2020), p. 023304.

[53] Yi-Lun Liao and Tess Smidt. “Equiformer: Equivariant Graph Attention Transformer for 3D Atomistic Graphs”. In: *The Eleventh International Conference on Learning Representations*. Sept. 2022.

[54] Simon S. Du et al. “Graph Neural Tangent Kernel: Fusing Graph Neural Networks with Graph Kernels”. In: *Advances in Neural Information Processing Systems*. Vol. 32. Curran Associates, Inc., Nov. 2019. arXiv: 1905.13192.

[55] Roman Novak et al. “Neural Tangents: Fast and Easy Infinite Neural Networks in Python”. In: *Eighth International Conference on Learning Representations*. Apr. 2020. arXiv: 1912.02803.

[56] Lechao Xiao et al. “Dynamical Isometry and a Mean Field Theory of CNNs: How to Train 10,000-Layer Vanilla Convolutional Neural Networks”. In: *Proceedings of the 35th International Conference on Machine Learning*. PMLR, July 2018, pp. 5393–5402.

[57] Boris Bonev et al. “Spherical Fourier Neural Operators: Learning Stable Dynamics on the Sphere”. In: *Proceedings of the 40th International Conference on Machine Learning*. Ed. by Andreas Krause et al. Vol. 202. Proceedings of Machine Learning Research. PMLR, 23–29 Jul 2023, pp. 2806–2823. URL: https://proceedings.mlr.press/v202/bonev23a.html.

[58] Fabian Fuchs et al. “SE(3)-Transformers: 3D Roto-Translation Equivariant Attention Networks”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 1970–1981. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/15231a7ce4ba789d13b722cc5c955834-Paper.pdf.

[59] Taco S. Cohen et al. “Spherical CNNs”. In: *International Conference on Learning Representations*. Feb. 2018. arXiv: 1801.10130.

[60] Taco S. Cohen and Max Welling. “Steerable CNNs”. In: *International Conference on Learning Representations*. 2017. URL: https://openreview.net/forum?id=rJQKYt5ll.

[61] Matthew A. Price and Jason D. McEwen. “Differentiable and accelerated spherical harmonic and Wigner transforms”. In: *Journal of Computational Physics* 510 (2024), p. 113109. DOI: 10.1016/j.jcp.2024.113109. eprint: arXiv:2311.14670.

[62] Oskar Nordenfors and Axel Flinth. *Ensembles Provably Learn Equivariance through Data Augmentation*. Oct. 2024. arXiv: 2410.01452.

[63] Gabriele Cesa, Leon Lang, and Maurice Weiler. “A Program to Build E(N)-Equivariant Steerable CNNs”. In: *International Conference on Learning Representations*. Oct. 2021.

[64] Jakob Nikolas Kather, Niels Halama, and Alexander Marx. *100,000 histological images of human colorectal cancer and healthy tissue*. Version v0.1. Zenodo, Apr. 2018. DOI: 10.5281/zenodo.1214456.

[65] Raghunathan Ramakrishnan et al. “Quantum chemistry structures and properties of 134 kilo molecules”. In: *Scientific Data* 1 (2014).

[66] Carlos Esteves, Jean-Jacques Slotine, and Ameesh Makadia. “Scaling Spherical CNNs”. In: *Proceedings of the 40th International Conference on Machine Learning*. Ed. by Andreas Krause et al. Vol. 202. Proceedings of Machine Learning Research. PMLR, 23–29 Jul 2023, pp. 9396–9411. URL: https://proceedings.mlr.press/v202/esteves23a.html.

[67] Maurice Weiler and Gabriele Cesa. “General $E(2)$-Equivariant Steerable CNNs”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. arXiv: 1911.08251.

[68] Maurice Weiler, Fred A. Hamprecht, and Martin Storath. “Learning Steerable Filters for Rotation Equivariant CNNs”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 849–858. arXiv: 1711.07289.

[69] Jaehoon Lee et al. “Finite Versus Infinite Neural Networks: an Empirical Study”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 15156–15172.

[70] J.R. Driscoll and D.M. Healy. “Computing Fourier Transforms and Convolutions on the 2-Sphere”. In: *Advances in Applied Mathematics* 15.2 (1994), pp. 202–250. ISSN: 0196-8858. DOI: https://doi.org/10.1006/aama.1994.1008.

A Basics of neural tangent kernel theory

A neural network $\mathcal{N} : \mathbb{R}^{n_\text{in}} \to \mathbb{R}^{n_\text{out}}$ which is trained using continuous gradient descent

$$\frac{\mathrm{d}\theta}{\mathrm{d}t} = -\eta \frac{\partial \mathcal{L}}{\partial \theta} \tag{45}$$

on a loss function $\mathcal{L}$ with learning rate η evolves according to

$$\frac{\mathrm{d}\mathcal{N}(x)}{\mathrm{d}t} = -\eta \sum_{i=1}^{n_\text{train}} \Theta_t(x, x_i) \frac{\partial \mathcal{L}}{\partial \mathcal{N}(x_i)} \,, \tag{46}$$

where the sum runs over the training samples x_i and $\Theta_t \in \mathbb{R}^{n_\text{out} \times n_\text{out}}$ is the NTK

$$\Theta_t(x, x') = \frac{\partial \mathcal{N}(x)}{\partial \theta} \left(\frac{\partial \mathcal{N}(x')}{\partial \theta} \right)^\top \,. \tag{47}$$

For finite-width networks, Θ_t depends on the initialization and the training time and is referred to as the *empirical* NTK. At infinite width, Θ_t becomes independent of the initialization (it still depends on the initialization distribution) since it approaches its expectation value over initializations

$$\Theta(x, x') = \mathbb{E}_{\theta \sim p_\text{init}} \left[\frac{\partial \mathcal{N}(x)}{\partial \theta} \left(\frac{\partial \mathcal{N}(x')}{\partial \theta} \right)^\top \right] \,. \tag{48}$$

It furthermore becomes constant throughout training and proportional to the unit matrix [18] in the NTK parametrization. For this reason, we drop the t-subscript on this *frozen* NTK and treat it as a scalar. In the following, we will always mean (48) when we refer to the NTK unless otherwise stated. The NTK parametrization of a linear layer has an additional $1/\sqrt{n_\text{fan in}}$ prefactor and uses independent standard Gaussians as initialization distributions. Hence, an MLP layer is given by

$$\mathcal{N}^{(\ell)}(x) = \sigma\big(\mathcal{N}^{(\ell-1)}(x)\big) = \sigma\left(\frac{1}{\sqrt{n_\text{fan in}}} W \mathcal{N}^{(\ell-2)}(x) + b \right) \,, \tag{49}$$

with nonlinearity σ, weights W and bias b.

B Proofs: Kernel recursions for GCNN-layers

In this section, we provide proofs for the theorems given in Section 4.1 in the main text.

Theorem 1 (Kernel recursions for group convolutional layers). *The layer-wise recursive relations for the NNGP and NTK of the group convolutional layer* (4) *are given by*

$$K^{(\ell+1)}_{g,g'}(f,f') = \frac{1}{|S_\kappa|} \int_{S_\kappa} \mathrm{d}h \; K^{(\ell)}_{gh,g'h}(f, f') \tag{13}$$

$$\Theta^{(\ell+1)}_{g,g'}(f,f') = K^{(\ell+1)}_{g,g'}(f,f') + \frac{1}{|S_\kappa|} \int_{S_\kappa} \mathrm{d}h \, \Theta^{(\ell)}_{gh,g'h}(f,f') \,. \tag{14}$$

Proof. We first compute the NNGP recursion relation. For group-convolution layers, the definition (3) of the NNGP reads

$$\begin{aligned} K^{(\ell+1)}_{g,g'}(f, f') &= \mathbb{E}\left[[\mathcal{N}^{(\ell+1)}(f)](g) \, [\mathcal{N}^{(\ell+1)}(f')](g') \right] &(50)\\ &= \frac{1}{S_\kappa} \int_G \mathrm{d}h \, \mathrm{d}h' \; \mathbb{E}\left[\kappa^{(\ell+1)}(g^{-1}h) \kappa^{(\ell+1)}(g'^{-1}h') \right] \mathbb{E}\left[[\mathcal{N}^{(\ell)}(f)](h) [\mathcal{N}^{(\ell)}(f')](h') \right] \,, &(51) \end{aligned}$$

where we have again dropped the $1/\sqrt{n_\ell}$-prefactors and channel dependencies since these converge to the expectation value in the infinite width limit. Next, we shift the integration variables by g and g' which leaves the Haar measure invariant by its definition

$$K^{(\ell+1)}_{g,g'}(f,f') = \frac{1}{S_\kappa}\int_G \mathrm{d}h\,\mathrm{d}h'\;\mathbb{E}\left[\kappa^{(\ell+1)}(h)\kappa^{(\ell+1)}(h')\right]\mathbb{E}\left[[\mathcal{N}^{(\ell)}(f)](gh)[\mathcal{N}^{(\ell)}(f')](g'h')\right]\,. \tag{52}$$

Since the kernel components are sampled independently from standard Gaussians at initialization, we only obtain a contribution to the integral when $h = h'$ and $\kappa^{(\ell+1)}$ has support at this point, i.e.

$$K^{(\ell+1)}_{g,g'}(f,f') = \frac{1}{S_\kappa}\int_{S_\kappa} \mathrm{d}h\;\mathbb{E}\left[[\mathcal{N}^{(\ell)}(f)](gh)[\mathcal{N}^{(\ell)}(f')](g'h)\right]\,. \tag{53}$$

Comparing to (3) shows that the right-hand side is just the NNGP $K^{(\ell)}(f,f')$ of the previous layer evaluated at group indices gh and $g'h$. This proves the NNGP recursion relation stated in the theorem.

For the NTK recursion relation, we start by specializing the general expression (2) to group-convolution layers and adapting it to the functional framework used for feature maps

$$\begin{aligned}
&\Theta^{(\ell+1)}_{g,g'}(f,f')\\
&= \int_{S_\kappa}\mathrm{d}h\;\mathbb{E}\left[\frac{\delta[\mathcal{N}^{(\ell+1)}(f)](g)}{\delta\kappa^{(\ell+1)}(h)}\,\frac{\delta[\mathcal{N}^{(\ell+1)}(f')](g')}{\delta\kappa^{(\ell+1)}(h)}\right]\\
&+\int_G \mathrm{d}\tilde{g}\,\mathrm{d}\tilde{g}'\;\mathbb{E}\left[\frac{\delta[\mathcal{N}^{(\ell+1)}(f)](g)}{\delta[\mathcal{N}^{(\ell)}(f)](\tilde{g})}\underbrace{\left(\sum_{\ell'=1}^{\ell}\int_{S_\kappa}\mathrm{d}\tilde{h}\;\frac{\delta[\mathcal{N}^{(\ell)}(f)](\tilde{g})}{\delta\kappa^{(\ell')}(\tilde{h})}\,\frac{\delta\mathcal{N}^{(\ell)}(\tilde{g}')}{\delta\kappa^{(\ell')}(\tilde{h})}\right)}_{\Theta^{(\ell)}_{\tilde{g},\tilde{g}'}(f,f')}\frac{\delta[\mathcal{N}^{(\ell+1)}(f')](g')}{\delta[\mathcal{N}^{(\ell)}(f')](\tilde{g}')}\right]
\end{aligned} \tag{54}$$

