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

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

Chennai Mathematical Institute
H1, SIPCOT IT Park,
Siruseri, Chennai, India

Abstract

Inspired by recent work of Kondor [11], and Cohen and Welling [2], 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. A novel aspect of our autoencoder and classifier is the use of nonlinearity in the Fourier domain, as in the recent papers of Thomas et al. [19] and Kondor et al. [12] . Our experiments indicate that this nonlinearity is strong enough to 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 Fourier 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 translations of the inputs. Invariance seems to be one reason why CNNs do so well, see [5].

Cohen and Welling [2] were among the first to use the representation theory of compact Lie groups to give invariance a sound mathematical framework. They developed on ideas from earlier works of Rao and Ruderman [17] and Sohl-Dickstein et al. [18] on the Lie group model, and built a model for $SO(2)$, the group of rotations of the plane. 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 are stable under deformations.

Given the incredible success of these deep networks, 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 [3] 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. [20] to learn representations of images invariant to rotations. Using sophisticated ideas from group representation theory Cohen and Welling [4] introduced Steerable CNN's. In order to deal with 3D images Cohen et al. [5] 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 [5] 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] to avoid going back and forth between the image space and the Fourier domain.

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 $T^l_g(f^l)$, for some fixed set of linear transformations $\{T^l_g\}_{g\in G}$. 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 novel nonlinearity in the Fourier domain which allowed them 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. [19]. See also Pratt et al. [16], wherein the authors implement CNNs in the Fourier domain.

In this paper we propose a 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 [2] and [11]. [1] Our model first learns a basis of images adapted to the group $SO(2)$, and we propose a simple autoencoder architecture for this which uses Fourier 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 Fourier nonlinearity. We build a classifier for images using Fourier nonlinearity.

2 Fourier coefficients of functions on S^1 as activations.

A rotation about the origin in the XY plane by an angle θ is given in the standard basis by the matrix $R(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$. This 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 use the notation S^1 instead of $\mathbb{T}$.

The activations in our model will be Fourier coefficients of functions on S^1, just like in the architecture of [12] where the activations were Fourier coefficients (matrices) of functions on $SO(3)$. We first recall the relevant notions from Fourier analysis.

[1] 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.

2.1 Fourier theory on S^1

It is easy to see that every group automorphism of S^1 is given by $e^{i\theta} \mapsto e^{in\theta}$, for some integer n. So the dual group of S^1 is the set of integers, which is the domain for the Fourier transform of a function $f \in L^2(S^1)$. The Fourier transform of f is the function $\hat{f}$ defined as

$$\hat{f}(n) = \frac{1}{2\pi}\int_0^{2\pi} f(\theta)e^{-in\theta}d\theta \tag{1}$$

The functions $\{e^{in\theta}\}_{n\in\mathbb{Z}}$ are an orthonormal basis for functions in $L^2(S^1)$. The inverse Fourier transform then gives us the Fourier expansion of a function on the circle,

$$f(\theta) = \sum_{n=-\infty}^{n=\infty} \hat{f}(n)e^{in\theta} \tag{2}$$

Remark 1. *We say a function $f \in L^2(S^1)$ is transformed by S^1 (or use the terminology, S^1 acts on $f \in L^2(S^1)$) by defining $\theta_0 \cdot f := f_{\theta_0}$ to be the function, $f_{\theta_0}(\theta) := f(\theta - \theta_0)$.*

Clearly $(\theta_1 + \theta_2)\cdot f = \theta_1 \cdot (\theta_2 \cdot f)$ and the identity ($\theta = 0$) fixes f.

It is easy to see from equation 2 that

$$\widehat{f_{\theta_0}}(n) = e^{-in\theta}\hat{f}(n) \tag{3}$$

It will be useful to think of the activations at each layer in a standard CNN's as functions $\mathbb{Z}\times\mathbb{Z} \to \mathbb{C}$, with the activations in the input layer mapping to $\mathbb{C}^3$, one coordinate each for the three R,G,B channels. The activation $f^{\ell+1}$ in layer $\ell+1$ is computed from the activation f^ℓ in layer ℓ by cross-correlation with a filter h^ℓ, also a function from $\mathbb{Z}\times\mathbb{Z} \to \mathbb{C}$.

