Harmonic Networks: Deep Translation and Rotation Equivariance

Daniel E. Worrall, Stephan J. Garbin, Daniyar Turmukhambetov and Gabriel J. Brostow
{d.worrall, s.garbin, d.turmukhambetov, g.brostow}@cs.ucl.ac.uk
University College London

Abstract

Translating or rotating an input image should not affect the results of many computer vision tasks. Convolutional neural networks (CNNs) are already translation equivariant: input image translations produce proportionate feature map translations. This is not the case for rotations. Global rotation equivariance is typically sought through data augmentation, but patch-wise equivariance is more difficult. We present Harmonic Networks or H-Nets, a CNN exhibiting equivariance to patch-wise translation and 360-rotation. We achieve this by replacing regular CNN filters with circular harmonics, returning a maximal response and orientation for every receptive field patch.

H-Nets use a rich, parameter-efficient and low computational complexity representation, and we show that deep feature maps within the network encode complicated rotational invariants. We demonstrate that our layers are general enough to be used in conjunction with the latest architectures and techniques, such as deep supervision and batch normalization. We also achieve state-of-the-art classification on rotated-MNIST, and competitive results on other benchmark challenges.

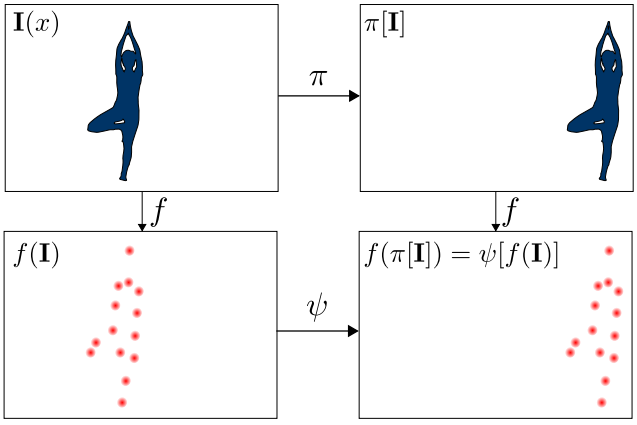

Figure 1. Patch-wise translation equivariance in CNNs arises from translational weight tying, so that a translation π of the input image $\mathbf{I}$, leads to a corresponding translation ψ of the feature maps $f(\mathbf{I})$, where $\pi \neq \psi$ in general, due to pooling effects. However, for rotations, CNNs do not yet have a feature space transformation ψ 'hard-baked' into their structure, and it is complicated to discover what ψ may be, if it exists at all. Harmonic Networks have a hard-baked representation, which allows for easier interpretation of feature maps—see Figure 3.

1. Introduction

We tackle the challenge of representing 360°-rotations in convolutional neural networks (CNNs) [14]. Currently, the convolutional layers of CNNs are constrained by design to map an image to a feature vector, and for *translated* versions of the image to map to proportionally-translated versions of the same feature vector [16] (ignoring edge effects)—see Figure 1. However, until now, if one *rotates* the input to a CNN, then the feature vectors do not necessarily rotate in a meaningful or easy to predict manner. The sought-after property, to directly relate input transformations to feature vector transformations, is called *equivariance*.

A special case of equivariance is invariance, which is when feature vectors remain constant under all transformations of the input. This can be a desirable property globally for a model, such as a classifier, but we should be careful not to restrict all intermediate levels of processing to be transformation invariant. For example, consider trying to detect a deformable object, such as a butterfly. The pose of a butterfly's wings is limited in range, and so there are only certain poses, which our detector should normally see. A transformation invariant detector, good at detecting wings, would detect them whether they were bigger, further apart, rotated, etc., and it would encode all these cases with the same representation. It would fail to notice nonsense situations, however, such as a butterfly with wings rotated past the usual range, because it has thrown that extra pose information away. An equivariant detector, on the other hand, does not dispose of local pose information, and so it hands on a richer and more useful representation to downstream processes.

Equivariance conveys more information about an input to downstream processes, it also constrains the space of possible learned models to those that are valid under the rules of natural image formation [24]. This makes learning more reliable and helps with generalization. For instance, consider CNNs. The key insight is that the statistics of natural images, embodied in the correlations between pixels, are a) invariant to translation, and b) highly localized. Thus features at every layer in a CNN are computed on local receptive fields, where weights are shared

across translated receptive fields. This weight-tying serves both as a constraint on the translational structure of image statistics, and as an effective technique to reduce the number of learnable parameters—see Figure 1. In essence, translational equivariance has been 'baked' into the architecture of existing CNN models. We do the same for rotation and refer to it as *hard-baking*.

The current widely accepted practice to cope with rotation is to train with aggressive data augmentation [11]. This certainly improves generalization, but is not exact, fails to capture local equivariances, and does not ensure equivariance at every layer within a network. How to maintain the richness of local rotation information, is what we present in this paper. Another disadvantage of data augmentation is that it leads to the so-called *black-box* problem, where there is a lack of feature map interpretability. Indeed, close inspection of first-layer weights in a CNN reveals that many of them are rotated, scaled, and translated copies of one another [27]. Why waste computation learning all of these redundant weights?

In this paper, we present *Harmonic Networks*, or *H-Nets*. They design patch-wise 360°-rotational equivariance into deep image representations, by constraining the filters to the family of *circular harmonics*. The circular harmonics are *steerable filters* [5], which means that we can represent all rotated versions of a filter, using just a finite, linear combination of *steering bases*. This overcomes the issue of learning multiple filter copies in CNNs, guarantees rotational equivariance, and produces feature maps that transform predictably under input rotation.

2. Related Work

Multiple existing approaches seek to encode rotational equivariance into CNNs. Many of these follow a broad approach of introducing filter or feature map copies at different rotations. None has dominated as standard practice.

