$SO(2)$-equivariance in Neural networks using tensor nonlinearity

Muthuvel Murugan
muthu@cmi.ac.in
K V Subrahmanyam
kv@cmi.ac.in

Chennai Mathematical Institute
Siruseri IT Park, Chennai, India

Abstract

Inspired by recent work of Kondor [11] and Cohen and Welling [3], we build rotation equivariant autoencoders to obtain a basis of images adapted to the group of planar rotations $SO(2)$, directly from the data. We do this in an unsupervised fashion, working in the Fourier domain of $SO(2)$. Working in the Fourier domain we build a rotation equivariant classifier to classify images. As in the recent papers of Thomas et al. [20] and Kondor et al. [12] we use tensor product nonlinearity to build our autoencoders and classifiers. We discover the basis using a small sample of inputs. As a consequence our classifier is *robust* to rotations - the classifier trained on upright images, classifies rotated versions of images, achieving state of the art. In order to deal with images under different scales simultaneously, we define the notion of a coupled-bases and show that a coupled-bases can be learned using tensor nonlinearity.

1 Introduction

Convolutional neural networks [13] have met with tremendous success on a wide range of learning problems. GPUs bring enormous computational power to such networks. But this is not the only reason for their impressive performance. CNNs learn features of images using nonlinearities such as RELU and are able to detect local patterns. Features are learned using cross-correlation with filters. Weight sharing ensures that even if the image is translated, a useful feature is still captured. So the networks performance is invariant to the group of 2-D translations. Invariance seems to be one reason why CNNs do so well, see [6].

The machine learning community has leveraged ideas and notions from group theory for a variety of tasks. Rao and Ruderman [17] used Lie groups to model visual perception. Sohl-Dickstein et al. [19] developed a Lie group model to learn visual transformations. In his thesis Kondor [10] used the representation theory of $SO(3)$ to extract nonlinear, invariant features of images wrapped around a sphere. Mimisevic [15] showed that learning relationships between images can be viewed as detecting rotations in the simultaneous eigenspaces of a collection of orthogonal matrices. Bruna and Mallat [1] introduced scattering networks which compute representations of images that are invariant to translations and that are stable under deformations. Sifre and Mallat [18] extended the work of [1] to other groups, building on earlier works of Duits and Burgeth [7]. Cohen and Welling [3] build a model to learn irreducible representations of $SO(2)$, to produce disentangled representations of images for classification.

Given the incredible success of CNNs, recent work has focused on building networks that have invariance to a larger group of symmetries. Jaderberg et al. [8] show how allowing spatial manipulation of data within CNNs results in networks which are invariant to scale, rotations, translations and warping. Cohen and Welling [4] designed Group Convolutional Networks (GCNN's) in order to learn representations of images which are invariant to the symmetries of the square and to translations. Harmonic nets were introduced by Worrall et al. [21] to learn representations of images invariant to rotations. Using sophisticated ideas from group representation theory Cohen and Welling [5] introduced Steerable CNN's. In order to deal with 3D images Cohen et al. [6] introduced Spherical CNNs, equivariant to $SO(3)$. They generalized cross-correlation of CNNs to spherical cross-correlation using ideas from non-commutative harmonic analysis. A drawback of [6] is the need to implement exact *group based convolutions* in order to achieve equivariance. Spherical CNNs work in the image space but need to go back and forth from the image space to the Fourier domain of functions on $SO(3)$. Clebsch-Gordon networks were introduced by Kondor et al. [12] and operate entirely in the Fourier domain of functions on $SO(3)$.

Recall the intuitive definition from Kondor et al. [12] of what it would mean for a network to be equivariant to a group G. Denoting the activations of neurons in layer l by f^l, mathematically, equivariance would mean that if the networks inputs are transformed by $g \in G$, f^l should transform as $T_g^l(f^l)$, for some fixed set of linear transformations $\{T_g^l\}_{g\in G}$. In Clebsch-Gordon networks such equivariance is shown to hold for $G = SO(3)$, by choosing the activations to be Fourier coefficients of functions on $SO(3)$. Kondor et al. [12] introduced a natural nonlinearity in the Fourier domain and used that to extract features of images invariant to $SO(3)$. This use of nonlinearity in the Fourier domain is relatively new and can also be seen in the recent works of Thomas et al. [20]. See also Pratt et al. [16], wherein the authors implement CNNs in the Fourier domain.

In this paper we propose a neural network model working in the Fourier domain which learns features of 2-D images invariant to the group $SO(2)$. Our model is inspired by the works of [3] and [11]. Our model first learns a basis of images adapted to the group $SO(2)$. We propose a simple autoencoder architecture for this that uses tensor nonlinearity. We learn the basis using a small sample of inputs. For images in different scales we define the notion of a coupled-bases of images adapted to rotations. We learn a coupled-bases using tensor nonlinearity. We build a classifier for images using tensor nonlinearity.