The activations in our network will be thought of as elements of $L^2(S^1)$ (we postpone the discussion of what to do to the input layer to Section2.2). Borrowing from what happens in a CNN, the activation at layer $\ell+1$ will be computed from the activation at layer ℓ by cross-correlation with a function $h^\ell \in L^2(S^1)$. Recall that cross-correlation of $h \in L^2(S^1)$ with $f \in L^2(S^1)$ is the function

$$[h\star f](\theta_0) = \frac{1}{2\pi}\int_0^{2\pi} h(\theta - \theta_0)f(\theta)d\theta \tag{4}$$

An easy computation shows

$$\widehat{h\star f}(n) = \overline{\hat{h}(n)}\cdot\hat{f}(n) \tag{5}$$

So we have

Proposition 2. *Let $f \in L^2(S^1)$ be an activation and let $h \in L^2(S^1)$. Under the action of $\theta \in S^1$, the Fourier coefficients of the cross-correlation 4 transform as $\widehat{h\star f}(n) \mapsto e^{-in\theta}\widehat{h\star f}(n)$.*

Proof. Using Equation 3, under the action of $\theta \in S^1$, the n-th Fourier coefficient of $[h\star f]_\theta$ is $e^{-in\theta}\widehat{h\star f}(n)$. We check that this is also equal to $\widehat{h\star f_\theta}(n)$. Again, using Equation 3 and Equation 5, this is equal to $e^{-in\theta}\overline{\hat{h}(n)}\cdot\hat{f}(n)$, as required. □

We now take up the input layer.

2.2 Representations of S^1

Assume we are dealing with images of size $N \times N$. We 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[2]. 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. We have $T(0) = T(2\pi)$, is the identity matrix. The transformations compose as $T(\theta_0) \circ T(\theta_1) = T(\theta_0 + \theta_1)$, so $T(\theta)$ has an inverse $T(-\theta)$ i.e. the transformations $T(\theta)$ form a group. So we have a group homomorphism ρ from S^1 to $GL(V)$ (the group of invertible linear transformations of V) given by $\rho(R(\theta)) = T(\theta)$. We will just abbreviate this by $\rho(\theta) = T(\theta)$. In such a scenario one says that V is a representation of S^1, and $g \in S^1$ acts on vectors in V via $g \cdot v = \rho(g)v$. 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 $T(\theta)(I)(x) = I(T(\theta)^{-1}(x))$, as the intensity value now at x is the intensity it was at $T(\theta)^{-1}(x)$. So we get a representation of S^1 on the space of linear functions, V^*.

Recall the following theorem from harmonic analysis. We sketch a proof in the appendix.

Theorem 3. *If W is a representation of S^1 then $W = \widehat{\oplus}_{n\in\mathbb{Z}} W_n$ where*

$$W_n = \{w \in W : \theta \cdot w = e^{-in\theta} w, \text{ for all } \theta \in S^1\}$$

Definition 4. *Let W be a representation of S^1. A subspace $W' \subseteq W$ is said to be invariant under S^1 if for all $\theta \in S^1$, and $w' \in W'$ we have $\theta \cdot w' \in W'$. We say W is irreducible if there is no proper subspace which is invariant under S^1.*

From the above theorem it is clear that each W_n is an invariant subspace. It is also clear that the only irreducible subspaces are one dimensional subspaces contained in W_n, for some n. We say the irreducible subspace of type n occurs in W with multiplicity $\dim(W_n)$.

We apply the above theorem to the space V^*, of $N \times N$ images under the action of S^1. Let $\{b_n^j\}$ be a basis of V_n^*. Each input image I can be written as $I = \sum_n \sum_{j=1}^{j=m_n} p_n^j b_n^j$. Here m_n is the multiplicity of the irreducible of type n. Using Theorem 3 it follows that when the image is transformed by θ, the $\{p_n^j\}$'s transform as $p_n^j \mapsto e^{-in\theta} p_n^j$, exactly like the Fourier coefficients of the cross-correlation, see Proposition 2. So we call $\{p_n^j\}_{j=1}^{j=m_n}$, the Fourier coefficients of the image I of type n. The totality of Fourier coefficients of an image of all types are the activations of the input layer.