Steerable filters At the root of H-Nets lies the property of *filter steerability* [5]. Filters exhibiting steerability can be constructed at any rotation as a finite, linear combination of base filters. This removes the need to learn multiple filters at different rotations, and has the bonus of constant memory requirements. As such, H-Nets could be thought of as using an infinite bank of rotated filter copies. A work, which combines steerable filters with learning is [18]. They build shallow features from steerable filters, which are fed into a kernel SVM for object detection and rigid pose regression. H-Nets use the same filters with an added rotation offset term, so that filters in different layers can have orientation-selectivity relative to one another.

Hard-baked transformations in CNNs While H-Nets hard-bake patch-wise 360°-rotation into the feature representation, numerous related works have encoded equivariance to discrete rotations. The following works can be grouped into those, which encode global equivariance versus patch-wise equivariance, and those which rotate filters versus feature maps.

[2] introduce equivariance to 90°-rotations and dihedral flips in CNNs by copying the transformed filters at different rotation–flip combinations. They lay out a general mathematical framework for this, based on Group Theory. [19] use an even larger number of rotations for texture classification and [21] also use many rotated handcrafted filter copies, opting not to learn the filters. To achieve equivariance to a greater number of rotations, these methods would need an infinite amount of computation. H-Nets achieve equivariance to all rotations, but with finite computation.

[4] feed in multiple rotated copies of the CNN input and fuse the output predictions. [12] do the same for a broader class of global image transformations, and propose a novel per-pixel pooling technique for output fusion. As discussed, these techniques lead to global equivariances only and do not produce interpretable feature maps. [3] go one step further and copy each feature map at four 90°-rotations. They propose 4 different equivariance preserving feature map transformations. Their CNN is similar to [2] in terms of what is being computed, but rotating feature maps instead of filters. A downside of this is that all inputs and feature maps have to be square; whereas, we can use any sized input.

Learning generalized transformations Others have tried to learn the transformations directly from the data. While this is an appealing idea, as we have said, for certain transformations it makes more sense to hard-bake these in for interpretability and reliability. [20] construct a higher-order Boltzmann machine, which learns tuples of transformed linear filters in input–output pairs. Although powerful, they have only shown this to work on shallow architectures. [6] introduced *capsules*, units of neurons designed to mimic the action of cortical columns. Capsules are designed to be invariant to complicated transformations of the input. Their outputs are merged at the deepest layer, and so are only invariant to global transformation. [17] present a method to regress equivariant feature detectors using an objective, which penalizes representations, which lie far from the equivariant manifold. Again, this only encourages global equivariance; although, this work could be adapted to encourage equivariance at every layer of a deep pipeline.

3. Problem analysis

Many computer vision systems strive to be view independent, such as object recognition, which is invariant to affine transformations, or boundary detection, which is equivariant to non-rigid deformations. H-Nets hard-bake 360°-rotation equivariance into their feature representation, by constraining the convolutional filters of a CNN to be from the family of circular harmonics. Below, we outline the formal definition of equivariance (Section 3.1), how the circular harmonics exhibit rotational equivariance (Section 3.2) and some properties of

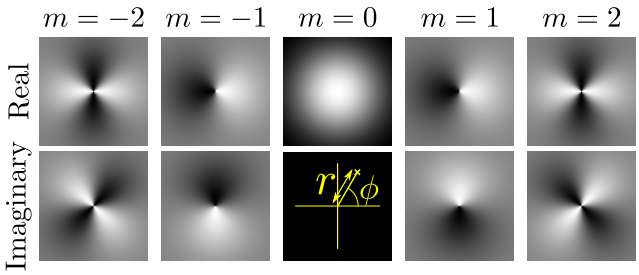

Figure 2. Real and imaginary parts of the complex Gaussian filter $\mathbf{W}_m(r,\phi';e^{-r^2},0)=e^{-r^2}e^{im\phi}$, for some rotation orders. As a simple example, we have set $R(r)=e^{-r^2}$ and $\beta=0$, but in general we learn these quantities. Cross-correlation, of a feature map of rotation order n with one of these filters of rotation order m, results in a feature map of rotation order $m+n$. Note the negative rotation order filters have flipped imaginary parts compared to the positive orders.

the circular harmonics, which we must heed for successful integration into the CNN framework (Section 3.2).

Continuous domain feature maps In deep learning we use feature maps, which live in a discrete domain. We shall instead use continuous spaces, because the analysis is easier. Later on in Section 4.2 we shall demonstrate how to convert back to the discrete domain for practical implementation, but for now we work entirely in continuous Euclidean space.

3.1. Equivariance

Equivariance is a useful property to have because transformations π of the input produce predictable transformations ψ of the features, which are interpretable and can make learning easier. Formally, we say that feature mapping $f:\mathcal{X}\to\mathcal{Y}$ is *equivariant* to a group of transformations if we can associate every transformation $\pi\in\Pi$ of the input $\mathbf{x}\in\mathcal{X}$ with a transformation $\psi\in\Psi$ of the features; that is,

$$\psi[f(\mathbf{x})]=f(\pi[\mathbf{x}]). \tag{1}$$

This means that the order, in which we apply the feature mapping and the transformation is unimportant—they *commute*. An example is depicted in Figure 1, which shows that in CNNs the order of application of integer pixel-translations and the feature map are interchangeable. An important point of note is that $\pi\neq\psi$ in general, so if we seek for Π to be rotations in the image domain, we do not require to find the set of f, such that Ψ "looks like" a rotation in feature space, rather we are searching for the set of f, such that there exists an *equivalent* class of transformations Ψ in feature space. A special case of equivariance is *invariance*, when $\Psi=\{\mathbb{I}\}$, the identity.