Acknowledgements

Both authors were supported by a partial grant from the Infosys Foundation. A GPU was purchased from a grant K.V received from SERB, MTR/2017/000325.

2 Activations in $SO(2)$-equivariant networks.

A rotation about the origin in the XY plane by an angle θ is given in the standard basis by the matrix $R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$. The group of planar rotations, $SO(2)$, is isomorphic to the complex circle group of roots of 1, $\mathbb{T}$, with $R(\theta)$ mapping to $e^{i\theta}$. We abuse notation and use

θ to denote the element $e^{i\theta} \in \mathbb{T}$. As a topological group $\mathbb{T}$ is isomorphic to $S^1 := \mathbb{R}/2\pi\mathbb{Z}$ and so we also use the notation S^1 instead of $\mathbb{T}$.

To describe our networks we will need some notions from harmonic analysis on S^1 and the representation theory of S^1. We collect the relevant background in Appendices A and B. We only recall two relevant definitions from these Appendices to set the stage.

Definition 1. *Let V be a complex vector space. A representation of S^1 on V is a group homomorphism $\rho : S^1 \to GL(V)$. We say S^1 acts on V. Given $\theta \in S^1, v \in V$ we denote by $\theta \cdot v$, or $\theta(v)$, the element $\rho(\theta)v$.*

Definition 2. *Let V, W be two representations of S^1. A complex linear map $\phi : V \to W$ is said to be S^1-equivariant (or an S^1-morphism) if for all $v \in V, \theta \in S^1$, $\phi(\theta \cdot v) = \theta \cdot (\phi(v))$.*

Assume we are dealing with images of size $N \times N$. Vectorize the image and regard it as a function on a complex vector space V of dimension N^2, with a natural basis indexed by the N^2 pixel positions. When an image is rotated about its centre by an angle θ, the N^2 pixels move, and extending this linearly to all vectors in V, we get a linear transformation of V[1]. This transformation can be described by an $N^2 \times N^2$ dimensional transformation matrix $T(\theta)$, whose entries are functions of the single variable θ. The complex vector space V can be shown to be a representation of S^1. Under the transformation $T(\theta)$, the image I is also transformed. The transformed image $T(\theta)(I)$ thought of as a function on V is given by $T(\theta)(I)(x) = I(T(\theta)^{-1}(x))$. So we get a representation of S^1 on V^* (which we identify with $N \times N$-images).

We apply Theorem 11(Appendix B) to the S^1-representation V^*. Using the notation from the statement of the theorem, let $\{\mathbf{b}^0_{j,k}\}_{j=1}^{j=m_k}$ be a basis of V^*_k. Each input image I can be written as $I = \sum_k \sum_{j=1}^{j=m_k} f^0_{j,k} \mathbf{b}^0_{j,k}$. Here m_k is the dimension of V^*_k and is called multiplicity of the irreducible representation of type k in V^*. It follows from Theorem 11(Appendix B) that when the image is transformed by θ, the $\{f^0_{j,k}\}$'s transform as $f^0_{j,k} \mapsto e^{-ik\theta} f^0_{j,k}$, exactly like the Fourier coefficients of the cross-correlation, see Proposition 9(Appendix A). We refer to the collection of complex numbers $\{f^0_{j,k}\}_{j=1}^{j=m_k}$ as the Fourier coefficients of the image I of type k.

2.1 $SO(2)$-NN's.

The activations of the classifier network and the autoencoder networks we build are collections of complex numbers which transform like the Fourier coefficients of functions on S^1.[2] Kondor et al. [12] define an $SO(3)$-equivariant network. We adapt their definition to suit our requirement.

Definition 3. *Let $\mathcal{N}$ be a $S+1$ layer feed-forward neural network which takes $N \times N$ images as inputs. We say $\mathcal{N}$ is an $SO(2)$-NN if the output of each layer ℓ, $\ell = 0, 1, \ldots, S$, can be expressed as a collection of complex numbers*

$$\{f^\ell_{1,-r}, f^\ell_{2,-r}, \ldots, f^\ell_{m^\ell_{-r},-r}, \ldots, \ldots, f^\ell_{1,0}, f^\ell_{2,0}, \ldots, f^\ell_{m^\ell_0,0}, \ldots, \ldots, f^\ell_{1,r} f^\ell_{2,r}, \ldots, f^\ell_{m^\ell_r,r}\}$$

with the property that when the input image is rotated by θ, the $f^\ell_{a,b}$ transform as

$$f^\ell_{a,b} \to e^{-ib\theta} f^\ell_{a,b}$$

[1] we do not worry about the fact that some pixels go out of bounds

[2] After completing this work we were pointed to Kondor et al. [12] and discovered that our model is similar to theirs, albeit much simpler, since the features learned in our model are invariant to a smaller group of transformations.

For an image I, the complex numbers observed in layer ℓ of $\mathcal{N}$ are called the Fourier coefficients of I in layer ℓ of $\mathcal{N}$.