2.3 S^1-equivariant maps

Definition 5. *If F and H are representations of S^1, an S^1-morphism (aka a S^1-equivariant map) from F to H is a (complex)-linear map ϕ from F to H such that for all $f \in F$, and $\theta \in S^1$, $\phi(\theta \cdot f) = \theta \cdot \phi(f)$.*

Now if F is an irreducible representation of type n with basis vector f and H is an irreducible representation of type m with basis vector h, and $n \neq m$, then the only equivariant map from F to H is the zero map since $\phi(\theta \cdot f) = \phi(e^{-in\theta} f) = e^{-in\theta}\phi(f)$ but the right hand side in the definition above gives $e^{-im\theta}\phi(f)$. On the other hand if they are of the same type,

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

then sending f to any complex multiple of h gives us an S^1-morphism. Note that any linear combination of vectors of the same type, is a vector of the same type. It follows that if the multiplicity of the irreducible representation of type n in F, H are f_n, h_n respectively, the dimension of the space of S^1-equivariant maps from F to H is $f_n \cdot h_n$, and such a map is given by an $h_n \times f_n$ matrix.

2.4 Fourier nonlinearity, Classifier and Autoencoder architectures

2.5 The classifier, a first attempt

Our classifier network operates in the Fourier domain. The activations in layer j are the collection of Fourier coefficients

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

Here m^ℓ_i is the multiplicity of Fourier coefficient of type i, $i = -r, -r+1, \ldots, k$ in layer ℓ. This makes sense even in the input layer as per the discussion in Section 2.2. We assume in this section that we know the types and multiplicities of Fourier coefficients in the input layer, and we also know a basis W of the image space and the action of S^1 on W. We will describe how to do this in Section 2.7. The types and multiplicities of the Fourier coefficients in the other layers will be *hyperparameters*, which we will tune.

We can think of layer ℓ as representing a vector space F^ℓ spanned by basis elements indexed by the Fourier coefficients. Since we know the action of S^1 on each Fourier coefficient, F^ℓ is in fact a representation of S^1 (extend the action linearly). Since we are only interested in S^1-equivariant maps between layers ℓ, $\ell+1$, it follows from the discussion following Definition 5 that the dimension of our search space of equivariant maps between F^ℓ and $F^{\ell+1}$ is $\sum_{i=-r}^{i=k} m^\ell_i m^{\ell+1}_i$. And the variables of the map can be put in a block diagonal matrix with blocks of size $m^{\ell+1}_i \times m^\ell_i$. These are the variables we will learn in a supervised manner.

2.5.1 Fourier nonlinearity

There is one issue with the above formulation, there is *no nonlinearity* in the network. To bring in nonlinearity, consider the tensor product $F^{\ell+1} \otimes F^{\ell+1}$. This vector space is a representation of S^1, and is spanned by basis vectors of the form $f^{\ell+1}_{j,k} \otimes f^{\ell+1}_{i,m}$. Under the action of $\theta \in S^1$ the basis vector transforms as $f^{\ell+1}_{j,k} \otimes f^{\ell+1}_{i,m} \mapsto e^{-i(k+m)\theta} f^{\ell+1}_{j,k} \otimes f^{\ell+1}_{i,m}$. To bring in nonlinearity, we take as the output of layer ℓ not only $F^{\ell+1}$, but also the unordered products of pairs of Fourier coefficients, taking note of their types. We call this *Fourier or tensor nonlinearity*.

2.6 The final classifier

Figure 1 shows a 2 layer classification network for 28×28 images, incorporating nonlinearity. We learn two S^1-equivariant maps ϕ_1, ϕ_2. The input image x is projected onto neurons in layer 0 using the known basis W and their types to obtain the Fourier coefficients of x. ϕ_1 applied to the activations in layer 0 gives us the activations in layer 1, $\phi_1(W^T x)$. The output of layer 1 is $\phi_1(W^T x) \oplus \phi_1(W^T x) \otimes \phi_1(W^T x)$. This is the input to the neurons in layer 2. These blocks repeat in a deeper network.