3.2. The Complex Circular Harmonics

With data augmentation CNNs may learn some rotation equivariance, but this is difficult to quantify [16]. H-Nets take the simpler approach of hard-baking this structure in. If f is

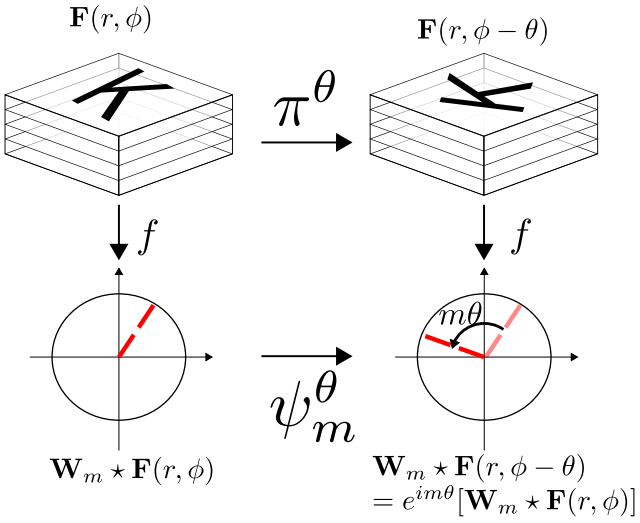

Figure 3. DOWN: Cross-correlation of the input patch with $\mathbf{W}_m$ yields a scalar complex-valued response. ACROSS-THEN-DOWN: Cross-correlation with the θ-rotated image yields another complex-valued response. BOTTOM: We transform from the unrotated response to the rotated response, through multiplication by $e^{im\theta}$.

the feature mapping of a standard convolutional layer, then 360°-rotational equivariance can be hard-baked in by restricting the filters to be of the from the circular harmonic family (proof in Supplementary Material)

$$\mathbf{W}_m(r,\phi;R,\beta)=R(r)e^{i(m\phi+\beta)}. \tag{2}$$

Here r,ϕ are the spatial coordinates of image/feature maps, expressed in polar form, $m\in\mathbb{Z}$ is known as the *rotation order*, $R:\mathbb{R}_+\to\mathbb{R}$ is a function, called the *radial profile*, which controls the overall shape of the filter, and $\beta\in[0,2\pi)$ is a *phase offset* term, which gives the filter orientation-selectivity. During training, we learn the radial profile and phase offset terms. Examples of the real component of $\mathbf{W}_m$ for a 'Gaussian envelope' and different rotation orders are shown in Figure 2. Since we are dealing with complex-valued filters, all filter responses are complex-valued, and we assume from now on that the reader understands that all feature maps are complex-valued, unless otherwise specified.

Rotational Equivariance of the Circular Harmonics Some deep learning libraries implement cross-correlation $\star$ rather than convolution $*$, and since the understanding is slightly easier to follow, we consider correlation below[1]. Consider correlating a circular harmonic of order m with a rotated image patch. We assume that the image patch is only able to rotate locally about the origin of the filter. This means that the response of the cross-correlation is only a scalar function of input image patch rotation θ. Furthermore this response is complex-valued.

[1]Strictly, cross-correlation with complex functions requires that one of the arguments is conjugated, but we do not do this in our model/implementation.

Using the notation from Equation 1, and recalling that we are working in polar coordinates r, ϕ, counter-clockwise rotation of an image $\mathbf{F}(r,\phi)$ about the origin by an angle θ is $\mathbf{F}(r,\phi-\theta)$, so the rotation transformation can be written as $\mathbf{F}(r,\pi^\theta[\phi]) = \mathbf{F}(r,\phi-\theta)$. It is a well-known result that [18, 5] (proof in Supplementary Material)

$$[\mathbf{W}_m \star \mathbf{F}(r,\pi^\theta[\phi])](\theta) = e^{im\theta}[\mathbf{W}_m \star \mathbf{F}(r,\phi)](\theta), \qquad (3)$$

where we have written $\mathbf{W}_m$ in place of $\mathbf{W}_m(r,\phi;R,\beta)$ for brevity. We see that the response to a θ-rotated image $\mathbf{F}(r,\pi^\theta[\phi])$ with a circular harmonic of order m is equivalent to the cross-correlation of the unrotated image $\mathbf{F}(r,\phi)$ with the harmonic, followed by multiplication by $e^{im\theta}$. While the rotation is done in input space, multiplication by $e^{im\theta}$ is performed in feature space, and so, using the notation from Equation 1, $\psi^\theta_m[\bullet] = e^{im\theta}\cdot\bullet$. This process is shown in Figure 3. Note that we have included a subscript m on the feature space transformation. This is important, because the kind of feature space transformation we apply is dependent on the rotation order of the harmonic. Because the phase of the response rotates with the input at frequency m, we say that the response is an m-equivariant feature map. By thinking of an input image as a complex-valued feature map with zero imaginary part, we could think of it as 0-equivariant.