2.2 $SO(2)$-NN's, a first attempt

In Section 2.4 we build autoencoders to compute a basis $\{\mathbf{b}^0_{j,k}\}_{j,k}$ of images under rotations. Let F^0 be the span of $\{\mathbf{b}^0_{j,k}\}_{j,k}$. For an image I, let $\{f^0_{j,k}\}_{j,k}$ be Fourier coefficients of I in the basis $\{\mathbf{b}^0_{j,k}\}_{j,k}$. It follows from the discussion in Section 2 that these complex numbers satisfy the requirements of layer $\ell = 0$ in Definition 3.

For $\ell \geq 1$, let F^ℓ be the complex vector space spanned by basis elements $\{\mathbf{b}^\ell_{j,k}\}_{j,k}$ (the indices that j,k run over determine the number and types of the Fourier coefficients occurring in the output of layer ℓ). We define an action of S^1 on the basis elements - $\theta \cdot \mathbf{b}^\ell_{j,k} = e^{-ik\theta}\mathbf{b}^\ell_{j,k}$ and extend the action to all of F^ℓ by linearity. So we have a representation of S^1 on F^ℓ. Using the same terminology as before, we call $\mathbf{b}^\ell_{j,k}$ a basis vector of type k. Let F^ℓ_k be the subspace of F^ℓ spanned by basis vectors of type k. Clearly $\theta \cdot v = e^{-ik\theta}v$ for all $v \in F^\ell_k$. Let ϕ be any linear map from F^ℓ_k to $F^{\ell+1}_k$. For every $v \in F^\ell_k$ we have $\phi(\theta \cdot v) = e^{-ik\theta}\phi(v)$. Such a linear map is given by a matrix of size $m^{\ell+1}_k \times m^\ell_k$. A block diagonal matrix $\phi^{\ell+1}_\ell$ with blocks of size $m^{\ell+1}_k \times m^\ell_k$, one for each k, also satisfies $\phi^{\ell+1}_\ell(\theta \cdot v) = \theta \cdot \phi^{\ell+1}_\ell(v)$ for all $v \in F^\ell$, so $\phi^{\ell+1}_\ell$ is an S^1-equivariant map from F^ℓ to $F^{\ell+1}$.

For $\ell = 0, \ldots, S-1$, we fix S^1-equivariant maps from F^ℓ to $F^{\ell+1}$. Let the output of layer $\ell+1$ be the coefficients of the vector obtained in layer $\ell+1$ in the basis $\{\mathbf{b}^{\ell+1}_{j,k}\}_{j,k}$. Since the Fourier coefficients at level $\ell = 0$ transform as required in the Definition 3 and since the maps between the layers are S^1-equivariant, this network is an $SO(2)$-NN.

2.2.1 Tensor nonlinearity

There is one issue with the above formulation, there is *no nonlinearity* in the above network. To fix this we start with vector spaces $G^1, \ldots, G^S$, G^ℓ having a basis $\{\mathbf{g}^\ell_{j,k}\}_{j,k}$. $\mathbf{g}^\ell_{j,k}$ is a basis vector of type k and the multiplicity of k in G^ℓ is n^ℓ_k. Note that $G^\ell \otimes G^\ell$ has a basis $\{\mathbf{g}^\ell_{j,k} \otimes \mathbf{g}^\ell_{s,m}\}_{j,k,s,m}$ and there is an action of S^1 on $G^\ell \otimes G^\ell$ with $\theta \cdot \mathbf{g}^\ell_{j,k} \otimes \mathbf{g}^\ell_{s,m} = e^{-i(k+m)\theta}\mathbf{g}^\ell_{j,k} \otimes \mathbf{g}^\ell_{s,m}$, so $\mathbf{g}^\ell_{j,k} \otimes \mathbf{g}^\ell_{s,m}$ is a basis vector of type $k+m$. If the coefficients of a vector $v \in G^\ell$ in the basis $\mathbf{g}^\ell_{j,k}$ are $h^\ell_{j,k}$ the coefficient of $\mathbf{g}^\ell_{j,k} \otimes \mathbf{g}^\ell_{s,m}$ in $v \otimes v$ is the product $h^\ell_{j,k}h^\ell_{s,m}$. Set $F^\ell = G^\ell \oplus (G^\ell \otimes G^\ell)$. Fix S^1-equivariant maps ϕ^ℓ from F^ℓ to $G^{\ell+1}$, for $\ell = 0, 1, \ldots, S-1$. Define $\phi^{\ell+1}_\ell$ to be the map sending $v \in F^\ell$ to $\phi^\ell(v) \oplus \phi^\ell(v) \otimes \phi^\ell(v) \in F^{\ell+1}$, in the basis $\{\mathbf{g}^{\ell+1}_{j,k} \otimes \mathbf{g}^{\ell+1}_{s,m}\}_{j,k,s,m} \cup \{\mathbf{g}^{\ell+1}_{j,k}\}_{j,k}$. Note that $\phi^{\ell+1}_\ell$ is a nonlinear map from F^ℓ to $F^{\ell+1}$ obtained by taking a tensor product, so we call this *tensor nonlinearity*.