We initialize ϕ_1, ϕ_2 at random, and minimize classification error.

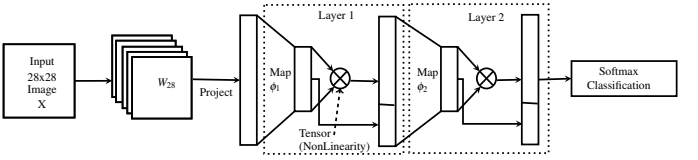

Figure 1: Classification network

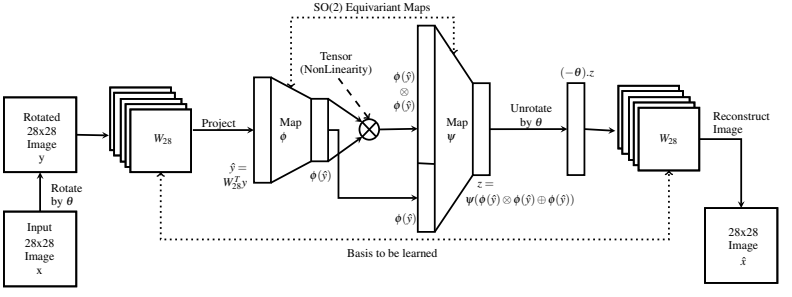

Figure 2: Autoencoder architecture (AE).

2.7 The autoencoder

It remains to describe the architecture to learn a basis $\{b_n^j\}_{j,n}$ of images under rotations described at the end of Section 2.2. The problem of discovering such a basis of images under rotations was first considered by Cohen and Welling [2]. So we call such a basis of images adapted to rotations a CW-basis. In [2] an expectation maximization algorithm was proposed for this.

The activations of our autoencoder are a collection of Fourier coefficients (the notation being analogous to what we used for the classifier). These are *hyperparameters* to be tuned.

$$\{a^\ell_{1,-s}, a^\ell_{2,-s}, \dots, a^\ell_{b^\ell_{-s},-s}, \dots, \dots, a^\ell_{1,0}, a^\ell_{2,0}, \dots, a^\ell_{b^\ell_0,0}, \dots, \dots, \dots, a^\ell_{1,k} a^\ell_{2,k}, \dots, a^\ell_{b^\ell_k,k}\}$$

Figure 2 gives the schematic diagram for an autoencoder with four layers of activations. The types and multiplicities of the Fourier coefficients in layer 0, layer 2 and layer 3 are identical. We learn a CW-basis W of the input image space, with Fourier types and multiplicities as in layer 0. We also learn S^1-equivariant maps ϕ, ψ ($SO(2)$-equivariant maps in the figure).

The input image is rotated by an angle θ and projected onto the current basis W to obtain $\hat{y}$, the Fourier coefficients of the rotated image, the activations of layer 0. Due to the rotation, the Fourier coefficients of the image of type n get scaled by $e^{-in\theta}$. An S^1-equivariant map ϕ applied to layer 0 activations gives us the activations in layer 1, $\phi(\hat{y})$. To bring in nonlinearity we take the activations $\phi(\hat{y}) \oplus \phi(\hat{y}) \otimes \phi(\hat{y})$. A second S^1-equivariant map ψ applied to these activations, gives us the activations in layer 2. Since these coefficients have the same types and multiplicities as that in layer 0, we scale activations of type n here by $e^{in\theta}$, to cancel the effect of the rotation to the input. We expect to recover the Fourier coefficients of the image now. We reconstruct the image using the current basis W and the coefficients in layer 3. We compare the reconstructed image with the original image and minimize reconstruction loss. We want W to be orthonormal, so we apply the regularization $\lambda||W^T W - Id||_2$.

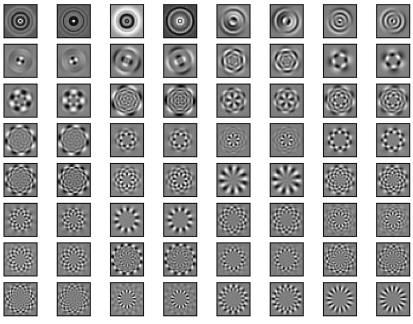

Figure 3: The W_{28} learned in AE

Figure 4: The rotation by the basis learned in experiment 1.