The use of rotation orders is central to how we define the equivariance properties of the H-Net, and their use is how we are able to define the predictable transformation properties of feature maps under input rotations. In particular, rotation order $m=0$ defines local invariance because, denoting $f_m = [\mathbf{W}_m \star \mathbf{F}(r,\phi)]$, then $\psi^\theta_0[f_m] = e^{i\cdot 0\theta}\cdot f_m = f_m$, which is independent of θ, and $m=1$ defines local equivariance to rotation because $\psi^\theta_1[f_m] = e^{i\theta} f_m$. As the input rotates $e^{i\theta} f_m$ is a complex-valued number of constant magnitude f_m, spinning round with a phase equal to θ. Naturally, we are not constrained to using rotation orders 0 or 1 only, and we make use of higher and negative orders in our work. Note how we have been able to obtain these properties for any angle θ, with just a single filter, albeit with two components. On a practical note, it is worth mentioning, that complex cross-correlation can be implemented efficiently using 4 real cross-correlations

$$\underbrace{\mathbf{W}^{\text{Re}}_m \star \mathbf{F}^{\text{Re}} - \mathbf{W}^{\text{Im}}_m \star \mathbf{F}^{\text{Im}}}_{\text{real response}} + i\underbrace{(\mathbf{W}^{\text{Re}}_m \star \mathbf{F}^{\text{Im}} + \mathbf{W}^{\text{Im}}_m \star \mathbf{F}^{\text{Re}})}_{\text{imaginary response}}. \qquad (4)$$

So circular harmonics can be implemented in current deep learning frameworks, with minor engineering.

Arithmetic and the Equivariance Condition Further important properties of the circular harmonics, which are proven in the Supplementary Material, are: 1) Chained cross-correlation of rotation orders m_1 and m_2 lead to a new response with rotation order $m_1 + m_2$. 2) Point-wise nonlinearities $h:\mathbb{C}\to\mathbb{C}$, acting solely on the magnitudes maintain rotational equivariance, so we can interleave cross-correlations with typical CNN nonlinearities adapted to the complex domain. 3) The summation of two responses of the same order m remains of order m. Thus to construct a CNN where the output is M-equivariant to the input rotation, we require that the sum of rotation orders along any path equals M, so

$$\sum_{i=1}^{N} m_i = M. \qquad (5)$$

This is the fundamental condition underpinning the equivariance properties of H-Net, so we call it the *equivariance condition*.

We note here that for our purposes, our filter $\mathbf{W}_{-m} = \overline{\mathbf{W}_m}$ (the complex conjugate), which saves on parameters, but this does not necessarily imply conjugacy of the responses unless $\mathbf{F}$ is real, which is only true at the input.

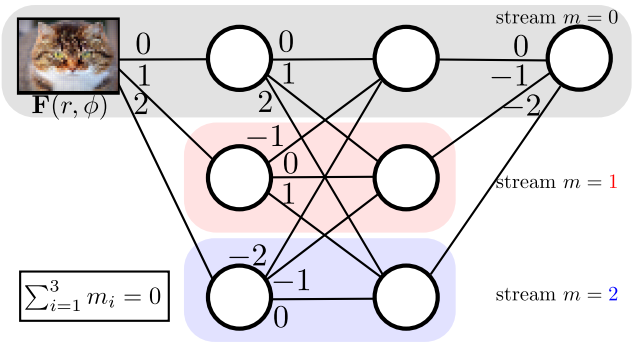

Figure 4. An example of a 2 hidden layer H-Net with $m=0$ output, input–output left-to-right. Each horizontal stream represents a series of feature maps (circles) of constant rotation order. The edges represent cross-correlations and are numbered with the rotation order of the corresponding filter. The sum of rotation orders along any path of consecutive edges through the network must equal $M=0$, to maintain disentanglement of rotation orders.

4. Method

We have considered the 360°-rotational equivariance of feature maps arising from cross-correlation with the circular harmonics, and we determined that the rotation orders of chained cross-correlations sum. Next, we use these results to construct a deep architecture, which can leverage the equivariance properties of circular harmonics.

4.1. Harmonic Networks

The rotation order of feature maps and filters sum upon cross-correlation, so to achieve a given rotation order at the output, we have to obey the equivariance condition. In fact, at every feature map in the network, the equivariance condition must be met, otherwise, it should be possible to arrive at the same feature map along two different paths, whose summed rotation orders are different. The problem with this is that we are combining complex features, with phases, which rotate at different frequencies, and so we say that the responses become *entangled*. The resultant

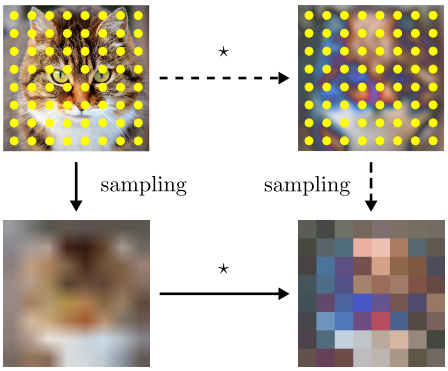

Figure 5. H-Nets operate in a continuous spatial domain, but we can implement them on pixel-domain data because sampling and cross-correlation commute. The schematic shows an example of a layer of an H-Net (magnitudes only). The solid arrows follow the path of the implementation, while the dashed arrows follow the possible alternative, which is easier to analyze, but computationally infeasible. The introduction of the sampling defines *centers of equivariance* at pixel centers (yellow dots), about which a feature map is rotationally equivariant.

feature map is no longer equivariant to a single rotation order, which makes it very difficult to work with. We resolve this by enforcing the equivariance condition at every feature map.

Our solution is to create separate streams of constant rotation order running through the network—see Figure 4. These streams contain multiple layers of feature maps, separated by rotation order zero cross-correlations and nonlinearities. Moving between streams, we use cross-correlations of rotation order equal to the difference between those two streams. It is very easy to check that the equivariance condition holds in these networks.

When multiple responses converge at a feature map, we have multiple choices of how to combine them. We could stack them, we could pool across them, or we could sum them [3]. To save on memory, we chose to sum responses of the same rotation order

$$\mathbf{Y}_p = \sum_{m,n:m+n=p} \mathbf{W}_m \star \mathbf{F}_n. \tag{6}$$

$\mathbf{Y}_p$ is then fed into the next layer. Usually in our experiments, we use streams of orders 0 and 1, which we found to work well and is justified by the fact that CNN filters tend to contain very little high frequency information [8].