Set the output of layer ℓ to be the coefficients of the vector $v^\ell \in F^\ell$ formed at layer ℓ (in the chosen basis). Set $\{f^0_{j,k}\}_{j,k}$ as in Section 2.2. The resulting network is an $SO(2)$-NN for the same reasons as before.

2.3 An $SO(2)$-NN classifier

Figure 1 shows a 2 layer $SO(2)$-NN classifier for 28×28 images which uses tensor nonlinearity. We construct the basis $W = \{\mathbf{b}^0_{j,k}\}_{j,k}$ - this is described in Section 2.4. Using this basis we obtain the Fourier coefficients $\{f^0_{j,k}\}_{j,k}$ of the input image x. Using the same notation as in

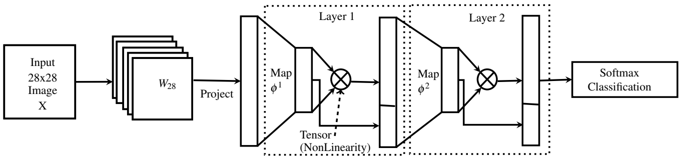

Figure 1: Classification network

Section 2.2.1, ϕ^1, ϕ^2 are two S^1-equivariant maps, ϕ^1 mapping F^0 to G^1 and ϕ^2 mapping F^1 to G^2. $\phi^1(W^Tx) \oplus (\phi^1(W^Tx) \otimes \phi^1(W^Tx)) \in F^1$ is the input to layer 2. These blocks repeat in a deeper network.

The softmax classification uses only the Fourier coefficients of type zero from the last layer, because they are features invariant to rotations. We initialize ϕ^1, ϕ^2 at random, and minimize classification error and learn ϕ^1, ϕ^2. The hyperparameters of the network are the types and multiplicities of the basis elements of G^1, G^2. In a deeper network we chose these hyperparameters for each layer.

2.4 The autoencoder

It remains to describe the architecture to learn a basis $\{\mathbf{b}^0_{j,k}\}_{j,k}$ of images under rotations. The problem of learning such a basis of images under rotations was first considered by Cohen and Welling [3]. So we call such a basis of images adapted to rotations a CW-basis. In [3] an expectation maximization algorithm was proposed for learning $\{\mathbf{b}^0_{j,k}\}_{j,k}$.

We use the same notation from Section 2.2 and Section 2.3. We start with an image x and rotate it by a random angle θ to obtain y. Let W be a matrix whose columns are the current basis $\{\mathbf{b}^0_{j,k}\}_{j,k}$ of F^0, with $k = -r, \ldots, r$ and $j = 1, \ldots, m^0_k$, m^0_k being the multiplicity of type k in F^0. The Fourier coefficients of y in the basis $\{\mathbf{b}^0_{j,k}\}_{j,k}$ is W^Ty. Defining G^1, F^1 as before, we construct a nonlinear map ϕ^1_0 from F^0 to F^1 using the S^1-equivariant map ϕ^1. The output of layer 1 is $\phi^1(W^Ty) \oplus \phi^1(W^Ty) \otimes \phi^1(W^Ty)$. ψ is an S^1-equivariant map from F^1 to F^0. We want the image of ψ to be y. In the chosen basis of F^0 we therefore expect the Fourier coefficients of $\psi(\phi^1(\hat{y}) \oplus \phi^1(\hat{y}) \otimes \phi^1(\hat{y}))$ of type k, to differ from the corresponding coefficient of x by the factor $e^{-ik\theta}$. To undo the effect of rotation, we act the output of ψ by $-\theta \in S^1$, so each Fourier coefficient of type k in $\psi(\phi^1(\hat{y}) \oplus \phi^1(\hat{y}) \otimes \phi^1(\hat{y}))$ gets multiplied by $e^{ik\theta}$. Using these Fourier coefficients and the current basis $\{\mathbf{b}^0_{j,k}\}_{j,k}$ we construct an image $\hat{x}$ and compare it with x since, if everything worked out properly, $\hat{x}$ would be very close to x.

We minimize the reconstruction error $|x - \hat{x}|^2$. Since we want W to be orthonormal, we use the regularizer $\lambda||W^TW - Id||_2$.

The hyperparameters of this network are the types and multiplicities of the basis of F^0 and the types and multiplicities in G^1. We learn W and the S^1-equivariant maps ϕ^1, ψ.