According to the layer definition (4), the derivatives evaluate to

$$\frac{\delta[\mathcal{N}^{(\ell+1)}(f)](g)}{\delta\kappa^{(\ell+1)}(h)} = \frac{1}{\sqrt{n_\ell S_\kappa}}[\mathcal{N}^{(\ell)}(f)](gh) \tag{55}$$

$$\frac{\delta[\mathcal{N}^{(\ell+1)}(f)](g)}{\delta[\mathcal{N}^{(\ell)}(f)](\tilde{g})} = \frac{1}{\sqrt{n_\ell S_\kappa}}\kappa^{(\ell+1)}(g^{-1}\tilde{g})\,. \tag{56}$$

Therefore, (54) becomes

$$\Theta^{(\ell+1)}_{g,g'}(f,f') = \frac{1}{S_\kappa}\int_{S_\kappa}\mathrm{d}h\;\mathbb{E}\left[[\mathcal{N}^{(\ell)}(f)](gh)[\mathcal{N}^{(\ell)}(f')](g'h)\right]$$

$$+\frac{1}{S_\kappa}\int_G \mathrm{d}\tilde{g}\,\mathrm{d}\tilde{g}'\;\Theta^{(\ell)}_{\tilde{g},\tilde{g}'}(f,f')\,\mathbb{E}\left[\kappa^{(\ell+1)}(g^{-1}\tilde{g})\kappa^{(\ell+1)}(g'^{-1}\tilde{g}')\right] \tag{57}$$

$$= K^{(\ell+1)}_{g,g'}(f,f') + \frac{1}{S_\kappa}\int_{S_\kappa}\mathrm{d}h\;\Theta^{(\ell)}_{gh,g'h}(f,f')\,, \tag{58}$$

where we have dropped the channel-prefactors as usual. The last line is just the NTK recursion to be proven. □

Theorem 2 (Kernel recursions for the lifting layer). *The layer-wise recursive relations for the NNGP and NTK of the lifting layer (6) are given by*

$$K^{(\ell+1)}_{g,g'}(f,f') = \frac{1}{|S_\kappa|}\int_{S_\kappa}\mathrm{d}x\;K^{(\ell)}_{\rho(g)x,\rho(g')x}(f,f')\,, \tag{18}$$

$$\Theta^{(\ell+1)}_{g,g'}(f,f') = \frac{1}{|S_\kappa|}\int_{S_\kappa}\mathrm{d}x\;\Theta^{(\ell)}_{\rho(g)x,\rho(g')x}(f,f') + K^{(\ell+1)}_{g,g'}(f,f')\,, \tag{19}$$

where the regular representation ρ_{reg} is defined in (5).

Proof. The NNGP of the lifting layer (6) is given by

$$
\begin{aligned}
K^{(\ell+1)}_{g,g'}(f,f') &= \mathbb{E}\left[[\mathcal{N}^{(\ell+1)}(f)](g)\,[\mathcal{N}^{(\ell+1)}(f')](g')\right] && (59)\\
&= \frac{1}{S_\kappa}\int_X \mathrm{d}x\,\mathrm{d}x'\ \mathbb{E}\left[\kappa(\rho(g^{-1})x)\kappa(\rho(g'^{-1})x')\right]\mathbb{E}\left[[\mathcal{N}^{(\ell)}(f)](x)[\mathcal{N}^{(\ell)}(f')](x')\right] && (60)\\
&= \frac{1}{S_\kappa}\int_X \mathrm{d}x\,\mathrm{d}x'\ \mathbb{E}\left[\kappa(x)\kappa(x')\right]\mathbb{E}\left[[\mathcal{N}^{(\ell)}(f)](\rho(g)x)[\mathcal{N}^{(\ell)}(f')](\rho(g')x')\right] && (61)\\
&= \frac{1}{S_\kappa}\int_{S_\kappa} \mathrm{d}x\ \mathbb{E}\left[[\mathcal{N}^{(\ell)}(f)](\rho(g)x)[\mathcal{N}^{(\ell)}(f')](\rho(g')x)\right] && (62)\\
&= \frac{1}{S_\kappa}\int_{S_\kappa} \mathrm{d}x\, K^{(\ell)}_{\rho(g)x,\rho(g')x}(f,f')\,, && (63)
\end{aligned}
$$

where we have moved the regular representation through $\mathcal{N}^{(\ell)}$ onto f by using equivariance. This proves the NNGP recursion-relation.

According to (2), the NTK recursion evaluates to

$$
\begin{aligned}
\Theta^{(\ell+1)}_{g,g'}(f,f') = &\int_{S_\kappa} \mathrm{d}x\ \mathbb{E}\left[\frac{\delta[\mathcal{N}^{(\ell+1)}(f)](g)}{\delta\kappa^{(\ell+1)}(x)}\frac{\delta[\mathcal{N}^{(\ell+1)}(f')](g')}{\delta\kappa^{(\ell+1)}(x)}\right]\\
&+\int_X \mathrm{d}\tilde{x}\,\mathrm{d}\tilde{x}'\ \mathbb{E}\left[\frac{\delta[\mathcal{N}^{(\ell+1)}(f)](g)}{\delta[\mathcal{N}^{(\ell)}(f)](\tilde{x})}\Theta^{(\ell)}_{\tilde{x},\tilde{x}'}(f,f')\frac{\delta[\mathcal{N}^{(\ell+1)}(f')](g')}{\delta[\mathcal{N}^{(\ell)}(f')](\tilde{x}')}\right]. \qquad (64)
\end{aligned}
$$

The derivatives in this expression are given by

$$
\begin{aligned}
\frac{\delta[\mathcal{N}^{(\ell+1)}(f)](g)}{\delta\kappa^{(\ell+1)}(x)} &= \frac{1}{\sqrt{n_\ell S_\kappa}}[\mathcal{N}^{(\ell)}(f)](\rho(g)x) && (65)\\
\frac{\delta[\mathcal{N}^{(\ell+1)}(f)](g)}{\delta[\mathcal{N}^{(\ell)}(f)](\tilde{x})} &= \frac{1}{\sqrt{n_\ell S_\kappa}}\kappa^{(\ell+1)}(\rho(g^{-1})\tilde{x})\,. && (66)
\end{aligned}
$$

Plugging this back into (64) yields the desired NTK recursion relation,

$$
\begin{aligned}
\Theta^{(\ell)}_{\tilde{x},\tilde{x}'}(f,f') &= \frac{1}{S_\kappa}\int_{S_\kappa}\mathrm{d}x\ \mathbb{E}\left[[\mathcal{N}^{(\ell)}(f)](\rho(g)x)[\mathcal{N}^{(\ell)}(f')](\rho(g')x)\right]\\
&\quad+\frac{1}{S_\kappa}\int_X \mathrm{d}\tilde{x}\,\mathrm{d}\tilde{x}'\ \Theta^{(\ell)}_{\tilde{x},\tilde{x}'}(f,f')\mathbb{E}\left[\kappa^{(\ell+1)}(\rho(g^{-1})\tilde{x})\kappa^{(\ell+1)}(\rho(g'^{-1})\tilde{x'})\right] && (67)\\
&= K^{(\ell+1)}_{g,g'}(f,f')+\frac{1}{S_\kappa}\int_{S_\kappa}\mathrm{d}x\ \Theta^{(\ell)}_{\rho(g)x,\rho(g')x'}(f,f')\,. && (68)
\end{aligned}
$$

□

Theorem 3 (Kernel recursions for group pooling layer). *The layer-wise recursive relations for the NNGP and NTK of the group pooling layer* (10) *are given by*

$$
\begin{aligned}
K^{(\ell+1)}(f,f') &= \frac{1}{(\mathrm{vol}(G))^2}\int_G \mathrm{d}g\int_G \mathrm{d}g'\ K^{(\ell)}_{g,g'}(f,f') && (20)\\
\Theta^{(\ell+1)}(f,f') &= \frac{1}{(\mathrm{vol}(G))^2}\int_G \mathrm{d}g\int_G \mathrm{d}g'\ \Theta^{(\ell)}_{g,g'}(f,f')\,. && (21)
\end{aligned}
$$

Proof. Since we integrate over the entire domain of the input feature maps $\mathcal{N}^{(\ell)}(f): G \to \mathbb{R}^{n_\ell}$ in the pooling layer (10), are the output features $\mathcal{N}^{(\ell)}(f) \in \mathbb{R}^{n_{\ell+1}}$ a channel-vector. Therefore, the NNGP of the group pooling layer is given by

$$K^{(\ell+1)}(f,f') = \mathbb{E}\left[\mathcal{N}^{(\ell+1)}(f)\,\mathcal{N}^{(\ell+1)}(f')\right] \tag{69}$$

$$= \frac{1}{(\mathrm{vol}(G))^2}\int_G \mathrm{d}g \int_G \mathrm{d}g'\ \mathbb{E}\left[[\mathcal{N}^{(\ell)}(f)](g)\,[\mathcal{N}^{(\ell)}(f)](g')\right] \tag{70}$$

$$= \frac{1}{(\mathrm{vol}(G))^2}\int_G \mathrm{d}g \int_G \mathrm{d}g'\ K^{(\ell)}_{g,g'}(f,f')\,. \tag{71}$$

The NTK recursion (2) is in this case

$$\Theta^{(\ell+1)}(f,f') = \int_G \mathrm{d}g \int_G \mathrm{d}g'\ \mathbb{E}\left[\frac{\delta\mathcal{N}^{(\ell+1)}(f)}{\delta[\mathcal{N}^{(\ell)}(f)](g)}\Theta^{(\ell)}_{g,g'}(f,f')\frac{\delta\mathcal{N}^{(\ell+1)}(f')}{\delta[\mathcal{N}^{(\ell)}(f')](g')}\right] \tag{72}$$

$$= \frac{1}{(\mathrm{vol}(G))^2}\int_G \mathrm{d}g \int_G \mathrm{d}g'\,\Theta^{(\ell)}_{g,g'}(f,f')\,, \tag{73}$$

which is the NTK-relation to be shown. □

C Equivariant kernels for roto-translations in the plane

In this appendix, we provide explicit expressions for the kernel recursions of lifting-, group convolutional- and group pooling layers for the special case of roto-translations in the plane, i.e. for the symmetry group $G = C_n \ltimes \mathbb{R}^2$. In this case, the general expressions in Theorems 1, 2 and 3 simplify and can be written in terms of the $\mathcal{A}$ operator (32) which can be computed efficiently in terms of ordinary 2d convolutions. However, before discussing the kernel recursions, we will first establish a simplifying notation for the GCNN layer-definitions.

C.1 GCNNs for $G = C_n \ltimes \mathbb{R}^2$

Due to the semidirect-product structure of G, any element $g \in G$ can be written uniquely as a product of a translation and a rotation, $g = tr$ with $t \in \mathbb{R}^2$ and $r \in C_n$[1]. We can therefore write a feature map $\mathcal{N}^{(\ell)}$ on G as a stack of n feature maps on $\mathbb{R}^2$,

$$\mathcal{N}^{(\ell)}(g = tr) = \mathcal{N}^{(\ell)}_r(t)\,. \tag{74}$$

Using this representation, the lifting- and group convolutional layers can be written in terms of ordinary two-dimensional convolutions as [28]

$$[\mathcal{N}^{(1)}(f)]_r(t) = \frac{1}{\sqrt{n_{\mathrm{in}}|S_\kappa|}}\int_{\mathbb{R}^2}\mathrm{d}x\ \kappa\big(\rho(r^{-1})(x-t)\big)f(x) \tag{75}$$

$$[\mathcal{N}^{(\ell+1)}(f)]_r(t) = \frac{1}{\sqrt{n_\ell|S_\kappa|}}\sum_{r'\in C_n}\int_{\mathbb{R}^2}\mathrm{d}t'\ \kappa_{r^{-1}r'}\big(\rho(r^{-1})(t'-t)\big)[\mathcal{N}^{(\ell)}(f)]_{r'}(t')\,, \quad \ell \geq 1\,, \tag{76}$$

where ρ is the fundamental representation of SO(2) on $\mathbb{R}^2$, given by two-dimensional rotation matrices.