Above, we see that the structure of the Harmonic Network is very simple. We replaced regular CNN filters with radially reweighted and phase shifted circular harmonics. This causes each filter response to be equivariant to input rotations with order m. To prevent responses of different rotation order from entangling upon summation, we separated filter responses into streams of equal rotation order.

Complex nonlinearities Between cross-correlations, we use complex nonlinearities, which act on the magnitudes of the

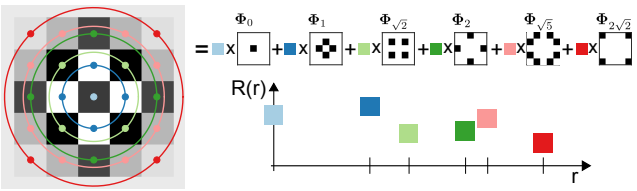

Figure 6. We map the radial profile onto the weights, as a linear combination of masking matrices $\mathbf{\Phi}_{r_i}$ and a radial profile tap.

complex feature maps only, to preserve rotational equivariance. An example is a complex version of the ReLU

$$\mathbb{C}\text{-ReLU}_b(Xe^{i\phi}) = \text{ReLU}(X+b)e^{i\phi}. \tag{7}$$

We can provide similar analogues for other nonlinearities and for Batch Normalization [7], which we use in our experiments.

We have thus far presented the Harmonic Network. Each layer is a collection of feature maps of different rotation orders, which transform predictably under rotation of the input to the network and the 360°-rotation equivariance is achieved with finite computation. Next we show how to implement this in practice.

4.2. Implementation: Discrete cross-correlations

Until now, we have operated on a domain with continuous spatial dimensions $\Omega = \mathbb{R} \times \mathbb{R} \times \{1, k_\ell\}$. However, the H-Net needs to operate on real-world images, which are sampled on a 2D-grid. Conveniently we can use the regular CNN architecture without any problems. This works on the fact that the order of sampling and cross-correlation is interchangeable [5]; they commute, so either we correlate in continuous space, then downsample, or downsample then correlate in the discrete space. Only the latter option is computationally tractable. Furthermore, since point-wise nonlinearities and sampling also commute, the entire H-Net, seen as a deep feature-mapping, commutes with sampling. This could allow us to implement the H-Net on non-regular grids; although, we did not explore this.

Viewing cross-correlation on discrete domains sheds some insight into how the equivariance properties behave. In Figure 5, we see that the sampling strategy introduces multiple origins, one for each feature map patch. We call these, *centers of equivariance*, because a feature map will exhibit local rotation equivariance about each of these points. If we move to using more exotic sampling strategies, such as strided cross-correlation or average pooling, then the centers of equivariance are ablated or shifted. If we were to use max-pooling, then the center of equivariance would be a complicated nonlinear function of the input image and harmonic weights. For this reason we have not used max-pooling in our experiments.

Complex cross-correlations We implement a grid-sampled version of the filters $\mathbf{W}_m$, using a sampling matrix $\mathbf{\Phi}$. The elements of $\mathbf{\Phi}$ denote the angle of the corresponding elements with respect to a chosen origin, which we define to be the center

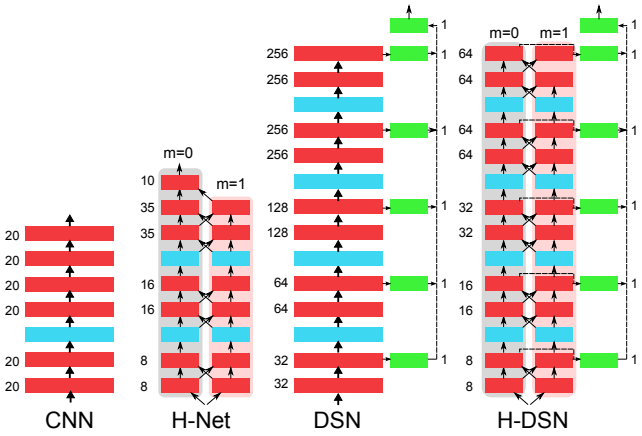

Figure 7. Networks used in our experiments. LEFT: MNIST networks, as per [2]. RIGHT deeply-supervised networks (DSN) [15] for boundary segmentation, as per [26]. Red boxes denote feature maps. Blue boxes are pooling (max for CNNs and average for H-Nets). Green boxes are side feature maps as per [26]; these are connected to the DSN with dashed lines for ease of viewing. All main cross-correlations are 3×3, unless otherwise stated in the experiments section.

of the filter. To map radial profile weights $\mathbf{R}(r_i)$ at radii r_i to the filter, we separate out each ring of elements equidistant from the origin in $\mathbf{\Phi}$, denoting each matrix as $\mathbf{\Phi}_{r_i}$ (see Figure 6), and weight each ring by a radial profile element. The phase offset β can be implemented by noting that

$$\sum_{i=1}^{I}\mathbf{R}(r_i)e^{i(m\mathbf{\Phi}_{r_i}+\beta)}=\sum_{i=1}^{I}\mathbf{R}(r_i)\begin{bmatrix}\mathbf{I}\cos\beta & -\mathbf{I}\sin\beta\\ \mathbf{I}\sin\beta & \mathbf{I}\cos\beta\end{bmatrix}\begin{bmatrix}\cos m\mathbf{\Phi}_{r_i}\\ i\sin m\mathbf{\Phi}_{r_i}\end{bmatrix} \tag{8}$$

where the complex exponential and trigonometric terms are element-wise, and $\mathbf{I}$ is the identity matrix. This is just a reweighting of the ring elements. In full generality, we could also use a per-radius phase β_{r_i}, which would allow for spiralic left- and right-handed features, but we did not investigate this.