Remark 4. *Note that while we set out to construct a basis of the image space (of 28 x 28 images) we end up with a linearly independent set of vector spanning a subspace of the image space. And these basis are good enough to express the given set of images. This is similar to what happens in singular value decomposition. So, technically, we should call the basis elements discovered, a CW-subbasis of the image space. We ignore this technicality.*

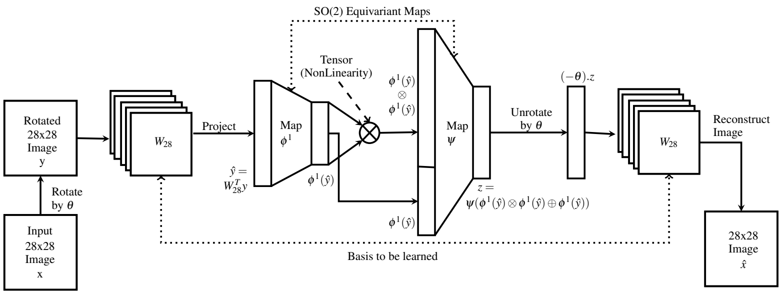

Figure 2: Autoencoder architecture (AE).

2.5 Coupling

We also find CW-bases of images in two different scales, simultaneously, with each influencing the discovery of the other. One reason to do so is to be able to use a classifier trained on large size images on smaller sized inputs, with no further training.

To formulate our requirements precisely we need a few definitions.

Definition 5. *Let V be a complex vector space. Denote by $V^{\otimes k}$ the k-fold tensor product of V with itself. The tensor algebra of V is the algebraic direct sum*

$$\mathbb{T}(V) = \oplus_{k \geq 0} V^{\otimes k}$$

Our interest is not in the algebra structure of $T(V)$, but finite dimensional S^1-invariant subspaces of $T(V)$ of the form V, $V \oplus V^{\otimes 2}$, $V \oplus V^{\otimes 2} \oplus V^{\otimes 3}$ and so on.

Now if V is a representation of S^1 then the above finite dimensional subspaces of the tensor algebra are also representations of S^1.

Definition 6. *We say S^1-representations U and $\tilde{U}$ are coupled if there exists a finite dimensional S^1-invariant subspace W (resp. V) of $\mathbb{T}(\tilde{U})$ (resp. $\mathbb{T}(U)$) and an S^1-equivariant surjective map $\phi : W \to U$ (resp. an S^1-equivariant surjective map $\phi : V \to \tilde{U}$).*

Definition 7. *Let V be the vector space of 14×14 images with an S^1-action and let W be the vector space of 28×28 images also with an S^1-action. Let X be a Cohen-Welling basis of V and Y be a Cohen-Welling basis of W. Let U denote the dual of the subspace spanned by X with its S^1-action and let $\tilde{U}$ denote the dual of the subspace spanned by Y with its S^1-action. We say X and Y are coupled if U and $\tilde{U}$ are coupled.*

Our definition suggests that in order to generate coupled CW-bases we will need to view the images at different scales simultaneously, and use tensor product nonlinearity to generate them, thereby forcing the coupling we are looking for.

Motivated by the above definition we say features obtained by projections on a coupled-bases are *coupled features*.

2.6 Coupled Autoencoder(CAE) architecture

The schematic diagram for learning a coupled-bases is given in Figure 3. We use the notation from Section 2.4. In this setup we learn CW-bases W_{14}, W_{28} for 14×14 images and 28×28 images in tandem. We feed both, an image X and a downsampled image x to the network. The network on top takes x and produces $\hat{X}$, a 28x28 image. The bottom network takes X and

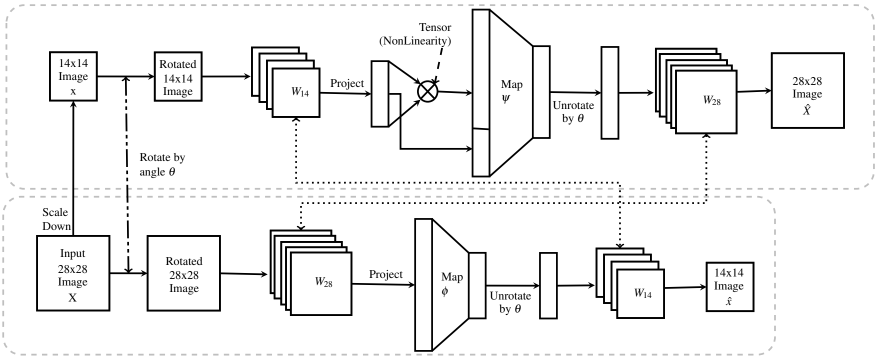

Figure 3: Coupled autoencoder architecture (CAE)

produces $\hat{x}$, a 14x14 image. The two networks are connected and we use the same W_{28}, and W_{14} in the top and bottom half.

We fix two subspaces F^0_{14}, (for 14x14 images) and F^0_{28} (28x28 images) by choosing basis elements with predefined types and multiplicities. We use the basis elements to define an S^1-action on F^0_{14} and F^0_{28} as in Section 2.2. G^1_{14} is the same as F^0_{14}. ψ is an S^1-equivariant map from F^1_{14} to F^0_{28}. In the bottom half, ϕ is an S^1-equivariant map from F^0_{28} to F^0_{14}.