Finally, for invariant problems like classification, the group pooling layer (10) is central to making the network invariant. For $C_n \ltimes \mathbb{R}^2$, it is given by

$$[\mathcal{N}^{(\ell+1)}(f)] = \frac{1}{\sqrt{n|\,\mathrm{supp}(\mathcal{N}^{(\ell)}(f))|}}\sum_{r\in C_n}\int \mathrm{d}x\ [\mathcal{N}^{(\ell)}(f)]_r(x)\,. \tag{77}$$

[1] In an abuse of notation, we will denote both the abstract translation group element and its representation as a vector in $\mathbb{R}^2$ by the same symbol.

C.2 Kernel recursions for $G = C_n \ltimes \mathbb{R}^2$

In analogy to the notation introduced in the previous section for feature maps, we write for the NNGP and NTK on $C_n \ltimes \mathbb{R}^2$

$$K_{g=tr,g'=t'r'}(f,f') = [K_{rr}(f,f')](t,t')\,, \qquad \Theta_{g=tr,g'=t'r'}(f,f') = [\Theta_{rr'}(f,f')](t,t') \tag{78}$$

to emphasize the dependency on the two translations $t, t' \in \mathbb{R}^2$. Furthermore, we repeat here the definition of the operator (32) for convenience

$$[\mathcal{A}_{S_\kappa}(K)](t,t') = \frac{1}{|S_\kappa|}\int_{S_\kappa} \mathrm{d}\tilde{t}\; K(t+\tilde{t}, t'+\tilde{t})\,, \tag{79}$$

Given these definitions, the recursive relations from Theorem 1 for group convolutions can be computed efficiently using the following

Lemma 9 (Kernel recursions of group convolutional layers for roto-translations). *In the case $G = C_n \ltimes \mathbb{R}^2$, the layer-wise recursive relations for the NNGP and NTK of the group convolutional layer* (76) *are given by*

$$[K^{(\ell+1)}_{rr'}(f,f')](t,t') = \sum_{\tilde{r}\in C_n} [\mathcal{A}_{\rho(r)S_\kappa}(\tilde{K}^{(\ell)}_{r\tilde{r},r'\tilde{r}}(f,f'))](t,\rho(rr'^{-1})t') \tag{80}$$

$$[\Theta^{(\ell+1)}_{rr'}(f,f')](t,t') = [K^{(\ell+1)}_{r\,r'}(f,f')](t,t') + \sum_{\tilde{r}\in C_n} [\mathcal{A}_{\rho(r)S_\kappa}(\tilde{\Theta}^{(\ell)}_{r\tilde{r},r'\tilde{r}}(f,f'))](t,\rho(rr'^{-1})t')\,, \tag{81}$$

where

$$[\tilde{K}^{(\ell)}_{r\,r'}(f,f')](t,t') = [K^{(\ell)}_{r\,r'}(f,f')](t,\rho(r'r^{-1})t') \tag{82}$$

$$\tilde{\Theta}^{(\ell)}_{r\,r'}(f,f') = [\Theta^{(\ell)}_{r\,r'}(f,f')](t,\rho(r'r^{-1})t')\,. \tag{83}$$

Proof. In order to compute the NNGP recursion relation in the notation (78), we first need to compute the unique decomposition of a general group multiplication gh, $g, h \in G$ into a rotation and a translation. This is possible since G is a semidirect product group. Starting from $g = t_g r_g$ and $h = t_h r_h$ with $t_g, t_h \in \mathbb{R}^2$ and $r_g, r_h \in C_n$, we have

$$gh = t_g r_g t_h r_h = t_g r_g t_h r_g^{-1} r_g r_h\,. \tag{84}$$

Since $\mathbb{R}^2$ is a normal subgroup of G (a further property implied by the semidirect product), $r_g t_h r_g^{-1} \in \mathbb{R}^2$. Therefore,

$$t_{gh} = t_g r_g t_h r_g^{-1} \in \mathbb{R}^2 \qquad \text{and} \qquad r_{gh} = r_g r_h \in C_n\,. \tag{85}$$

Since the action of t_{gh} on a vector $x \in \mathbb{R}^2$ is given by

$$\rho(t_{gh})x = x + \rho(r_g^{-1})t_h + t_g\,, \tag{86}$$

we obtain for the NNGP recursion from Theorem 1,

$$[K^{(\ell+1)}_{rr'}(f,f')](t,t') = \frac{1}{|S_\kappa|}\sum_{\tilde{r}\in C_4}\int_{S_\kappa} \mathrm{d}\tilde{t}\; [K^{(\ell)}_{r\tilde{r},r'\tilde{r}}(f,f')](\rho(r^{-1})\tilde{t}+t, \rho(r'^{-1})\tilde{t}+t')\,. \tag{87}$$

This we will now write in terms of the $\mathcal{A}$-operator (79). However, since the $\mathcal{A}$-operator shifts both slots of the argument kernel by y, whereas the first argument in (87) is shifted by $\rho(r^{-1})\tilde{t}$, while the second argument is shifted by $\rho(r'^{-1})\tilde{t}$, we cannot write (87) directly in terms of $\mathcal{A}(K)$, but need to compute $\mathcal{A}$ at a transformed argument instead and then transform back. To this end, first consider

$$
\begin{aligned}
[\mathcal{A}_{S_\kappa}(\tilde{K}^{(\ell)}_{rr'}(f,f'))](t,t') &= \frac{1}{|S_\kappa|}\int_{S_\kappa}\mathrm{d}\tilde{t}\ [\tilde{K}^{(\ell)}_{rr'}(f,f')](t+\tilde{t},t'+\tilde{t}) && (88)\\
&= \frac{1}{|S_\kappa|}\int_{S_\kappa}\mathrm{d}\tilde{t}\ [K^{(\ell)}_{rr'}(f,f')](t+\tilde{t},\rho(r'r^{-1})(t'+\tilde{t}))\,. && (89)
\end{aligned}
$$

Therefore, we obtain for the RHS of the NNGP recursion

$$
\begin{aligned}
&\sum_{\tilde{r}\in C_n}[\mathcal{A}_{\rho(r)S_\kappa}(\tilde{K}^{(\ell)}_{r\tilde{r},\,r'\tilde{r}}(f,f'))]\big(t,\rho(rr'^{-1})t'\big)\\
&= \frac{1}{|S_\kappa|}\sum_{\tilde{r}\in C_n}\int_{\rho(r)S_\kappa}\mathrm{d}\tilde{t}\ [K^{(\ell)}_{r\tilde{r},r'\tilde{r}}(f,f')]\Big(t+\tilde{t},\rho(r'r^{-1})\big(\rho(rr'^{-1})t'+\tilde{t}\big)\Big) && (90)\\
&= \frac{1}{|S_\kappa|}\sum_{\tilde{r}\in C_n}\int_{\rho(r)S_\kappa}\mathrm{d}\tilde{t}\ [K^{(\ell)}_{r\tilde{r},r'\tilde{r}}(f,f')]\big(t+\tilde{t},t'+\rho(r'r^{-1})\tilde{t}\big) && (91)\\
&= \frac{1}{|S_\kappa|}\sum_{\tilde{r}\in C_n}\int_{S_\kappa}\mathrm{d}\tilde{t}\ [K^{(\ell)}_{r\tilde{r},r'\tilde{r}}(f,f')]\big(t+\rho(r)\tilde{t},t'+\rho(r')\tilde{t}\big) && (92)
\end{aligned}
$$

The last line is just (87), proving the NNGP recursion relation.

For the NTK, we start from the NTK-recursion in Theorem 1. The structure of the integral appearing in that recursion is the same as the one of the integral in the NNGP recursion. Therefore, the NTK recursion is given by

$$
\begin{aligned}
[\Theta^{(\ell+1)}_{rr'}(f,f')](t,t') &= [K^{(\ell+1)}_{rr'}(f,f')](t,t')\\
&\quad + \frac{1}{S_\kappa}\sum_{\tilde{r}\in C_4}\int_{S_\kappa}\mathrm{d}\tilde{t}\ [\Theta^{(\ell)}_{r\tilde{r},\,r'\tilde{r}}(f,f')](\rho(r^{-1})\tilde{t}+t,\rho(r'^{-1})\tilde{t}+t')\,. && (93)
\end{aligned}
$$

The integral can be written in terms of the $\mathcal{A}$-operator following the same steps as for the NNGP above. □

Therefore, by first computing the kernels $\tilde{K}^{(\ell)}$ and $\tilde{\Theta}^{(\ell)}$ and then applying the $\mathcal{A}$-operator, it is possible to efficiently compute the kernel-recursions in this case. Similarly, the recursive kernel-relations for the lifting layer can also be written efficiently in terms of the $\mathcal{A}$-operator, as detailed in

Lemma 10 (Kernel recursions of lifting layers for roto-translations). *The layer-wise recursive relations for the NNGP and NTK of the group convolutional layer (75) are given by*

$$
\begin{aligned}
[K^{(\ell+1)}_{rr'}(f,f')](t,t') &= \big[\mathcal{A}_{\rho(r)S_\kappa}(\tilde{K}^{(\ell)}_{r\,r'}(f,f'))\big](t,\rho(rr'^{-1})t') && (94)\\
[\Theta^{(\ell+1)}_{rr'}(f,f')](t,t') &= [K^{(\ell+1)}_{rr'}(f,f')](t,t') + \big[\mathcal{A}_{\rho(r)S_\kappa}(\tilde{\Theta}^{(\ell)}_{r\,r'}(f,f'))\big](t,\rho(rr'^{-1})t')\,, && (95)
\end{aligned}
$$

where

$$
\begin{aligned}
[\tilde{K}^{(\ell)}_{rr'}(f,f')](t,t') &= [K^{(\ell)}(f,f')](t,\rho(r'r^{-1})t') && (96)\\
[\tilde{\Theta}^{(\ell)}_{rr'}(f,f')](t,t') &= [\Theta^{(\ell)}(f,f')](t,\rho(r'r^{-1})t')\,. && (97)
\end{aligned}
$$

Proof. According to Theorem 2, the NNGP recursion is given by

$$
[K^{(\ell+1)}_{rr'}(f,f')](t,t') = \frac{1}{S_\kappa}\int_{S_\kappa}\mathrm{d}x\ K^{(\ell)}\big(\rho(r)x+t,\rho(r')x+t'\big)\,. \tag{98}
$$

Comparing this expression to (87) shows that the sum over $\tilde{r}$ as well as the r, r'-indices on $K^{(\ell)}$ are absent in (98) but otherwise the two expressions agree. Therefore, we can use the same argument as above to rewrite (98) in terms of the $\mathcal{A}$-operator and only need to drop the r, r'-indices in the definition of $\tilde{K}^{(\ell)}$ as well as pick the $\tilde{r} = e$ contribution in the sum. Similarly, we can show the NTK recursion relation starting form the NTK-recursion in Theorem 2. □

Finally, in the group pooling layer, the kernels are trivialized over their r- and t-indices, resulting in a kernel without spatial indices:

Lemma 11 (Kernel recursions of group pooling layers for roto-translations). *The layer-wise recursive relations for the NNGP and NTK of the group convolutional layer* (77) *are given by*

$$K^{(\ell+1)}(f, f') = \frac{1}{n|\operatorname{supp}(\mathcal{N}^{(\ell)}(f))|} \sum_{r,r' \in C_n} \int \mathrm{d}t\, \mathrm{d}t' \; [K^{(\ell)}_{rr'}(f, f')](t, t') \tag{99}$$

$$\Theta^{(\ell+1)}(f, f') = \frac{1}{n|\operatorname{supp}(\mathcal{N}^{(\ell)}(f))|} \sum_{r,r' \in C_n} \int \mathrm{d}t\, \mathrm{d}t' \; [\Theta^{(\ell)}_{rr'}(f, f')](t, t') \,. \tag{100}$$

Proof. The integral over two copies of the group in Theorem 3 factorize for $G = C_n \ltimes \mathbb{R}^2$ into integrals over the translations in $\mathbb{R}^2$ and sums over the discrete rotations in C_n. This immediately implies the recursions in the statement of the lemma. □

The expressions given in the lemmata in this section can be straightforwardly implemented and therefore allow for explicit calculations of the NTK and NNGP of realistically-sized GCNNs.