2.8 The coupled autoencoder

We find CW-bases of images in two different scales, simultaneously, with each influencing the discovery of the other, so as to be able to use a classifier trained on large size images on downsampled inputs. In Appendix B we give a schematic diagram to learn a coupled bases CAE-W_{14} and CAE-W_{28} and coupling maps ϕ, ψ. We first formulate the question precisely in the language of tensor algebras, and define the notion of a coupled-bases and coupled-features.

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 Appendix B and learn CAE-W_{14}, CAE-W_{28} and the coupling maps ϕ, ψ. We also learn CW-basis for the Fashion-MNIST dataset [21].Details of the datasets are given in the Appendix F.

In Figure 3 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 4.

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-rot. Figure 5, Figure 6 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.

4 Experiments - Classification using the learned CW-basis

4.1 Results on Classification, MNIST

In Figure 7 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.

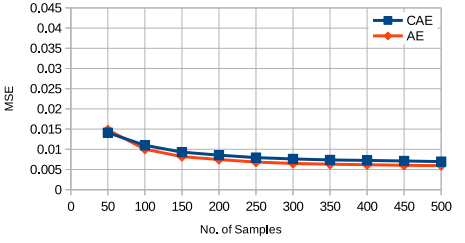

Figure 5: MSE - Reconstruction of images

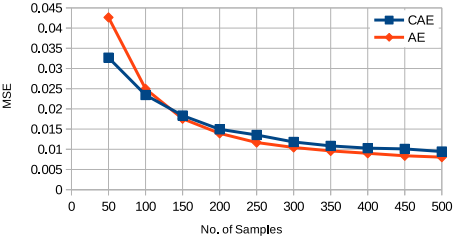

Figure 6: MSE - Rotation of images

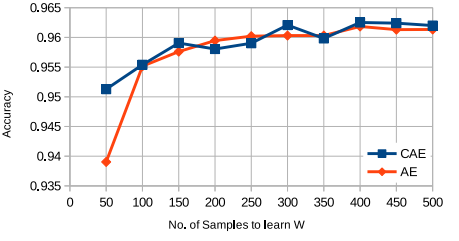

Figure 7: Classification accuracy

In Table 1 we give the mean accuracy(and stdev) of our classifier 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 (comparable parameter space) and Spherical CNN . Since Spherical CNN's work by wrapping an image around the 2-sphere causing distortion, this comparison is unfair. For a fairer comparison we use MNIST wrapped around the Northern hemisphere as NR and MNIST-rot wrapped around the northern hemisphere as R. We were not able to run Clebsch-Gordan nets (Kondor et al. [12]) on our GPU.

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. Our classifier performs better in all scenarios.

Coupling interchangeability To test how coupled the CAE-W_{28} and CAE-W_{14} are, the

	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

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 8 (in Appendix B) - 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} outperforms, 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 [21]. We rotate each data point around the origin by an angle chosen uniformly between 0 and 2π to create F-MNIST-rot. The CW basis was learned in the AE architecture with Fashion-MNIST as input. Our classifier for this dataset is a four layer network (96K parameters). We compare our results with a depth 5 CNN having 102K parameters.

4.3 Implementation details

Our classifier and autoencoder were implemented in TensorFlow. We used Adam optimizer. We also implemented an S^1-equivariant batch normalization, normalizing only neurons of Fourier type 0. Batch normalization does improves accuracy. The accuracies reported in row 3 of Table 1 is after batch normalization. Without batch normalization, the accuracies were close to those given in row 6. More details of the implementation are in Appendix D.

In the entire paper we assumed we were working over complex numbers for ease of presentation. Our implementation was however done over reals. In Appendix E we discuss this transition. This also explains why we continue to use $SO(2)$ in our title.