The bottom network takes an image X, rotates it by θ to get Y, and projects the resulting image on the current W_{28} to get $\hat{Y} = W^T_{28}Y$, the Fourier coefficients of the rotated image Y. ϕ is applied to $\hat{Y}$ and we expect to get $\hat{y}$. We apply $-\theta$ to $\phi(\hat{y})$ to cancel the effect of rotation. Using these Fourier coefficients and the current basis W_{14} we construct an image $\hat{x}$, which we expect to be close to x.

The top network takes x and rotates it by the same θ to get y, and project the resulting image on the current W_{14} to get $\hat{y}$, the Fourier coefficients of the rotated image y. Applying ψ to $\hat{y} \oplus (\hat{y} \otimes \hat{y})$ we get a vector in F^0_{28}. We apply $-\theta$ to this vector and use these Fourier coefficients and the current W_{28} to construct an image $\hat{X}$, which we expect to be close to X.

We minimize the sum of the reconstruction errors $|X - \hat{X}|^2 + |x - \hat{x}|^2$. We also use a regularizer $\lambda_1 ||W^T_{28}W_{28} - Id||_2 + \lambda_2 ||W^T_{14}W_{14} - Id||$.

The hyperparameters of this network are the types and multiplicities of F^0_{14} and F^0_{28}. We learn W_{28}, W_{14} and the two S^1-equivariant maps ϕ, ψ.

In Appendix C we give a proof that the learned bases W_{28}, W_{14} are coupled.

3 Experiments - Learning CW-basis

We learn a CW-basis for MNIST (see, LeCun et al. [14]). About 500 samples suffices to learn a good CW-basis, W_{28}. No pre-processing is done to the input images. To deal with downsampling, we implement the coupled-autoencoder of Section 2.6 and learn CAE-W_{14}, CAE-W_{28} and the coupling maps ϕ, ψ. We also learn CW-basis for the Fashion-MNIST dataset [22].Details of the datasets are given in the Appendix G.

In Figure 4 we visualize 64 of the W_{28} basis-vectors learned. We use the learned basis to rotate MNIST images. These results are shown in Figure 5.

We evaluate these bases in terms of image reconstruction error (MSE) and rotation reconstruction error by comparing with scikit-image rotation. We report the errors for MNIST-

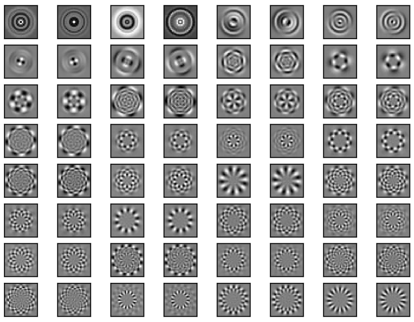
Figure 4: The W_{28} learned in AE

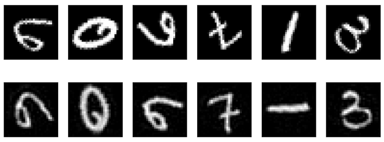
Figure 5: The rotation by the basis learned in experiment 1.

rot. Figure 6, Figure 7 show graphs of the errors for W_{28} and CAE-W_{28} as a function of the number of input samples used to learn the bases. Using only 50 samples, we discover CW-bases good at rotation and reconstruction, in both architectures.

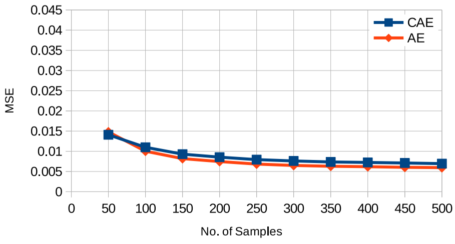

Figure 6: MSE - Reconstruction of images

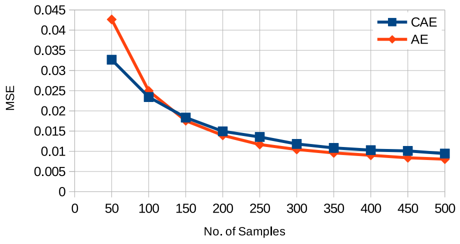

Figure 7: MSE - Rotation of images

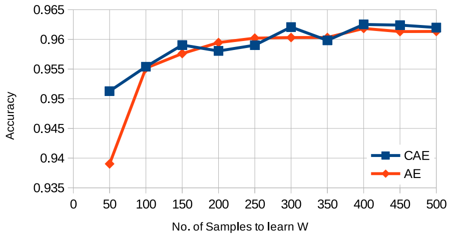

Figure 8: Classification accuracy

4 Experiments - Classification using the learned CW-basis

4.1 Results on Classification, MNIST

In Figure 8 we plot the accuracy of these various W_{28} obtained above when deployed for classification. We plot the accuracy of the classifier as a function of the number of samples used to construct the CW-bases. When the number of samples is as low as 50 a CAE-W_{28} performs better than an AE-W_{28}. Beyond 100 samples the difference is insignificant.