D Equivariant NTK in the Fourier domain for 3d rotations

D.1 Group convolutions in the Fourier domain for $G = \mathrm{SO}(3)$

For compact groups, it is possible to define a Fourier transformation. The group convolution (4) then becomes a point-wise product in the Fourier domain. For the case of $G = \mathrm{SO}(3)$, the Fourier transformation is given in terms of Wigner matrices $\mathcal{D}^l_{mn}$,

$$f(R) = \sum_{l=0}^{\infty} \frac{2l+1}{8\pi^2} \sum_{m,n=-l}^{l} \hat{f}^l_{mn} \overline{\mathcal{D}^l_{mn}(R)} \tag{101}$$

$$\hat{f}^l_{mn} = \int_{\mathrm{SO}(3)} \mathrm{d}R \; f(R) \mathcal{D}^l_{mn}(R) \,, \tag{102}$$

where $R \in \mathrm{SO}(3)$ is a rotation matrix. Note that the presented convention corresponds to the one in the `s2fft` package [61].

The rotations act naturally on the sphere S^2 on which the Fourier transform is given in terms of spherical harmonics Y^l_m,

$$f(x) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} \hat{f}^l_m Y^l_m(x) \tag{103}$$

$$\hat{f}^l_m = \int_{S^2} \mathrm{d}x \; f(x) \overline{Y^l_m(x)} \,. \tag{104}$$

These Fourier transformations are e.g. used in *spherical CNNs* [59] and *steerable convolutional networks* [68], which define equivariant group convolution layers with respect to SO(3) and act on input features defined on the sphere S^2. The change to the Fourier space is motivated by the fact that group convolutions reduce to simple multiplications of the corresponding Fourier components. For SO(3), the group convolutions (4) for filter support $S_\kappa \subseteq \mathrm{SO}(3)$ are defined as

$$[\mathcal{N}^{(\ell+1)}(f)](R) = \frac{1}{\sqrt{n_\ell |S_\kappa|}} \int_{S_\kappa} \mathrm{d}S \; \kappa\big(R^{-1}S\big)[\mathcal{N}^{(\ell)}(f)](S) \,. \tag{105}$$

Using

$$\mathcal{D}^l_{mn}(R^{-1}) = \overline{\mathcal{D}^l_{nm}(R)} \,, \tag{106}$$

$$\overline{\mathcal{D}^l_{mn}(R)} = (-1)^{m-n}\mathcal{D}^l_{-m,-n}(R) \,, \tag{107}$$

$$\int_{\mathrm{SO}(3)} \mathrm{d}R \, \overline{\mathcal{D}^l_{mn}(R)}\mathcal{D}^{l'}_{m'n'}(R) = \frac{8\pi^2}{2l+1}\delta_{ll'}\delta_{nn'}\delta_{mm'} \,, \tag{108}$$

the Fourier components (102) of the layer in (105) can be written compactly as

$$[\widehat{\mathcal{N}^{(\ell+1)}(f)}]^l_{mn} = \frac{1}{\sqrt{n_\ell |S_\kappa|}} \sum_{p=-l}^{l} [\widehat{\mathcal{N}^{(\ell)}(f)}]^l_{mp}\overline{\hat{\kappa}^l_{np}} \,. \tag{109}$$

Note that we have assumed a real-valued kernel κ.

Similarly, the lifting layer (6) for features on S^2 is

$$[\mathcal{N}^{(1)}(f)](R) = \frac{1}{\sqrt{n_{\mathrm{in}} |S_\kappa|}} \int_{S^2} \mathrm{d}x \; \kappa\big(R^{-1}x\big) f(x) \,, \tag{110}$$

which, in terms of the Fourier coefficients (104) becomes

$$[\widehat{\mathcal{N}^{(1)}(f)}]^l_{mn} = \frac{1}{\sqrt{n_{\mathrm{in}} |S_\kappa|}} \frac{8\pi^2}{2l+1} \hat{f}^l_m \overline{\hat{\kappa}^l_n} \,. \tag{111}$$

Again, we have assumed a real-valued kernel κ and used the relations

$$Y^l_m(Rx) = \sum_{n=-l}^{l} \overline{\mathcal{D}^l_{mn}(R)} Y^l_n(x) \,, \tag{112}$$

$$\overline{Y^l_m(x)} = (-1)^m Y^l_{-m}(x) \,, \tag{113}$$

$$\int_{S^2} \mathrm{d}x \, \overline{Y^l_m(x)} Y^{l'}_{m'}(x) = \delta_{ll'}\delta_{mm'} \,. \tag{114}$$

D.2 Kernel recursions for $G = \mathrm{SO}(3)$

As we have seen in Section D.1 SO(3) group convolutions are frequently computed in the Fourier domain. In this section, we show how also the kernel recursions from Theorems 1 and 2 for group-convolution layers and lifting layers can be computed in the Fourier domain corresponding to the spherical convolutions presented in Section D.1. In the following we will assume filters κ with global support $S_\kappa = \mathrm{SO}(3)$ or $S_\kappa = S^2$, respectively. The reason is that equations (108) and (114) otherwise have to be replaced by expressions including the Wigner's $3j$ symbols. Due to the current lack of an efficient JAX-based implementation providing their computation, we decided to restrict ourselves to the more efficient case of global filters.

In terms of the Fourier coefficients defined in (33), the recursive Kernel-relations for the spherical convolution layer (105) are specified in the following

Lemma 12 (Kernel recursions of SO(3) group-convolutions in the Fourier domain). *The layer-wise recursive relations for the NNGP and NTK of the group convolutional layer* (109) *for $G = \mathrm{SO}(3)$ and global filters are given by*

$$[\widehat{K^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} = \frac{1}{2l+1}\delta_{ll'}\delta_{n,-n'}\sum_{p=-l}^{l}(-1)^{n-p}[\widehat{K^{\ell}(f,f')}]^{l,l'}_{mp,m'(-p)} \tag{115}$$

$$[\widehat{\Theta^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} = [\widehat{K^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} + \frac{1}{2l+1}\delta_{ll'}\delta_{n,-n'}\sum_{p=-l}^{l}(-1)^{n-p}[\widehat{\Theta^{\ell}(f,f')}]^{l,l'}_{mp,m'(-p)}\,. \tag{116}$$

Proof. We identify elements of SO(3) with 3×3 rotation matrices $R, S, \ldots$. Then, the recursive relations for the group convolution layer from Theorem 1 are

$$K^{(\ell+1)}_{R,R'}(f,f') = \frac{1}{8\pi^2}\int_{\mathrm{SO}(3)}\mathrm{d}S\; K^{(\ell)}_{RS,R'S}(f,f') \tag{117}$$

$$\Theta^{(\ell+1)}_{R,R'}(f,f') = K^{(\ell+1)}_{R,R'}(f,f') + \frac{1}{8\pi^2}\int_{\mathrm{SO}(3)}\mathrm{d}S\;\Theta^{(\ell)}_{RS,R'S}(f,f')\,. \tag{118}$$

The Fourier coefficients of the NNGP are given by a double Fourier integral of the form (33), so the recursion (117) becomes

$$[\widehat{K^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} = \frac{1}{8\pi^2}\iiint_{[\mathrm{SO}(3)]^3}\mathrm{d}S\,\mathrm{d}R\,\mathrm{d}R'\;K^{(\ell)}_{RS,R'S}(f,f')\mathcal{D}^l_{mn}(R)\mathcal{D}^{l'}_{m'n'}(R')\,. \tag{119}$$

Plugging in the Fourier expansion of the kernel $K^{(\ell)}_{RS,R'S}(f,f')$ yields

$$\begin{aligned}[\widehat{K^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} = \frac{1}{8\pi^2}\iiint_{[\mathrm{SO}(3)]^3}\mathrm{d}S\,\mathrm{d}R\,\mathrm{d}R'\Bigg(&\sum_{p,p'=0}^{\infty}\frac{2p+1}{8\pi^2}\frac{2p'+1}{8\pi^2}\\ &\times\sum_{q,r=-p}^{p}\sum_{q',r'=-p'}^{p'}[\widehat{K^{(\ell)}(f,f')}]^{p,p'}_{qr,q'r'}\overline{\mathcal{D}^p_{qr}(RS)\mathcal{D}^{p'}_{q'r'}(R'S)}\Bigg)\\ &\times\mathcal{D}^l_{mn}(R)\mathcal{D}^{l'}_{m'n'}(R')\,.\end{aligned} \tag{120}$$

Using (108), (107) and

$$\mathcal{D}^l_{mn}(RS) = \sum_{p=-l}^{l}D^l_{mp}(R)\mathcal{D}^l_{pn}(S)\,, \tag{121}$$

we can simplify the expression to

$$\begin{aligned}[\widehat{K^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} &= \frac{1}{8\pi^2}\sum_{p,p'=0}^{\infty}\sum_{q,r=-p}^{p}\sum_{q',r'=-p'}^{p'}(-1)^{r-u}\frac{8\pi^2}{2l+1}\\ &\qquad\times\delta_{pl}\delta_{mq}\delta_{nu}\delta_{l'p'}\delta_{m'q'}\delta_{n'u'}\delta_{pp'}\delta_{u,-u'}\delta_{r,-r'}[\widehat{K^{(\ell)}(f,f')}]^{p,p'}_{qr,q'r'} \qquad (122)\\ &= \frac{1}{2l+1}\delta_{ll'}\delta_{n,-n'}\sum_{r=-l}^{l}(-1)^{r-n}[\widehat{K^{(\ell)}(f,f')}]^{l,l'}_{mr,m'(-r)}\,. \qquad (123)\end{aligned}$$

Renaming the summation index $r \to p$ yields the desired result. The computation for the NTK is analogous. □

Similarly, for the lifting layer (110) for features on the sphere, the kernel recursions can be expressed in terms of the Fourier coefficients (104) according to the following

Lemma 13 (Kernel recursions of spherical lifting layer in the Fourier domain). *The layer-wise recursive relations for the NNGP and NTK of the lifting layer* (111) *for features on* S^2 *to features on* SO(3) *with global filters are given by*

$$[\widehat{K^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} = \frac{1}{4\pi}\left(\frac{8\pi^2}{2l+1}\right)^2 (-1)^n \delta_{ll'}\delta_{n,-n'}[\widehat{K^{(\ell)}(f,f')}]^{l,l}_{m,m'} \tag{124}$$

$$[\widehat{\Theta^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} = [\widehat{K^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} + \frac{1}{4\pi}\left(\frac{8\pi^2}{2l+1}\right)^2 (-1)^n \delta_{ll'}\delta_{n,-n'}[\widehat{\Theta^{(\ell)}(f,f')}]^{l,l}_{m,m'}\,. \tag{125}$$