5 Conclusion

We use simple neural network architectures to learn a CW-basis of images adapted to rotations. We show that Fourier nonlinearity is strong enough to learn such a CW-basis. Starting with a CW-basis we build a classifier in the Fourier domain using tensor nonlinearity in the Fourier domain. Our classifier is naturally robust to rotations, and shows good accuracy. The notion of a coupled CW-bases is a natural way to deal with the issue of downsampling. Using Fourier nonlinearity we learn a coupled CW-bases of images in different scales, simultaneously. Although we only consider the group $SO(2)$, the ideas and definitions in this paper apply 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. We do not give details. In Appendix C we visualize the basis learned and see that they are very similar to the standard Fourier basis of images. Using the group $\mathbb{Z}_{28} \times \mathbb{Z}_{28} \times S^1$ constructed a classifier in the Fourier domain of $\mathbb{Z}_{28} \times \mathbb{Z}_{28} \times S^1$. For Fashion-MNIST we get an accuracy of 88.8 in the NR/NR regime, better than what we report using only $SO(2)$.

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] 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.

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

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

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

[6] Roe W Goodman and Nolan R Wallach. *Symmetry, Representations, and Invariants*. Graduate Texts in Mathematics. Springer, Dordrecht, 2009.

[7] Kennether Hoffman and Ray Kunze. Linear algebra, 2nd, 1990.

[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] 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.

[19] 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.

[20] 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.

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

Appendix A Proof of Theorem 3

Proof. Since ρ is a homomorphism it follows that each $T(\theta)$ is a diagonalizable operator (see Goodman and Wallach [6][Theorem 1.3.5]). Since the $T(\theta)$ for various θ commute, the family of operators $T(\theta)$ is a *commuting family of diagonalizable operators* and so they can be *simultaneously* diagonalized (see Hoffman and Kunze [7][Chapter 6]). There is a invertible orthogonal matrix P such that $PT(\theta)P^{-1}$ is a diagonal matrix for all θ. So we can write $W = \oplus_j W_j$ with $T(\theta)W_j \subseteq W_j$, for all j since W_j are simultaneous eigen spaces for the $T(\theta)$. Each W_j is in fact a one-dimensional sub-representation of S^1 and we call it an irreducible representation of S^1, say spanned by w_j. Since $T(\theta) = T(\theta + 2\pi)$ and $T(0) = Id$ (the identity matrix) it follows that there is an integer n_j such that $T(\theta)w_j = e^{-in_j\theta}w_j$. This integer n_j is the *type* of the irreducible representation W_j and we write $W = \oplus_j W_{j,n_j}$ to indicate the type. The span of all w_j of the same type $n_j = n$ gives us W_n and we can write $W = \oplus_n W_n$ completing the proof. The proof works even if W is infinite dimensional. □

Appendix B Coupled bases

We first formulate the question precisely by making a definition.

Definition 6. *We say S^1-representations U and $\tilde{U}$ are coupled if U is the image of a S^1-morphism ϕ of a finite dimensional sub-representation of the tensor algebra of $\tilde{U}$, and $\tilde{U}$ is the image of a S^1-morphism ψ, of a finite dimensional sub-representation of the tensor algebra of U.*

Definition 7. *Let U be the vector space of 14×14 images and let W be the vector space of 28×28 images. Assume the rotation group S^1 acts on both. We say a Cohen-Welling basis X of U is coupled to a Cohen-Welling basis Y of W if the vector space dual of the subspace spanned by X (with its S^1-action) is coupled to the vector space dual of the subspace spanned by Y.*

Unraveling the definition, there is a sub-representation $\tilde{X}$ of $X \oplus X \otimes X \oplus X \otimes X \otimes X \oplus \ldots$ such that $\phi(\tilde{X}) = Y$ and a sub-representation $\tilde{Y}$ of $Y \oplus Y \otimes Y \oplus Y \otimes Y \otimes Y \oplus \ldots$ such that $\psi(\tilde{Y}) = X$. The definition above is motivated by our question of whether one can build CW-bases in two different scales, with one influencing the discovery of the other, so as to be able to use a classifier trained on larger images on downsampled images also, with little loss in accuracy and without additional training. 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 influence we are looking for.

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

We give a schematic diagram to discover a coupled-bases.