4.3. Computational complexity

We have increased the computational complexity of cross-correlation, but also increased the representative power of the network. Here we analyze the computational complexity in terms of number of multiplications. In the standard cross-correlation, for an input of size $h\cdot w\cdot i_{\mathbb{Z}}$, (height, width, input channels) and filters of size $k\cdot k\cdot o_{\mathbb{Z}}$ (height, width, output channels), the number of multiplications to form a feature map of the same size as the input is $M(\mathbb{Z})=hwk^2i_{\mathbb{Z}}o_{\mathbb{Z}}$. In the H-Net, we have f rotation orders on the input and r rotation orders on the output, so perform fr complex cross-correlations. Each complex cross-correlation can be formed from 4 real cross-correlations, so the number of multiplications is $4M(\mathbb{H})fr$, where $i_{\mathbb{H}}$ and $o_{\mathbb{H}}$ are the number of input and output channels, respectively. Thus for similar levels of computation we equate the two to yield $M(\mathbb{Z})=4M(\mathbb{H})fr$. Rearranging; setting $i_{\mathbb{H}}=o_{\mathbb{H}}$, $i_{\mathbb{Z}}=o_{\mathbb{Z}}$ and $f=r$; and taking the square root of both sides, we arrive at a simple rule of thumb for network design,

$$i_{\mathbb{Z}}=2fi_{\mathbb{H}}. \tag{9}$$

For example, if we want to build an H-Net with similar computation to a regular CNN with 64 channels per layer, then if we use 2 rotation orders $m\in\{0,1\}$, then the number of H-Net channels is $64/(2\cdot2)=16$.

Method	Test error (%)	# params
SVM [13]	11.11	-
Transformation RBM [25]	4.2	-
Conv-RBM [22]	3.98	-
CNN [2]	5.03	22k
CNN [2] + data aug*	3.50	22k
$P4$CNN rotation pooling [2]	3.21	25k
$P4$CNN [2]	2.28	25k
H-Net (Ours)	**1.69**	33k

Table 1. Results. Our model sets a new state-of-the-art on the rotated MNIST dataset, reducing the test error by 26%. * Our reimplementation

5. Experiments

We validate our rotation equivariant formulation below, performing some introspective investigations, and measuring against relevant baselines for classification on the rotated-MNIST dataset [13] and boundary detection on the Berkeley Segmentation Dataset [1]. We selected our baselines as representative examples of the current state-of-the-art and to demonstrate that H-Nets can be used on different architectures for different tasks. The networks we used are in Figure 7.

5.1. Benchmarks

Here we compare H-Nets for classification and boundary detection. Classification is a typical rotation invariant task, and should suit H-Nets very well. In contrast, boundary detection is a rotation equivariant task. The key to the success of the H-Net is that it can achieve global equivariance, without sacrificing local equivariance of features.

MNIST Of course, this is a small dataset, with simple visual structures, but it is a good indication of how introducing the right equivariances into CNNs can aid inference. We investigate classification on the rotated MNIST dataset (new version) [13] as a baseline. This has 10000 training images, 2000 validation images, and 50000 test images. The 360°-rotations and small training set size make this a difficult task for classical CNNs. We compare against a collection of previous state-of-the-art papers and [2], who build a deep CNN with filter copies at 90°-rotations. We try to mimic their network architecture for H-Nets as best as we can, using 2 rotation order streams with

Method	ODS	OIS	# params
Xie et al., [26]*	0.64	0.65	2346k
Kivinen et al., [10]	0.702	0.715	-
H-Net (Ours)	**0.726**	**0.742**	116k

Table 2. Our model beats the baselines when testing on the Berkeley Segmentation Dataset (BSD500) [1]. No pretraining was used for any models. *Our implementation.

$m \in \{0,1\}$ through to the deepest layer, and complex-valued versions of ReLU nonlinearities and Batch Normalization (see Method). We also replace max-pooling with mean-pooling layers, as shown in Figure 7. We perform stochastic gradient descent on a cross-entropy loss using Adam [9] and an adaptive learning rate, which we divide by 10 if there has been no improvement in validation accuracy in the last 10 epochs. We train multiple models with randomly chosen hyperparameters, and report the test error of the model that performed best on the validation set, training on a combined training and validation set Table 1 lists our results. This model actually has 33k params, which is about 50% larger than the standard CNN and [2], which have 22k. This is because it uses 5×5 convolutions instead of 3×3. Interestingly, it does not overfit on such a small dataset and it still outperforms the standard CNN trained with rotation augmentations, which we do not use. We set the new state-of-the-art, with a 26% improvement on the previous best model.

Deep Boundary Detection Boundary detection is equivariant to non-rigid transformations; although edge presence is locally invariant to orientation. The current state-of-the-art depends on finetuning ImageNet-pretrained networks to regress boundary probabilities on a per-patch basis. To demonstrate that hard-baked rotation equivariance serves as a strong generalization tool, we compared against a previous state-of-the-art architecture [26], *without pretraining*. We tried to mimic [26] as closely as possible, with differences highlighted in Figure 7. The main difference is that we divide the number of all feature maps by 2, for faster, more stable training. They use a VGG network [23] extended with deeply supervised network (DSN) [15] *side-connections*. These are 1×1-convolutions, which perform weighted averages across all relevant feature maps, resized to match the input. A binary cross-entropy loss is applied to each side connection, to stabilize learning. A final 'fusion' layer is created by taking a weighted linear combination of the side-connections, this is the final output. We adapt side-connections to H-Nets, by using the complex magnitude of feature maps before taking a weighted average. This means that the resultant boundary predictions are locally invariant to rotation. We added a small sparsity regularizer to our cost function, because we found it improved results slightly. We call the Harmonic variant of the DSN, an *H-DSN*.