Proof. Starting from the recursive relations for the lifting layer in Theorem 2, the recursions in real space for $G = \mathrm{SO}(3)$ are

$$K^{(\ell+1)}_{R,R'}(f,f') = \frac{1}{4\pi}\int_{S^2}\mathrm{d}x\, K^{(\ell)}_{Rx,\,R'x}(f,f') \tag{126}$$

$$\Theta^{(\ell+1)}_{R,R'}(f,f') = K^{(\ell+1)}_{g,g'}(f,f') + \frac{1}{4\pi}\int_{S^2}\mathrm{d}x\ \Theta^{(\ell)}_{Rx,\,R'x}(f,f')\,. \tag{127}$$

Expressing the Fourier coefficients of the NNGP according to (33) gives

$$[\widehat{K^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} = \frac{1}{4\pi}\int_{S^2}\mathrm{d}x \iint_{[\mathrm{SO}(3)]^2}\mathrm{d}R\,\mathrm{d}R'\ K^{(\ell)}_{Rx,R'x}(f,f')\mathcal{D}^l_{mn}(R)\mathcal{D}^{l'}_{m'n'}(R')\,. \tag{128}$$

We can now plug in the Fourier expansion of the kernel on S^2

$$K^{(\ell)}_{x,x'}(f,f') = \sum_{l,l'=0}^{\infty}\sum_{m=-l}^{l}\sum_{m'=-l'}^{l'}[\widehat{K^{(\ell)}(f,f')}]^{l,l'}_{m,m'}Y^l_m(x)Y^{l'}_{m'}(x')\,, \tag{129}$$

to obtain

$$\begin{aligned}[\widehat{K^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} = \frac{1}{4\pi}\int_{S^2}\mathrm{d}x \iint_{[\mathrm{SO}(3)]^2}\mathrm{d}R\,\mathrm{d}R' \Bigg(&\sum_{p,p'=0}^{\infty}\sum_{q=-p}^{p}\sum_{q'=-p'}^{p'}[\widehat{K^{(\ell)}(f,f')}]^{p,p'}_{q,q'} \\ &\times Y^p_q(Rx)Y^{p'}_{q'}(R'x')\Bigg)\mathcal{D}^l_{mn}(R)\mathcal{D}^{l'}_{m'n'}(R')\,. \end{aligned}\tag{130}$$

Using (108), (114), (112) and (113) one can rewrite and simplify the expression as

$$\begin{aligned}[\widehat{K^{(\ell+1)}(f,f')}]^{l,l'}_{mn,m'n'} &= \frac{1}{4\pi}\sum_{p,p'=0}^{\infty}\sum_{q,r=-p}^{p}\sum_{q',r'=-p'}^{p'}\frac{8\pi^2}{2l+1}\frac{8\pi^2}{2l'+1}(-1)^{r'} \\ &\qquad\times \delta_{lp}\delta_{mq}\delta_{nr}\delta_{l'p'}\delta_{m'q'}\delta_{n'r'}\delta_{pp'}\delta_{r,-r'}[\widehat{K^{(\ell)}(f,f')}]^{p,p'}_{q,q'} \end{aligned}\tag{131}$$

$$= \frac{1}{4\pi}\left(\frac{8\pi^2}{2l+1}\right)^2 (-1)^n\delta_{l,l'}\delta_{n,-n'}[\widehat{K^{(\ell)}(f,f')}]^{l,l}_{m,m'}\,, \tag{132}$$

which is the claimed result. □

E Proofs: Data augmentation versus group convolutions at infinite width

In this section, we provide proofs for the theorems given in Section 5 in the main text.

Theorem 5. *Let μ_t^{aug} and μ_t be the mean predictions after t training steps of infinite ensembles of two neural network architectures $\mathcal{N}^{\text{aug}}$ and $\mathcal{N}$. Let $\mathcal{N}^{\text{aug}}$ be trained on the fully G-augmented training data of $\mathcal{N}$ and assume that the NTKs of the two architectures are related by*

$$\Theta(f, f') = \frac{1}{\operatorname{vol}(G)} \int_G \mathrm{d}g\ \Theta^{\text{aug}}(f, \rho_{\text{reg}}(g) f')\,. \tag{34}$$

Then, μ_t^{aug} and μ_t converge in the infinite width limit to the same function for all t for quadratic losses, up to quadratic corrections in the learning rate.

Proof. For a neural network $\mathcal{N}$, we can expand the change $\Delta\mathcal{N}$ in output due to one training step of gradient descent in the learning rate η

$$\Delta\mathcal{N}_{t+1}(f) = \mathcal{N}_{t+1}(f) - \mathcal{N}_t(f) = (\theta_{t+1} - \theta_t)^\top \frac{\partial \mathcal{N}_t(f)}{\partial\theta} + O(\eta^2) \tag{133}$$

$$= -\frac{\eta}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} \underbrace{\left(\frac{\partial \mathcal{N}_t(f)}{\partial\theta}\right)^\top \frac{\partial \mathcal{N}_t(f)}{\partial\theta}}_{\Theta_t(f,f_i)} \mathcal{L}'(\mathcal{N}_t(f_i), y_i) + O(\eta^2)\,, \tag{134}$$

where Θ_t is the empirical NTK at training step t, y_i are the training labels and $\mathcal{L}'$ is the derivative of the per-sample loss with respect to the output of the network. Taking the mean and the infinite width limit yields

$$\Delta\mu_{t+1}(f) = -\frac{\eta}{n_{\text{train}}} \sum_{i=1}^{n_{\text{train}}} \Theta(f, f_i)\, \mathcal{L}'(\mu_t(f_i), y_i)\,, \tag{135}$$

since we have assumed that $\mathcal{L}'$ is linear in its first argument.

The network $\mathcal{N}^{\text{aug}}$ on the other hand is trained using full data augmentation over G, so we can decompose the sum over training samples into a sum over the training samples in (135) and a sum over G. Note that since we assume full data augmentation and a finite training set, we restrict to G being finite in this section. We obtain

$$\Delta\mu_{t+1}^{\text{aug}}(f) = -\frac{\eta}{n_{\text{train}}|G|} \sum_{g\in G} \sum_{i=1}^{n_{\text{train}}} \Theta^{\text{aug}}(f, \rho_{\text{reg}}(g) f_i) \mathcal{L}'(\mu_t^{\text{aug}}(\rho_{\text{reg}}(g) f_i), y_i)\,. \tag{136}$$

As mentioned in the main text, we will prove the statement inductively over training steps t. At $t = 0$, the mean output of all neural networks is zero in the infinite width limit [36, 37]. For the induction step, assume that $\mu_t^{\text{aug}} = \mu_t$. Then, $\mu_{t+1}^{\text{aug}} = \mu_{t+1}$ if $\Delta\mu_{t+1}^{\text{aug}} = \Delta\mu_{t+1}$. Since the ensemble mean of networks trained with data augmentation is exactly equivariant [48, 62], we have $\mu_t^{\text{aug}}(\rho_{\text{reg}}(g) f_i) = \mu_t^{\text{aug}}(f_i) = \mu_t(f_i)$ by the induction assumption. Therefore,

$$\Delta\mu_{t+1}^{\text{aug}}(f) = -\frac{\eta}{n_{\text{train}}|G|} \sum_{g\in G} \sum_{i=1}^{n_{\text{train}}} \Theta^{\text{aug}}(f, \rho_{\text{reg}}(g) f_i) \mathcal{L}'(\mu_t(f_i), y_i)\,. \tag{137}$$

Using assumption (34) concludes the proof,

$$\Delta\mu_{t+1}^{\text{aug}}(f) = -\eta \sum_{i=1}^{n_{\text{train}}} \Theta(f, f_i) \mathcal{L}'(\mu_t(f_i), y_i) = \Delta\mu_{t+1}(f)\,. \tag{138}$$

□

Theorem 6. *Let $\mathcal{N}^{\mathrm{FC}}$ be an MLP acting on feature maps with output in $\mathbb{R}$ and architecture*

$$\mathcal{N}^{\mathrm{FC}} = \mathrm{FC}^{(L)} \circ \sigma \circ \cdots \circ \mathrm{FC}^{(3)} \circ \sigma \circ \mathrm{FC}^{(1)}\,, \tag{35}$$

where FC denotes a dense MLP layer and σ a point-wise nonlinearity. Let $\mathcal{N}^{\mathrm{GC}}$ be a G-invariant GCNN with architecture

$$\mathcal{N}^{\mathrm{GC}} = \mathrm{GPool} \circ \mathrm{GConv}(S_\kappa^L) \circ \sigma \circ \mathrm{GConv}(S_\kappa^{L-2}) \circ \sigma \circ \cdots \circ \mathrm{GConv}(S_\kappa^3) \circ \sigma \circ \mathrm{Lifting}(S_\kappa^1)\,, \tag{36}$$

where S_κ^ℓ are the supports of the convolutional filters with $S_\kappa^1 = X$, the domain of the input feature maps, and the other S_κ^ℓ are invariant under G. Then, the G-averages of the kernels of the MLP are given by the kernels of the GCNN,

$$K^{\mathrm{GC}}(f, f') = \frac{1}{\mathrm{vol}(G)} \int \mathrm{d}g\ K^{\mathrm{FC}}(f, \rho_{\mathrm{reg}}(g) f') \tag{37}$$

$$\Theta^{\mathrm{GC}}(f, f') = \frac{1}{\mathrm{vol}(G)} \int \mathrm{d}g\ \Theta^{\mathrm{FC}}(f, \rho_{\mathrm{reg}}(g) f')\,. \tag{38}$$

Proof. In order to proof the kernel equalities, we will construct the kernels for the fully connected architecture (35) and the group convolutional architecture (36) by explicitly iterating the recursion relations.

The iteration starts with the input kernels which for the fully-connected network are

$$K^{\mathrm{FC}(0)}(f, f') = \frac{1}{\mathrm{vol}(X)} \int \mathrm{d}x\ f(x) f'(x)\,, \qquad \Theta^{\mathrm{FC}(0)}(f, f') = 0\,, \tag{139}$$

since the different points in the domain X of the input function take the role of different channels when the image tensor is flattened. The first layer of the FC-network is a fully-connected layer. These update the kernels according to [18]

$$K^{\mathrm{FC}(\ell+1)}(f, f') = K^{\mathrm{FC}(\ell)}(f, f') \tag{140}$$

$$\Theta^{\mathrm{FC}(\ell+1)}(f, f') = K^{\mathrm{FC}(\ell+1)}(f, f') + \Theta^{\mathrm{FC}(\ell)}(f, f')\,. \tag{141}$$

In order to write kernel transformation like this more compactly, we will collect all relevant kernels at layer ℓ into an $\mathbb{R}^4$ vector $\Xi^{\mathrm{FC}(\ell)}(f, f')$ according to

$$\Xi^{\mathrm{FC}(\ell)}(f, f') = \begin{pmatrix} K^{\mathrm{FC}(\ell)}(f, f) \\ K^{\mathrm{FC}(\ell)}(f, f') \\ K^{\mathrm{FC}(\ell)}(f', f') \\ \Theta^{\mathrm{FC}(\ell)}(f, f') \end{pmatrix}, \tag{142}$$

where the components $K^{\mathrm{FC}(\ell)}(f, f)$ and $K^{\mathrm{FC}(\ell)}(f', f')$ are needed for the nonlinear layers below. In the Ξ^{FC}-notation, (140), (141) can be summarized by a function $\mathcal{G} : \mathbb{R}^4 \to \mathbb{R}^4$ mapping $\Xi^{\mathrm{FC}(\ell)}(f, f') \mapsto \Xi^{\mathrm{FC}(\ell+1)}(f, f')$, defined by

$$\mathcal{G}\begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ \Theta \end{pmatrix} = \begin{pmatrix} k_1 \\ k_2 \\ k_3 \\ k_2 + \Theta \end{pmatrix}. \tag{143}$$