B.1 Coupled Autoencoder(CAE) architecture

The schematic diagram for learning a coupled-bases is given in Figure 8. The idea is similar to the autoencoder described in Section 2.7. However 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 scaled down version of the image x to the network. The network on top takes x and produces $\hat{X}$, a

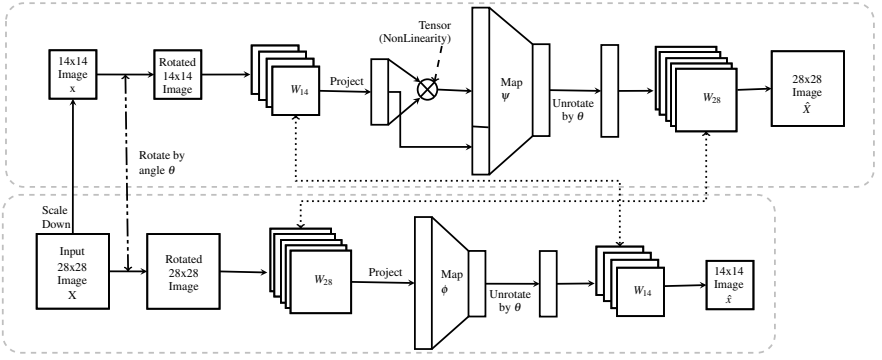

Figure 8: Coupled autoencoder architecture (CAE)

28x28 image. The bottom network takes X and 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 layer. The network on top has four layers of activations, layers 0,1,2,3, as in the autoencoder in Figure 2. The types and multiplicities of the activations in layer 2 and layer 3 are the same (as multisets). The bottom half of the network has three layers of activations. The types and multiplicities of the Fourier coefficients in layer 1 and layer 2 in the bottom are the same (as multisets).

The types and multiplicities of activations in layer 2 on the top are the same as that of layer 0 in the bottom. The types and multiplicities of activations in layer 1 on the bottom are the same as that of layer 0 in the top.

We learn the W_{14}, W_{28} and two $SO(2)$-equivariant maps ψ and ϕ.

Both W_{28}, W_{14} are initialized at random. The bottom network takes an image X, rotates it by θ and projects the resulting image on the current W_{28}, to get $\hat{Y} = W_{28}^T Y$, the Fourier coefficients of the rotated image, the activations in layer 0 in the bottom. Applying ϕ we get $\phi(\hat{Y})$ the activations in layer 1 in the bottom. To cancel the effect of rotation, we unrotate, i.e. multiply the the activations in layer 1 in the bottom of type n by $e^{in\theta}$, to get the activations in layer 2 in the bottom. Recall these have the same types and multiplicities as that of layer 0 on the top, the types and multiplicities of 14×14 images. We take a linear combination of the basis elements of the current W_{14}, with coefficients the activations in layer 2 in the bottom to get $\hat{x}$, a 14×14 image.

The top network takes x and rotates it by the same θ. This is projected onto the current W_{14} to get $\hat{y}$ the Fourier coefficients or activations in layer 0 on the top. We bring in Fourier nonlinearity and so the activations in layer 1 on the top are now $\hat{y} \oplus (\hat{y} \otimes \hat{y})$ (we make a note of their types). We apply ψ to the activations in layer 1. To cancel the effect of rotation we unrotate, by scaling the activations of type n in layer 1 by $e^{in\theta}$. The types and multiplicities of these Fourier coefficients match that of 28×28 images. So we use the current W_{28} and the activations in layer 3 on the top to construct a 28×28 image, $\hat{X}$. We minimize the sum of the reconstruction errors $|X - \hat{X}|^2 + |x - \hat{x}|^2$.

Appendix C Bases learned for the translation group.

The bases learned for the translation group are visualized below.