	Samples used to learn W	R / R	R / NR	NR / NR	NR / R
CNN	-	91.06	91.73	99.32	43.28
Spherical CNN	-	88.73	90.83	95.95	91.62
Ours	12000	96.94 (0.35)	97.01 (0.23)	98.15 (0.08)	99.43 (0.06)
14 / 28 Coupled	12000	96.17 (0.35)	96.51 (0.27)	97.10 (0.70)	97.51 (0.09)
14 / 28 Scaled	12000	94.79 (0.59)	95.56 (0.55)	93.86 (0.87)	91.66 (1.36)
Ours	500	96.40 (0.09)	96.64 (0.06)	97.41 (0.09)	98.24 (0.05)
14 / 28 Coupled	500	95.78 (0.12)	95.98 (0.09)	96.53 (0.12)	97.03 (0.07)
14 / 28 Scaled	500	94.37 (0.39)	95.34 (0.23)	92.67 (1.02)	89.62 (1.51)

Table 1: MNIST - Accuracies - rotated, unrotated combinations

	Samples used to learn W	R / R	R / NR	NR / NR	NR / R
CNN	-	80.86 (0.57)	79.83 (0.66)	90.68 (0.31)	20.86 (0.46)
Ours	20000	86.34 (0.18)	84.67 (0.27)	86.70 (0.29)	85.42 (0.18)

Table 2: Fashion MNIST - Accuracies - rotated, unrotated combinations

In Table 1 we give the mean accuracy(and stdev) of our classifier (around 30K parameters) when trained and tested on combinations using MNIST(NR) and MNIST-rot(R). The R/NR column for example denotes the accuracy when trained on MNIST-rot and tested on MNIST. No data augmentation is done to the training data. We compare with CNN (30K parameters) and Spherical CNN (58K parameters).

Let's recall how spherical CNN's are trained and tested on MNIST data - training is done by projecting the MNIST digit onto the northern hemisphere of a sphere (they call this their NR data). They obtain their R data by applying a random $SO(3)$-element to each NR datapoint. Spherical CNN's are trained on R/NR and tested on R/NR. A spherical CNN trained on NR and tested on R performs well, indicating that the features learned are invariant to $SO(3)$. We think of $SO(2)$ as a subgroup of $SO(3)$ consisting of rotations which fix the z-axis. This is the reason we compare our $SO(2)$-invariant classifier with spherical CNN. Since in our experiments NR is MNIST and R is MNIST-rot, for a fairer comparison we train the spherical CNN differently. NR continues to be MNIST projected on the northern hemisphere of a sphere. As R we use MNIST-rot projected on the northern hemisphere of the sphere. Since we can get this applying an $SO(2)$-element to a NR data point, we feel the comparison is fair. We could not compare with Clebsch-Gordon nets (Kondor et al. [12]) since we were not able to run those nets on our GPU.

To deploy our classifier we have to first learn W, a basis of images adapted to rotations. So in the two tables we have a column indicating how many training samples were used to learn the W. So this column is relevant only to our models and is not applicable to either a CNN or a Spherical CNN.

The third row shows the accuracy of our classifier which used CAE-W_{28}. This was obtained using the CAE-architecture trained on all of MNIST-rot train data. In the NR/NR regime standard CNN's achieve an accuracy of about 99.4, better than our classifier. In the other regimes our classifier performs better.

Coupling interchangeability To test how coupled the CAE-W_{28} and CAE-W_{14} are, the classifier trained in row 3 with CAE-W_{28} was presented with down sized 14x14 images for classification. No additional training was done. Instead we use the top half of the coupling network from Figure 3 (in Section 2.6) - given a test image y we compute $\hat{y}$ =CAE-$W_{14}^T y$ and feed the activation $\psi((\hat{y} \otimes \hat{y}) \oplus \hat{y})$ to the trained classifier of row 3. These results are reported in row 4 as [14/28 Coupled]. For comparison we took the 14x14 images and scaled them up to 28x28 and fed them to the trained classifier of row 3. These results are reported in Row 5 as [14/28 Scaled]. CAE-W_{14} performs better, indicating that a coupled-bases retains scale information.

We repeated the same experiment when the coupled network was given 500 samples to learn the CAE-W_{28}, CAE-W_{14}. In row 7 we see there is only a marginal drop in performance.

4.2 Results on Classification, Fashion-MNIST

In Table 2 we report the mean classification accuracy(and stdev) on the Fashion-MNIST data set [22] for which we first created F-MNIST-rot. The CW basis was learned in the AE architecture with Fashion-MNIST as input. We used a four layer classifier (96K parameters). We compare our results with a depth 5 CNN having 102K parameters.