Therefore, the kernels of the first fully-connected layer take the form

$$\Xi^{\mathrm{FC}(1)}(f, f') = \mathcal{G}(\Xi^{\mathrm{FC}(0)}(f, f')) = \frac{1}{\mathrm{vol}(X)} \int \mathrm{d}x \begin{pmatrix} f(x) f(x) \\ f(x) f'(x) \\ f'(x) f'(x) \\ f(x) f'(x) \end{pmatrix}. \tag{144}$$

In the architecture (35), fully connected layers are alternated with nonlinearities, which act according to [18]

$$\Lambda^{\mathrm{FC}(\ell)}(f,f') = \begin{pmatrix} K^{\mathrm{FC}(\ell)}(f,f) & K^{\mathrm{FC}(\ell)}(f,f') \\ K^{\mathrm{FC}(\ell)}(f',f) & K^{\mathrm{FC}(\ell)}(f',f') \end{pmatrix} \tag{145}$$

$$K^{\mathrm{FC}(\ell+1)}(f,f') = \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\,\Lambda^{\mathrm{FC}(\ell)}(f,f'))}[\sigma(u)\sigma(v)] \tag{146}$$

$$\dot{K}^{\mathrm{FC}(\ell+1)}(f,f') = \mathbb{E}_{(u,v)\sim\mathcal{N}(0,\,\Lambda^{\mathrm{FC}(\ell)}(f,f'))}[\sigma'(u)\sigma'(v)] \tag{147}$$

$$\Theta^{\mathrm{FC}(\ell+1)}(f,f') = \dot{K}^{\mathrm{FC}(\ell+1)}(f,f')\Theta^{\mathrm{FC}(\ell)}(f,f')\,. \tag{148}$$

on the kernels. We will denote the corresponding action on the Ξ^{FC}-vectors by a function $\mathcal{F}_\sigma : \mathbb{R}^4 \to \mathbb{R}^4$. Therefore, a fully-connected layer followed by a nonlinearity can be written as

$$\Xi^{\mathrm{FC}(\ell+2)}(f,f') = \mathcal{F}_\sigma(\mathcal{G}(\Xi^{\mathrm{FC}(\ell)}(f,f')))\,. \tag{149}$$

Hence, in this notation, the kernels of the entire FC network are given by

$$\Xi^{\mathrm{FC}}(f,f') = \mathcal{G}\Big(\mathcal{F}_\sigma\Big(\cdots\mathcal{G}\Big(\mathcal{F}_\sigma\Big(\mathcal{G}(\Xi^{\mathrm{FC}(0)}(f,f'))\Big)\Big)\cdots\Big)\Big)\,. \tag{150}$$

Next, we compute the kernels of the GCNN. The input kernels in this case are

$$K^{\mathrm{GC}(0)}_{x,x'}(f,f') = f(x)f'(x)\,, \qquad \Theta^{\mathrm{GC}(0)}_{x,x'}(f,f') = 0\,. \tag{151}$$

According to (36), the first layer of the network is a lifting layer whose recursion relation was given in Theorem 2. Again, we define an $\mathbb{R}^4$-vector to collect all kernel components necessary for computing the kernels of the network,

$$\Xi^{\mathrm{GC}(\ell)}_{g,g'}(f,f') = \begin{pmatrix} K^{\mathrm{GC}(\ell)}_{g,g}(f,f) \\ K^{\mathrm{GC}(\ell)}_{g,g'}(f,f') \\ K^{\mathrm{GC}(\ell)}_{g',g'}(f',f') \\ \Theta^{\mathrm{GC}(\ell)}_{g,g'}(f,f') \end{pmatrix}. \tag{152}$$

In terms of Ξ^{GC} the kernels of the lifting layer are given by (note that the filter of the lifting layer has global support by assumption)

$$\Xi^{\mathrm{GC}(1)}_{g,g'}(f,f') = \mathcal{G}\left(\frac{1}{\mathrm{vol}(X)}\int_X \mathrm{d}x \begin{pmatrix} K^{\mathrm{GC}(0)}_{\rho(g)x,\rho(g)x}(f,f) \\ K^{\mathrm{GC}(0)}_{\rho(g)x,\rho(g')x}(f,f') \\ K^{\mathrm{GC}(0)}_{\rho(g')x,\rho(g')x}(f',f') \\ \Theta^{\mathrm{GC}(0)}_{\rho(g)x,\rho(g')x}(f,f') \end{pmatrix}\right) = \frac{1}{\mathrm{vol}(X)}\int_X \mathrm{d}x \begin{pmatrix} f(\rho(g)x)f(\rho(g)x) \\ f(\rho(g)x)f'(\rho(g')x) \\ f'(\rho(g')x)f'(\rho(g')x) \\ f(\rho(g)x)f'(\rho(g')x) \end{pmatrix}. \tag{153}$$

For later convenience, we note here that

$$\Xi^{\mathrm{GC}(1)}_{h,g^{-1}h}(f,f') = \frac{1}{\mathrm{vol}(X)}\int_X \mathrm{d}x \begin{pmatrix} f(\rho(h)x)f(\rho(h)x) \\ f(\rho(h)x)f'(\rho(g^{-1}h)x) \\ f'(\rho(g^{-1}h)x)f'(\rho(g^{-1}h)x) \\ f(\rho(h)x)f'(\rho(g^{-1}h)x) \end{pmatrix} \tag{154}$$

$$= \frac{1}{\mathrm{vol}(X)}\int_X \mathrm{d}x \begin{pmatrix} f(x)f(x) \\ f(x)f'(\rho(g^{-1})x) \\ f'(\rho(g^{-1})x)f'(\rho(g^{-1})x) \\ f(x)f'(\rho(g^{-1})x) \end{pmatrix} \tag{155}$$

$$= \Xi^{\mathrm{FC}(1)}(f,\rho_{\mathrm{reg}}(g)f')\,, \tag{156}$$

where we shifted the integration variable in the second step and used (144).

After the lifting layer, we act with a point-wise nonlinearty whose recursion relations are given in Corollary 4. Since this transformation is independent for the different g, g'-components, we can write it using the same function $\mathcal{F}_\sigma$ introduced above as

$$\Xi^{\mathrm{GC}(\ell+1)}_{g,g'}(f,f') = \mathcal{F}_\sigma(\Xi^{\mathrm{GC}(\ell)}_{g,g'}(f,f'))\,. \tag{157}$$

A GCNN layer transforms the NNGP and NTK according to Theorem 1. We can write this in terms of $\Xi^{\mathrm{GC}(\ell)}_{g,g'}$ as

$$\Xi^{\mathrm{GC}(\ell+1)}_{g,g'}(f,f') = \frac{1}{|S^\ell_\kappa|}\int_{S_\kappa} \mathrm{d}h_\ell\ \mathcal{G}(\Xi^{\mathrm{GC}(\ell)}_{gh_\ell,g'h_\ell}(f,f'))\,, \tag{158}$$

with $\mathcal{G}$ as introduced in (143). The final pooling layer acts according to Theorem 3, which we can write as

$$\Xi^{\mathrm{GC}(\ell)}_{g,g'}(f,f') = \frac{1}{(\mathrm{vol}(G))^2}\int_G \mathrm{d}g \int_G \mathrm{d}g'\, \Xi^{\mathrm{GC}(\ell)}_{g,g'}(f,f')\,. \tag{159}$$

With the expressions (157), (158) and (159), we can write the kernels of the entire network as

$$\begin{aligned}\Xi^{\mathrm{GC}}(f,f') = \frac{1}{(\mathrm{vol}(G))^2}\int_G \mathrm{d}g \int_G \mathrm{d}g'\ \frac{1}{|S^L_\kappa|}\int_{S^L_\kappa}\mathrm{d}h_L\ \mathcal{G}\Bigg(\mathcal{F}_\sigma\Bigg(\frac{1}{|S^{L-2}_\kappa|}\int_{S^{L-2}_\kappa}\mathrm{d}h_{L-2}\ \mathcal{G}\Bigg(\mathcal{F}_\sigma\Bigg(\cdots \\ \cdots\frac{1}{|S^3_\kappa|}\int_{S^3_\kappa}\mathrm{d}h_3\ \mathcal{G}\left(\mathcal{F}_\sigma(\Xi^{\mathrm{GC}(1)}_{gh_Lh_{L-2}\cdots h_5h_3,g'h_Lh_{L-2}\cdots h_5h_3}(f,f'))\right)\cdots\Bigg)\Bigg)\Bigg)\Bigg)\,.\end{aligned} \tag{160}$$

In order to simplify this expression, we shift h_3 and absorb $gh_Lh_{L-2}\cdots h_5$ into it. This will not change the integration domain of h_3 since S^3_κ is by assumption invariant under G. Then, the integrals over $h_L, h_{L-2}, \ldots, h_5$ become trivial and cancel against their $1/|S^\ell_\kappa|$-prefactors. We are left with

$$\begin{aligned}\Xi^{\mathrm{GC}}(f,f') = \frac{1}{(\mathrm{vol}(G))^2}\int_G \mathrm{d}g \int_G \mathrm{d}g'\ \mathcal{G}\Bigg(\mathcal{F}_\sigma\Bigg(\mathcal{G}\Bigg(\mathcal{F}_\sigma\Bigg(\cdots \\ \cdots\ \frac{1}{|S^3_\kappa|}\int_{S^3_\kappa}\mathrm{d}h_3\ \mathcal{G}\left(\mathcal{F}_\sigma(\Xi^{\mathrm{GC}(1)}_{h_3,g'g^{-1}h_3}(f,f'))\right)\cdots\Bigg)\Bigg)\Bigg)\Bigg)\,.\end{aligned} \tag{161}$$

Finally, we trivialize the g'-integral by shifting g^{-1} to absorb g'. Thus, we obtain

$$\Xi^{\mathrm{GC}}(f,f') = \frac{1}{\mathrm{vol}(G)}\int_G \mathrm{d}g\ \mathcal{G}\left(\mathcal{F}_\sigma\left(\mathcal{G}\left(\mathcal{F}_\sigma\left(\cdots\frac{1}{|S^3_\kappa|}\int_{S^3_\kappa}\mathrm{d}h_3\ \mathcal{G}\left(\mathcal{F}_\sigma(\Xi^{\mathrm{GC}(1)}_{h_3,g^{-1}h_3}(f,f'))\right)\cdots\right)\right)\right)\right) \tag{162}$$

$$= \frac{1}{\mathrm{vol}(G)}\int_G \mathrm{d}g\ \mathcal{G}\left(\mathcal{F}_\sigma\left(\mathcal{G}\left(\mathcal{F}_\sigma\left(\cdots\mathcal{G}\left(\mathcal{F}_\sigma(\Xi^{\mathrm{FC}(1)}(f,\rho_{\mathrm{reg}}(g)f'))\right)\cdots\right)\right)\right)\right) \tag{163}$$

$$= \frac{1}{\mathrm{vol}(G)}\int_G \mathrm{d}g\ \Xi^{\mathrm{FC}}(f,\rho_{\mathrm{reg}}(g)f')\,, \tag{164}$$

where we used (156), trivializing the integral over h_3, and then identified Ξ^{FC} from (150). The statement follows by taking the second and fourth components of (164). □