We also compared with [10], who use a mean-and-covariance-RBM. There technique has five main contributions:

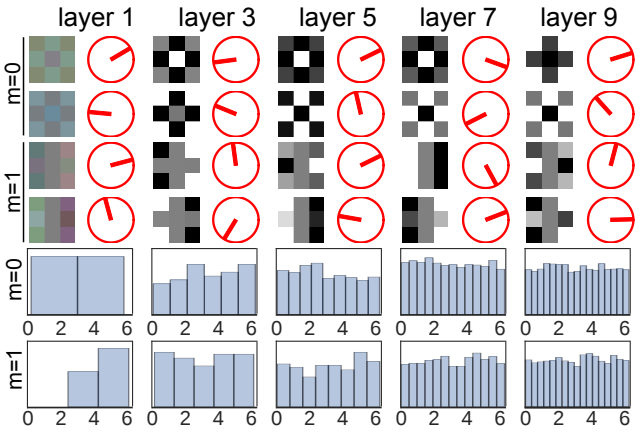

Figure 8. Randomly selected filters and phase histograms from the BSDS500 trained H-DSN. Filter are aligned at $\beta=0$; and the oriented circles represent phase. We see few filter copies and no blank filters, as usually seen in CNNs. We also see a balanced distribution over phases, indicating that boundaries, and their deep feature representations, are uniformly distributed in orientation.

1) zero-mean, unit variance normalization of inputs, 2) sparsity regularization of hidden units, 3) averaged ground truth edge annotations, 4) average outputs to 16 input rotations, 5) non-maximum suppression of results by the Canny method. We perform the first 2 methods, but leave the last 3 alone. In particular, they do not pretrain on ImageNet, and attempt some kind of rotation averaging for global equivariance, so are a good baseline to measure against. We tested on the Berkeley Segmentation Dataset (BSD500) [1]. As shown in Table 2 for non-pretrained models, H-Nets deliver superior performance over current state-of-the-art architectures, including [10], who also encode rotation equivariance. Most noticeable of all is that we only use 5% of the params of [26], showing how by restricting the search space of learnable models through hard-baking local rotation equivariance, we need not learn so many parameters.

5.2. Model Insight

Here we investigate some of the properties of the H-Net implementation, making sure that the motivations behind H-Net design are achieved by the implementation.

Filter Visualization The real parts of the filters, from the first layer of the boundary-detection-trained H-Net, are shown in Figure 8. They are aligned at zero phase ($\beta = 0$) for ease of viewing. Since the network is trained on zero-mean, unit variance, normalized color images, the weights do not have the natural colors we would see in real-world images. Nonetheless, there is useful information we can glean from inspecting these. Most 1st layer filters detect color boundaries, there are no blank filters as one usually sees in CNNs, and there are few reoriented copies. We also see from the phase histograms that all phases are utilized by filters throughout the network, indicating full use of the phase information. This is interesting, because it means

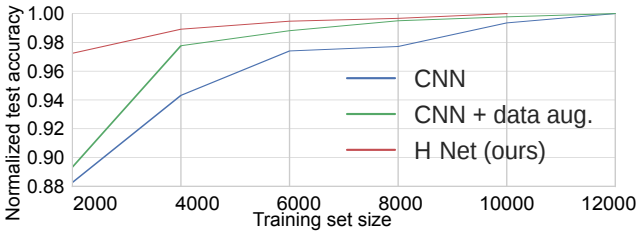

Figure 9. Data ablation study. On the rotated MNIST dataset, we experiment with test accuracy for varying sizes of the training set. We normalize the maximum test accuracy of each method to 1, for direct comparison of the falloff with training size. Clearly H-Nets are more data-efficient than regular CNNs, which need more data to discover rotational equivariance unaided.

that the model's parameters are being used fully, with low redundancy, which we surmise comes from easier optimization on the equivariant manifold.

Data ablation H-Nets are parameter-efficient, but here we investigate whether they are data-efficient. It is no secret that CNNs are massively data-hungry. Krizhevsky's landmark paper [11] in 2012 used 60 million parameters, trained on 1.2 million 256×256 RGB images quantized to 256 bits and split between 1000 classes, for a total of 10 bits of information per weight. Even this vast amount of data was insufficient for training, and aggressive data augmentation was needed to improve results. We ran an experiment on the rotated MNIST dataset to show that since we need not learn rotation equivariance, we require less data than competing methods, which is indeed the case (see Figure 9). Interestingly, and predictably, regular CNNs trained with data augmentation still perform worse than H-Nets, because they only learn global invariance to rotation, rather than local equivariances at each layer.

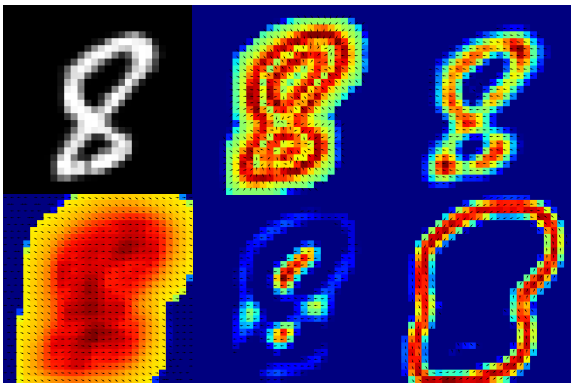

Figure 10. Feature maps from the MNIST network. The arrows display phase, and the colors display magnitude information (jet color scheme). There are diverse features encoding edges, corners, whole objects, negative spaces, and outlines.