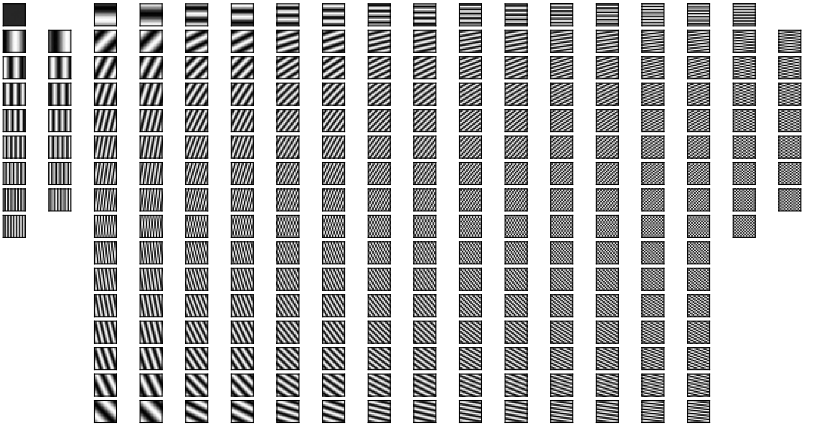

Figure 9: The W_{16} for translation group learned in AE

Appendix D Implementation details

In this section we give details of the hyperparameters of our model, the types and multiplicities of the activations in different layers. The type and multiplicities of the Fourier coefficients in layer 0 and layer 1 of the autoencoder architecture (see Figure 2) used to discover the CW-basis of 28×28 images are given below. The first row gives the types and multiplicities of the activations in layer 0 and the second gives the types and multiplicities of activations in layer 1. Multiplicities are given in brackets. Here $\pm 1 - 4(5)$ means that types ± 1,± 2,± 3,and ± 4 were chosen to have mutiplicity 5.

$$0(10), \pm 1 - 4(5), \pm 5 - 9(4), \pm 10 - 14(3), \pm 15 - 19(2), \pm 20 - 24(1)$$

$$0(8), \pm 1 - 4(4), \pm 5 - 9(3), \pm 10 - 14(2), \pm 15 - 19(1)$$

Appendix E Working over reals in Tensor Flow

Let S^1 act on a vector space W. It can be shown that for every irreducible subspace W_j with type $n_j \neq 0$ and basis vector w_j, there is an irreducible subspace with type $-n_j$, with basis vector the complex conjugate $\bar{w}_j$ of w_j, see for example [9][2.3.1]. In fact this can be deduced from our identification of $SO(2)$ with diagonal matrices having entries $e^{-i\theta}, e^{i\theta}$. Setting $b_{j1} = \frac{w_j + \bar{w}_j}{2}$ and $b_{j2} = \frac{w_j - \bar{w}_j}{2i}$ it is easy to see that b_{j1}, b_{j2} are real and they transform according to the columns of the matrix $R(n_j\theta)$. This two dimensional subspace (over the real numbers $\mathbb{R}$) of the image space is invariant to the real rotation group, $SO(2)$ and is irreducible for the action of $SO(2)$ on W. For our implementation purposes we work over real numbers. We call n_j the type of this irreducible representation of $SO(2)$. On the other hand subspaces W_j of type $n_j = 0$ with basis vector w_j, are invariant vectors, and satisfy $\rho(R(\theta))w_j = w_j$. These are one dimensional irreducible representations of $SO(2)$. Over reals Theorem 3 takes

the form, $W = \oplus_{n\geq 0} W_n$. Here W_n is the subspace spanned by $SO(2)$-invariant subspaces of type n. Working over $\mathbb{R}$, the tensor product of two irreducible $SO(2)$ representations of type $s \geq t \geq 0$ splits into a direct sum of two irreducible $SO(2)$ representations of type $s+t, s-t$.

Appendix F Datasets used

We have used MNIST and MNIST_rot dataset in our experiments. Each sample in the dataset is a 28x28 gray scale image. MNIST dataset contains handwritten digits(upright, we call them NR). There are 60000 train samples and 10000 test samples. MNIST_rot dataset contains handwritten digits(rotated, we call them R) MNIST_rot has 12000 train samples and 50000 test samples.

We have used Fashion MNIST in our experiments. Each sample in the dataset is a 28x28 gray scale image. There are 10 different classes. Each label is one among the ten labels index by 0 to 9. The label index and the corresponding description is given in Table 3

Label	Description
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

Table 3: Fashion MNIST - Label index and Description

There are 60000 train samples and 10000 test samples. We call this set as upright (NR). For the rotated case (we call it as R), we rotate each sample in the test and the train by a random angle between 0 and 360.