Theorem 7. *Let $\mathcal{N}^{K\ltimes N}$ be the $K \ltimes N$-invariant GCNN with architecture (36) and K-invariant filter supports S^ℓ_κ which for the GConv-layers decompose as $S^\ell_\kappa = K^\ell_\kappa \times N^\ell_\kappa$, $K^\ell_\kappa \subseteq K$, $N^\ell_\kappa \subseteq N$. Let $\mathcal{N}^N$ be the N-invariant GCNN with architecture (36) and filter supports $N^L_\kappa, \ldots, N^3_\kappa$ and S^1_κ. Then, the NNGPs and NTKs of these networks are related by*

$$K^{K\ltimes N}(f,f') = \frac{1}{\mathrm{vol}(K)}\int_K \mathrm{d}k\ K^N(f,\rho_{\mathrm{reg}}(k)f') \tag{39}$$

$$\Theta^{K\ltimes N}(f,f') = \frac{1}{\mathrm{vol}(K)}\int_K \mathrm{d}k\ \Theta^N(f,\rho_{\mathrm{reg}}(k)f')\,. \tag{40}$$

Proof. In this proof, we will use the same notation as in the proof for Theorem 6 above and use results from there as well. We start by considering the kernels $\Xi^{K\ltimes N}$ of $\mathcal{N}^{K\ltimes N}$ by specializing (160) to the case $G = K \ltimes N$. Due to the semidirect product structure of G, there is a unique decomposition $g = kn$ for each $g \in G$ into $k \in K$ and $n \in N$. Since by assumption the filter supports S_κ^ℓ on G also factorize over K and N, we can split all G-integrations in (160) over N and K and obtain

$$\Xi^{K\ltimes N}(f,f') = \frac{1}{(\mathrm{vol}(K))^2}\frac{1}{(\mathrm{vol}(N))^2}\int_K \mathrm{d}k \int_N \mathrm{d}n \int_K \mathrm{d}k' \int_N \mathrm{d}n' \, \frac{1}{|K_\kappa^L||N_\kappa^L|}\int_{K_\kappa^L} \mathrm{d}j_L \int_{N_\kappa^L} \mathrm{d}m_L \; \mathcal{G}\Bigg(\mathcal{F}_\sigma \cdots$$
$$\cdots \frac{1}{|K_\kappa^3||N_\kappa^3|}\int_{K_\kappa^3} \mathrm{d}j_3 \int_{N_\kappa^3} \mathrm{d}m_3 \; \mathcal{G}\left(\mathcal{F}_\sigma(\Xi^{K\ltimes N(1)}_{knj_Lm_L\cdots j_3m_3,k'n'j_Lm_L\cdots j_3m_3}(f,f'))\right)\cdots\Bigg)\,. \tag{165}$$

In order to trivialize the integrals over K, as was done with the integrals over G in (161), we need to rewrite the first group index of $\Xi^{K\ltimes N(1)}$ such that all j_ℓ appear next to each other. To this end, we introduce several unit elements

$$knj_Lm_L\cdots j_7m_7j_5m_5j_3m_3 = knj_Lm_L\cdots j_7m_7j_5j_3\,\underbrace{j_3^{-1}m_5j_3}_{\in N}\,m_3 \tag{166}$$
$$= knj_Lm_L\cdots j_7j_5j_3\,\underbrace{(j_5j_3)^{-1}m_7j_5j_3}_{\in N}\,\underbrace{j_3^{-1}m_5j_3}_{\in N}\,m_3 \tag{167}$$
$$\vdots$$
$$= kj_L\cdots j_3\,\underbrace{(j_L\cdots j_3)^{-1}nj_L\cdots j_3}_{\in N}\,\underbrace{(j_{L-2}\cdots j_3)^{-1}m_L\cdots}_{\in N}\,\underbrace{j_3^{-1}m_5j_3}_{\in N}\,m_3\,. \tag{168}$$

We perform the same rewriting also on the second group index of $\Xi^{K\ltimes N(1)}$. Since N is a normal subgroup of G, $knk^{-1} \in N$ for all $k \in K$, $n \in N$ and the Haar measure on N is invariant under shifts of the form $n \longrightarrow knk^{-1}$. Furthermore, the integration domains N_κ^ℓ are by assumption invariant under this transformation. Hence, we shift n, n' and the m_ℓ by

$$n \longrightarrow j_L\cdots j_3n(j_L\cdots j_3)^{-1} \tag{169}$$
$$n' \longrightarrow j_L\cdots j_3n'(j_L\cdots j_3)^{-1} \tag{170}$$
$$m_\ell \longrightarrow j_{\ell-2}\cdots j_3m_\ell(j_{\ell-2}\cdots j_3)^{-1} \qquad \ell > 3\,. \tag{171}$$

With this (165) becomes

$$\Xi^{K\ltimes N}(f,f') = \frac{1}{(\mathrm{vol}(K))^2}\frac{1}{(\mathrm{vol}(N))^2}\int_K \mathrm{d}k \int_N \mathrm{d}n \int_K \mathrm{d}k' \int_N \mathrm{d}n' \, \frac{1}{|K_\kappa^L||N_\kappa^L|}\int_{K_\kappa^L} \mathrm{d}j_L \int_{N_\kappa^L} \mathrm{d}m_L \; \mathcal{G}\Bigg(\mathcal{F}_\sigma \cdots$$
$$\cdots \frac{1}{|K_\kappa^3||N_\kappa^3|}\int_{K_\kappa^3} \mathrm{d}j_3 \int_{N_\kappa^3} \mathrm{d}m_3 \; \mathcal{G}\left(\mathcal{F}_\sigma(\Xi^{K\ltimes N(1)}_{kj_L\cdots j_3nm_L\cdots m_3,k'j_L\cdots j_3n'm_L\cdots m_3}(f,f'))\right)\cdots\Bigg) \tag{172}$$

$$= \frac{1}{(\mathrm{vol}(K))^2}\frac{1}{(\mathrm{vol}(N))^2}\int_K \mathrm{d}k \int_N \mathrm{d}n \int_K \mathrm{d}k' \int_N \mathrm{d}n' \, \frac{1}{|N_\kappa^L|}\int_{N_\kappa^L} \mathrm{d}m_L \; \mathcal{G}\Bigg(\mathcal{F}_\sigma \cdots$$
$$\cdots \frac{1}{|K_\kappa^3||N_\kappa^3|}\int_{K_\kappa^3} \mathrm{d}j_3 \int_{N_\kappa^3} \mathrm{d}m_3 \; \mathcal{G}\left(\mathcal{F}_\sigma(\Xi^{K\ltimes N(1)}_{j_3nm_L\cdots m_3,k'k^{-1}j_3n'm_L\cdots m_3}(f,f'))\right)\cdots\Bigg) \tag{173}$$

$$= \frac{1}{\mathrm{vol}(K)}\frac{1}{(\mathrm{vol}(N))^2}\int_K \mathrm{d}k \int_N \mathrm{d}n \int_N \mathrm{d}n' \, \frac{1}{|N_\kappa^L|}\int_{N_\kappa^L} \mathrm{d}m_L \; \mathcal{G}\Bigg(\mathcal{F}_\sigma \cdots$$
$$\cdots \frac{1}{|K_\kappa^3||N_\kappa^3|}\int_{K_\kappa^3} \mathrm{d}j_3 \int_{N_\kappa^3} \mathrm{d}m_3 \; \mathcal{G}\left(\mathcal{F}_\sigma(\Xi^{K\ltimes N(1)}_{j_3nm_L\cdots m_3,k^{-1}j_3n'm_L\cdots m_3}(f,f'))\right)\cdots\Bigg)\,. \tag{174}$$

Here, we shifted $j_3 \longrightarrow (k j_L \cdots j_5)$ in the first step, trivializing the integrals over $j_L, \dots, j_5$ which then cancel against their $1/|K_\kappa^\ell|$-prefactors. In the second step, we first trivialized the integral over k' by shifting $k' \longrightarrow k'k$ and then canceled it against its $1/\mathrm{vol}(K)$-prefactor.

Next, we perform another manipulation on the group indices of $\Xi^{K\ltimes N(1)}$ by first inserting suitable unit elements,

$$j_3 n m_L \cdots m_3 = \underbrace{j_3 n j_3^{-1}}_{\in N} \underbrace{j_3 m_L j_3^{-1}}_{\in N} \underbrace{j_3 m_{L-2} j_3^{-1}}_{\in N} \cdots \underbrace{j_3 m_3 j_3^{-1}}_{\in N} j_3 \,, \tag{175}$$

and similarly for the second group index of $\Xi^{K\ltimes N(1)}$. After shifting

$$n \longrightarrow j_3^{-1} n j_3 \,, \qquad n' \longrightarrow j_3^{-1} n' j_3 \,, \qquad m_\ell \longrightarrow j_3^{-1} m_\ell j_3 \quad \ell \geq 3 \,, \tag{176}$$

in (174), we obtain

$$\begin{aligned}\Xi^{K\ltimes N}(f, f') = \frac{1}{\mathrm{vol}(K)} \frac{1}{(\mathrm{vol}(N))^2} \int_K \mathrm{d}k \int_N \mathrm{d}n \int_N \mathrm{d}n' \, \frac{1}{|N_\kappa^L|} \int_{N_\kappa^L} \mathrm{d}m_L \; \mathcal{G}\Bigg(\mathcal{F}_\sigma \cdots \\ \cdots \frac{1}{|K_\kappa^3||N_\kappa^3|} \int_{K_\kappa^3} \mathrm{d}j_3 \int_{N_\kappa^3} \mathrm{d}m_3 \; \mathcal{G}\left(\mathcal{F}_\sigma(\Xi^{K\ltimes N(1)}_{n m_L \cdots m_3 j_3, k^{-1} n' m_L \cdots m_3 j_3}(f, f'))\right) \cdots \Bigg) \,.\end{aligned} \tag{177}$$

As in the proof for Theorem 6, we will now write $\Xi^{K\ltimes N(1)}$ in terms of $\Xi^{N(1)}$. Using the shorthand $\tilde{m} = m_L \cdots m_3$ and the analogous steps to (156), we find

$$\Xi^{K\ltimes N(1)}_{n\tilde{m}j_3, k^{-1}n'\tilde{m}j_3}(f, f') = \frac{1}{|S_\kappa^1|} \int_{S_\kappa^1} \mathrm{d}x \begin{pmatrix} f(\rho(n\tilde{m}j_3)x) f(\rho(n\tilde{m}j_3)x) \\ f(\rho(n\tilde{m}j_3)x) f'(\rho(k^{-1}n'\tilde{m}j_3)x) \\ f'(\rho(k^{-1}n'\tilde{m}j_3)x) f'(\rho(k^{-1}n'\tilde{m}j_3)x) \\ f(\rho(n\tilde{m}j_3)x) f'(\rho(k^{-1}n'\tilde{m}j_3)x) \end{pmatrix} \tag{178}$$

$$= \frac{1}{|S_\kappa^1|} \int_{S_\kappa^1} \mathrm{d}x \begin{pmatrix} f(\rho(n\tilde{m})x) f(\rho(n\tilde{m})x) \\ f(\rho(n\tilde{m})x) f'(\rho(k^{-1}n'\tilde{m})x) \\ f'(\rho(k^{-1}n'\tilde{m})x) f'(\rho(k^{-1}n'\tilde{m})x) \\ f(\rho(n\tilde{m})x) f'(\rho(k^{-1}n'\tilde{m})x) \end{pmatrix} \tag{179}$$

$$= \Xi^{N(1)}_{n\tilde{m}, n'\tilde{m}}(f, \rho_{\mathrm{reg}}(k) f') \,, \tag{180}$$

where for the second equality, we have shifted $x \longrightarrow \rho(j_3^{-1})x$, which leaves S_κ^1 invariant by assumption. Plugging (180) into (177) trivializes the j_3-integral which cancels against its $1/|K_\kappa^3|$-prefactor, yielding

$$\begin{aligned}\Xi^{K\ltimes N}(f, f') = \frac{1}{\mathrm{vol}(K)} \int_K \mathrm{d}k \; \frac{1}{(\mathrm{vol}(N))^2} \int_N \mathrm{d}n \int_N \mathrm{d}n' \, \frac{1}{|N_\kappa^L|} \int_{N_\kappa^L} \mathrm{d}m_L \; \mathcal{G}\Bigg(\mathcal{F}_\sigma \cdots \\ \cdots \frac{1}{|N_\kappa^3|} \int_{N_\kappa^3} \mathrm{d}m_3 \; \mathcal{G}\left(\mathcal{F}_\sigma(\Xi^{N(1)}_{n m_L \cdots m_3, n' m_L \cdots m_3}(f, \rho_{\mathrm{reg}}(k) f'))\right) \cdots \Bigg)\end{aligned} \tag{181}$$

$$= \frac{1}{\mathrm{vol}(K)} \int_K \mathrm{d}k \; \Xi^N(f, \rho_{\mathrm{reg}}(k) f') \,, \tag{182}$$

where we have identified Ξ^N by comparing to (160). The statement of the theorem follows by considering the second and fourth components of (182). □

F Further experimental results

In this appendix, we provide further details and results of the numerical experiments presented in Section 6.