Figure 11. View best in color. Orientated feature maps for the H-DSN. The color wheel shows orientation coding. Note that between layers boundary orientations are colored differently because each feature has a different β. This visualization demonstrates the consistency of orientation within a feature map and across multiple layers. The images are taken from layers 2, 4, 6, 8, and 10 in a clockwise order from largest to smallest.

Feature maps We visualize feature maps in the lower layers of an MNIST trained H-Net (see Figure 10). For given input, we see the feature maps encode very complicated structures. Left to right, we see the H-Net learns to detect edges, corners, object presence, negative space, and outlines of objects. We perform this for the BSD500 trained H-DSN (see Figure 11). It showsequivariance is preserved through to the deepest feature maps. It also highlights the compact representation of feature presence and pose, which regular CNNs cannot do.

6. Conclusions

We presented a convolutional neural network that is locally equivariant to patch-wise translation and, for the first time, to continuous 360°-rotation. We achieved this by restricting the filters to circular harmonics, essentially hard-baking rotation into the architecture. This can be implanted onto other architectures too. The use of circular harmonics pays dividends in that we receive higher representational power using fewer parameters. This leads to good generalization, even with less (or less augmented) training data. The only disadvantage we've seen so far is the higher per-filter computational cost, but our guidance for network design balances that cost against the more expressive representation. The better interpretability of the feature maps is a bonus, because we know how they transform under input image rotations. We applied our network to the problem of classifying rotated-MNIST, setting a new state-of-the-art. We also applied our network to boundary detection, again achieving state-of-the-art results, for non-pretrained networks. We have shown that 360°-rotational equivariance is both possible and useful.

Future work Extension of this work could involve hard-baking yet more transformations into the equivariance properties of the Harmonic Network, possibly extending to 3D. This will allow yet more expressibility in network representations, extending the benefits we have seen afforded by rotation equivariance to a larger class of models and applications.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(5):898–916, May 2011. 6, 7

[2] T. S. Cohen and M. Welling. Group equivariant convolutional networks. *arXiv preprint arXiv:1602.07576*, 2016. 2, 6, 7

[3] S. Dieleman, J. De Fauw, and K. Kavukcuoglu. Exploiting cyclic symmetry in convolutional neural networks. *arXiv preprint arXiv:1602.02660*, 2016. 2, 5

[4] B. Fasel and D. Gatica-Perez. Rotation-invariant neoperceptron. In *18th International Conference on Pattern Recognition (ICPR 2006), 20-24 August 2006, Hong Kong, China* [4], pages 336–339. 2, 9

[5] W. T. Freeman and E. H. Adelson. The design and use of steerable filters. *IEEE Transactions on Pattern analysis and machine intelligence*, 13(9):891–906, 1991. 2, 4, 5

[6] G. E. Hinton, A. Krizhevsky, and S. D. Wang. Transforming auto-encoders. In *Artificial Neural Networks and Machine Learning - ICANN 2011 - 21st International Conference on Artificial Neural Networks, Espoo, Finland, June 14-17, 2011, Proceedings, Part I* [6], pages 44–51. 2, 9

[7] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015* [7], pages 448–456. 5, 9

[8] J. Jacobsen, J. C. van Gemert, Z. Lou, and A. W. M. Smeulders. Structured receptive fields in cnns. *CoRR*, abs/1605.02971, 2016. 5

[9] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 7

[10] J. J. Kivinen, C. K. I. Williams, and N. Heess. Visual boundary prediction: A deep neural prediction network and quality dissection. In *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, AISTATS 2014, Reykjavik, Iceland, April 22-25, 2014* [10], pages 512–521. 7, 9

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States.* [11], pages 1106–1114. 2, 8, 9

[12] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys. TI-POOLING: transformation-invariant pooling for feature learning in convolutional neural networks. *CoRR*, abs/1604.06318, 2016. 2

[13] H. Larochelle, D. Erhan, A. C. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007* [13], pages 473–480. 6, 9

[14] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems 2, [NIPS Conference, Denver, Colorado, USA, November 27-30, 1989]* [14], pages 396–404. 1, 9

[15] C. Lee, S. Xie, P. W. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2015, San Diego, California, USA, May 9-12, 2015* [15]. 6, 7, 9

[16] K. Lenc and A. Vedaldi. Understanding image representations by measuring their equivariance and equivalence. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015* [16], pages 991–999. 1, 3, 9

[17] K. Lenc and A. Vedaldi. Learning covariant feature detectors. *CoRR*, abs/1605.01224, 2016. 2

[18] K. Liu, Q. Wang, W. Driever, and O. Ronneberger. 2d/3d rotation-invariant detection using equivariant filters and kernel weighted mapping. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012* [18], pages 917–924. 2, 4, 9

[19] D. Marcos, M. Volpi, and D. Tuia. Learning rotation invariant convolutional filters for texture classification. *arXiv preprint arXiv:1604.06720*, 2016. 2

[20] R. Memisevic and G. E. Hinton. Learning to represent spatial transformations with factored higher-order boltzmann machines. *Neural Computation*, 22(6):1473–1492, 2010. 2

[21] E. Oyallon and S. Mallat. Deep roto-translation scattering for object classification. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015* [21], pages 2865–2873. 2, 9

[22] U. Schmidt and S. Roth. Learning rotation-aware features: From invariant priors to equivariant descriptors. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pages 2050–2057, 2012. 6

[23] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 7

[24] S. Soatto. Actionable information in vision. In *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009* [24], pages 2138–2145. 1, 9

[25] K. Sohn and H. Lee. Learning invariant representations with local transformations. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012* [25]. 6, 9

[26] S. Xie and Z. Tu. Holistically-nested edge detection. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015* [26], pages 1395–1403. 6, 7, 9

[27] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I* [27], pages 818–833. 2, 9