4.3 Implementation details

Our classifiers and autoencoders were implemented in TensorFlow. We use Adam optimizer. We implemented an S^1-equivariant batch normalization, normalizing only neurons of Fourier type 0. The accuracies reported in row 3 of Table 1 is after batch normalization. Without normalization, the accuracies were close to those given in row 6. Hyperparameter settings are discussed in Appendix E. Implementation in reals is discussed in Appendix F.

5 Conclusion

We use simple neural network architectures to learn a CW-basis of images adapted to rotations. Our neural nets operate in the Fourier domain. Tensor nonlinearity arising from taking tensor products of representations is a natural nonlinearity in the Fourier domain, and this is the only nonlinearity our neural networks use. Starting with a CW-basis we build a classifier operating in the Fourier domain using tensor nonlinearity. Our classifier is naturally robust to rotations, and shows good accuracy. Working in the Fourier domain we give a natural definition of bases coupled to work with images in different scales. We use tensor nonlinearity to learn a coupled-bases of images and show how a coupled-bases can be used on downsampled images. It would be nice to find other uses of a coupled-bases.

The ideas and definitions in this paper apply (in theory) to all finite groups, and many interesting infinite groups as well. Using Fourier analysis on the group $\mathbb{Z}_{16} \times \mathbb{Z}_{16}$ we build an autoencoder learning a CW-basis of images equivariant to these translations. In Appendix D we visualize the basis learned and observe that they are (expectedly) similar to the standard Fourier basis of images. We haven't performed many experiments with this basis but believe this is an interesting line of research. We constructed a classifier working in the Fourier domain of $\mathbb{Z}_{28} \times \mathbb{Z}_{28} \times S^1$. For Fashion-MNIST this gives an accuracy of 88.8 in the NR/NR regime, better than what we report using only $SO(2)$-equivariance. It would be interesting to see if the ideas in this paper can be made to work for non-commutative groups.

References

[1] Joan Bruna and Stéphane Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013.

[2] Roger W Carter, Ian Grant Macdonald, and Graeme B Segal. *Lectures on Lie groups and Lie algebras*. Cambridge Univ. Press, 1995.

[3] Taco S. Cohen and Max Welling. Learning the Irreducible Representations of Commutative Lie Groups. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32, pages 1755–1763, February 2014. ISBN 9781634393973.

[4] Taco S. Cohen and Max Welling. Group Equivariant Convolutional Networks. *Proceedings of The 33rd International Conference on Machine Learning*, 48, feb 2016.

[5] Taco S Cohen and Max Welling. Steerable cnns. *arXiv preprint arXiv:1612.08498*, 2016.

[6] Taco S Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. *arXiv preprint arXiv:1801.10130*, 2018.

[7] Remco Duits and Bernhard Burgeth. Scale spaces on lie groups. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 300–312. Springer, 2007.

[8] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *NIPS*, 2015.

[9] Kenichi Kanatani. Shape from texture. In *Group-Theoretical Methods in Image Understanding*, pages 327–355. Springer, 1990.

[10] Imre Risi Kondor. *Group theoretical methods in machine learning*. Columbia University, 2008.

[11] Risi Kondor. N-body networks: a covariant hierarchical neural network architecture for learning atomic potentials. *arXiv preprint arXiv:1803.01588*, 2018a.

[12] Risi Kondor, Zhen Lin, and Shubhendu Trivedi. Clebsch–gordan nets: a fully fourier space spherical convolutional neural network. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 10117–10126. Curran Associates, Inc., 2018.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[14] Yann LeCun, Corinna Cortes, and Christopher J.C. Burges. The MNIST database of handwritten digits. URL `http://yann.lecun.com/exdb/mnist/`.

[15] Roland Memisevic. Learning to relate images. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1829–1846, 2013.

[16] Harry Pratt, Bryan Williams, Frans Coenen, and Yalin Zheng. Fcnn: Fourier convolutional neural networks. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 786–798. Springer, 2017.

[17] Rajesh P N Rao and Daniel L Ruderman. Learning Lie groups for invariant visual perception. *Advances in Neural Information Processing Systems*, 816:810–816, 1999. ISSN 1049-5258. doi: 10.1.1.50.8859.

[18] Laurent Sifre and Stéphane Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1233–1240, 2013.

[19] Jascha Sohl-Dickstein, Jimmy C. Wang, and Bruno A Olshausen. An Unsupervised Algorithm For Learning Lie Group Transformations. *CoRR*, abs/1001.1:8, January 2010.

[20] Nathaniel Thomas, Tess Smidt, Steven M. Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds. *CoRR*, abs/1802.08219, 2018.

[21] Daniel E Worrall, Stephan J Garbin, Daniyar Turmukhambetov, and Gabriel J Brostow. Harmonic networks: Deep translation and rotation equivariance. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 2, 2017.

[22] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.