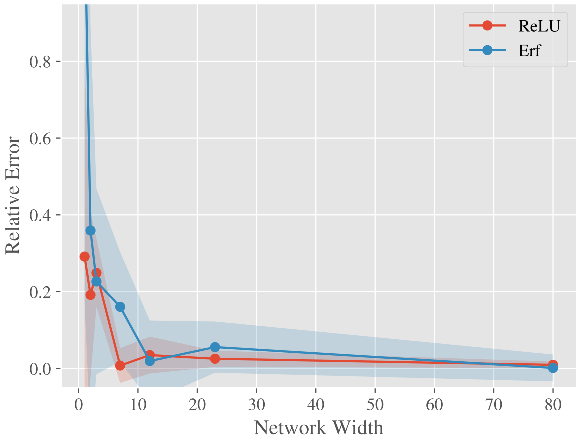

Figure 5: **Convergence of the Monte-Carlo estimates of the NNGP to their infinite-width limits for $G = C_4 \ltimes \mathbb{R}^2$.** Plotted is the relative error averaged over the components of a 3×3 Gram matrix for networks with a ReLU or an error function nonlinearity. The bands correspond to ± one standard deviation of the estimator.

F.1 Kernel convergence

Figure 5 shows the convergence of Monte-Carlo estimates of the NNGP to the analytical infinite-width expression derived using the theorems in Section 4.1.

F.2 Medical image experiments

In the infinite-width limit, the NTK becomes deterministic and time-independent under the gradient flow dynamics. In the case of MSE loss, the differential equation describing the mean output of a network at time t becomes a linear ODE, thus allowing for an analytic expression at arbitrary time. In the limit of infinite training time, the mean is given by (43) [18]. This relation is effectively a kernel method that can be used to generate prediction of the infinitely wide network.

The task consists of classifying histological images [64] containing nine classes of tissues, two of which are cancerous. The original images have a resolution of 224×224 pixels each and have been down-scaled to a resolution of 32×32 pixels to reduce the kernel evaluation time. Note that the size of the final kernel matrix, that needs to be inverted, is independent of the resolution because we use a group pooling or `SumPool` layer, respectively. Since the analytic solution in (43) only applies for MSE loss, we constructed target vectors $\mathcal{Y} = \{y_0, \dots, y_N\}$ from classes c according to $\boldsymbol{e}_c - \frac{1}{9}\mathbf{1}$ as is standard in the NTK literature [69].

The CNN and GCNN architectures that were used are shown in Table 1. Note that the infinite-width limit refers to the number of channels, which is why we only need to specify the kernel sizes. The same training and test data was used for both models with a test data size of 1000 images. Both architectures have been implemented in the `neural-tangents` package [55].

F.3 Molecular energy regression

We used the same kernel method resulting from the infinite-width and infinite-time limit as explained in Section F.2. Both the grid on S^2 as well as on SO(3) are equiangular Driscoll & Healy grids [70] with

Table 1: Architectures used for the medical image classification described in Section 6. For convolutional, group-convolutional and lifting layers, the argument is the kernel size (all kernels are squared). Both pooling layers are global. The number of output neurons is finite and has to correspond to the 9 classes.

CNN	GCNN
`Conv(3)`	`Lifting(3)`
`ReLU`	`ReLU`
`Conv(3)`	`GConv(3)`
`ReLU`	`ReLU`
`Conv(3)`	`GConv(3)`
`ReLU`	`ReLU`
`Conv(3)`	`GConv(3)`
`ReLU`	`ReLU`
`Conv(3)`	`GConv(3)`
`ReLU`	`ReLU`
`SumPool`	`GPool`
`Dense`	`Dense`
`ReLU`	`ReLU`
`Dense(9)`	`Dense(9)`

Table 2: Architectures used for the molecular energy regression described in Section 6. 29 identical networks (encaptured by curly braces) process the inputs associated to each atom. Their outputs are then summed together. For group-convolutional and lifting layers, the output bandlimit L is stated. The pooling layer is global and the single output neuron represents the predicted energy of the network.

MLP	GCNN
29 per-atom networks	*29 per-atom networks*
$\left\{\begin{array}{c}\texttt{Dense}\\ \texttt{ReLU}\\ \texttt{Dense}\end{array}\right\} \dots \left\{\begin{array}{c}\texttt{Dense}\\ \texttt{ReLU}\\ \texttt{Dense}\end{array}\right\}$	$\left\{\begin{array}{c}\texttt{Lifting(3)}\\ \texttt{Erf}\\ \texttt{GConv(3)}\end{array}\right\} \dots \left\{\begin{array}{c}\texttt{Lifting(3)}\\ \texttt{Erf}\\ \texttt{GConv(3)}\end{array}\right\}$
combined to molecule network	*combined to molecule network*
`FanInSum`	`FanInSum`
`Dense(1)`	`Dense(1)`

resolution $2L \times (2L-1)$[2] on S^2 and $(2L-1) \times 2L \times (2L-1)$ on SO(3) (parametrized in Euler angles). L is the corresponding bandlimit defining the cutoff in the Fourier domain, i.e. only Fourier coefficients $l < L$ are considered. The input signals are sampled for $L = 6$.

As the labels we have used the internal energies U_0 of the molecules at 0 K after substracting the atomic reference energies. The hyperparameter β in (44) was chosen as described in [66] according to

$$\beta = \frac{\cos(\pi/4)) - 1)^2}{\log(0.05)} \tag{183}$$

The precise architectures of the MLP based network and the SO(3)-invariant network are listed in Table 2. The MAE loss was evaluated on a test set of 100 molecules.

[2] In the original work by [70] the grid contained actually $2L \times 2L$ points, but we have adapted our grid to the convention used in the `s2fft` package.

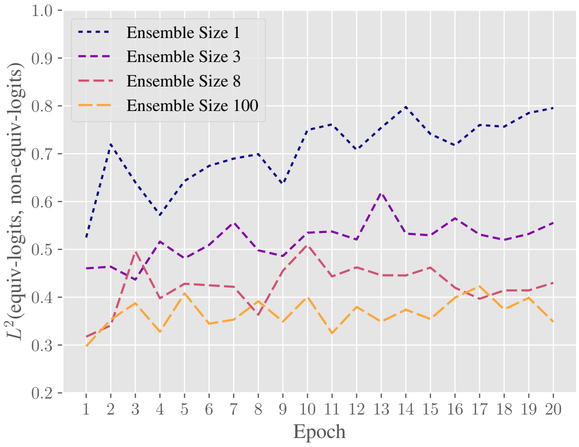

Figure 6: **Convergence of finite-width ensembles trained with data augmentation to ensembles of GCNNs on CIFAR10.** Shown is the L^2-distance between the logits of the equivariant ensemble and the non-equivariant ensemble trained with data augmentation for different ensemble sizes on out of distribution data. For larger ensembles, the distance decreases.

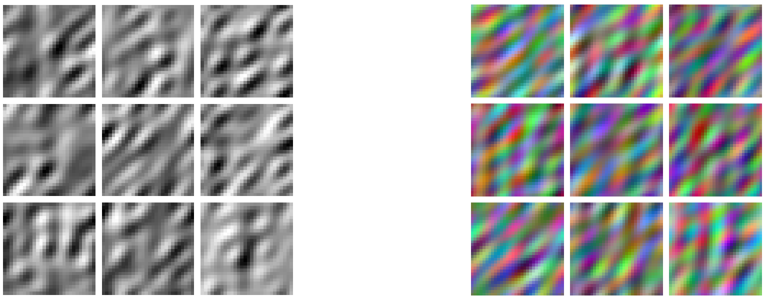

Figure 7: Examples for out of distribution data for MNIST (left) and CIFAR10 (right).

F.4 Data augmentation versus group convolutions at finite width

Figure 6 shows that large ensembles trained with data augmentation on CIFAR10 converge to GCNNs even out of distribution. Samples of the out of distribution data, whose mean and variance were normalized to 0 and 1, respectively, are provided in Figure 7. The architectures used for the ensemble members are detailed in Table 3.

Table 3: Architectures used for the ensemble members in the experiments described in Section 6. For convolutional, group-convolutional and lifting layers, the arguments are input channels, output channels and kernel size (all kernels are squared). For max-pooling layers, the arguments are kernel size and stride. For the GCNNs, the max pooling is done only over spatial dimensions, not group dimensions. The kernel sizes were selected such that the GCNNs are exactly equivariant for the respective input sizes of 28 × 28 and 32 × 32.

MNIST		CIFAR10	
CNN	GCNN	CNN	GCNN
Conv(1, 4, 3)	Lifting(1, 4, 3)	Conv(3, 4, 3)	Lifting(3, 4, 3)
ReLU	ReLU	ReLU	ReLU
MaxPool(2, 2)	SpatialMaxPool(2, 2)	MaxPool(2, 2)	SpatialMaxPool(2, 2)
Conv(4, 16, 4)	GConv(4, 16, 4)	Conv(4, 16, 4)	GConv(4, 16, 4)
ReLU	ReLU	ReLU	ReLU
MaxPool(2, 2)	SpatialMaxPool(2, 2)	MaxPool(2, 2)	SpatialMaxPool(2, 2)
Conv(16, 32, 3)	GConv(16, 32, 3)	Conv(16, 32, 3)	GConv(16, 32, 3)
ReLU	ReLU	ReLU	ReLU
Conv(32, 64, 3)	GConv(32, 64, 3)	Conv(32, 64, 4)	GConv(32, 64, 4)
ReLU	ReLU	ReLU	ReLU
Conv(64, 128, 1)	GConv(64, 128, 1)	Conv(64, 128, 1)	GConv(64, 128, 1)
ReLU	ReLU	ReLU	ReLU
Conv(128, 32, 1)	GConv(128, 32, 1)	Conv(128, 32, 1)	GConv(128, 32, 1)
ReLU	ReLU	ReLU	ReLU
Conv(32, 10, 1)	GConv(32, 10, 1)	Conv(32, 10, 1)	GConv(32, 10, 1)
	GPool		GPool