Soft Rotation Equivariant Convolutional Neural Networks

Eduardo Castro
University of Porto
INESC TEC
Porto, Portugal
emcastro@inesctec.pt

José Costa Pereira
Noah Ark Lab, Huawei
INESC TEC
London, UK
jose.c.pereira@inesctec.pt

Jaime S. Cardoso
University of Porto
INESC TEC
Porto, Portugal
jaime.cardoso@inesctec.pt

***Abstract*—A key to the generalization ability of Convolutional Neural Networks (CNNs) is the idea that patterns that appear in one region of the image have a high probability of appearing in other regions. This notion is also true for other spatial relationships, such as orientation. Motivated by the fact that in the early layers of CNNs distinct filters often encode for the same feature at different angles, we propose to incorporate the rotation equivariant prior in these models. In this work, different regularization strategies that capture the notion of *approximate equivariance* were designed and quantitatively evaluated in their ability to generate rotation-equivariant models and their effect on the model's capacity to generalize to unseen data. Some of these strategies consistently lead to higher test set accuracies when compared to a baseline model, on classification tasks. We conclude that the rotation equivariance prior should be adopted in the general setting when modeling visual data.**

***Index Terms*—deep learning, convolutional neural networks, rotation equivariance**

I. Introduction

The development and popularization of deep learning have played a central role in many of the recent breakthroughs in computer vision research. Neural networks (NN) are now frequent in state-of-the-art systems. In contrast with the previous approaches based on handcrafted feature extractors, these models enable data-driven feature learning. In this interpretation, hidden layers are sequentially applied to the input for feature extraction, while the output layer acts as a classifier or regressor. Thus, one of the challenges of deep learning is how to make the feature learning process more robust.

From the lense of the machine learning field, one of the challenges of image data is that individual pixels are usually not very informative on their own. However, their combination forms intricate patterns that are highly relevant to different visual tasks. Deep neural networks are often used to model these patterns. When this is the case, the features encoded in the neurons' activations become progressively higher-level in the sense that they are more informative towards the ultimate goal of the model.

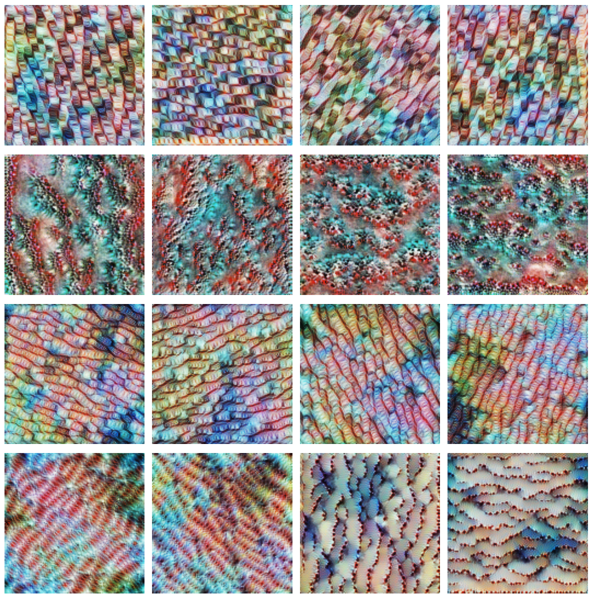
This work was partially financed by the ERDF – European Regional Development Fund – through the Operational Programme for Competitiveness and Internationalisation – COMPETE 2020 Programme – and by National Funds through the Portuguese funding agency FCT – Fundação para a Ciência e a Tecnologia – within the project "POCI-01-0145-FEDER-028857" and the PhD grant "*SFRH/BD/136274/2018*". The authors would also like to aknowledge NVIDIA for their generous donation of a TitanX gpu.

Convolutional neural networks (CNNs) are the most common type of NN in vision applications. Essentially, two properties make them better suited for modeling natural images, when compared to the multilayer perceptron model (MLP). While in MLPs, the activation of a neuron depends on all the responses of the previous layer, for CNNs, this activation depends only on a small local region of the input (local connectivity). Also, the weights of CNNs are shared across spatial dimensions in the same layer (weight sharing). From a probabilistic perspective, we can understand CNNs as MLPs with a very strong prior on the weights' distribution, which assigns zero probability to values that do not respect either one of the conditions of local connectivity or weight sharing. In practice, this means that feature extraction is the same regardless of the image region that is being processed. Equivalently, we can say that feature extraction is equivariant under translations.

Intuitively, this seems a good prior as patterns that emerge in one image region are probably useful in other regions as well. This intuition is validated by empirical results, as CNNs generalize better than MLPs in the majority of visual tasks. From statistical learning theory, this prior is seen as a restriction of the hypothesis space. The fact that it leads to better solutions reflects the fact that the functions kept in the hypothesis space are generally more useful for visual tasks.

The importance of translation equivariant feature extractors in NNs motivates the search for other types of equivariances of potential interest to the majority of visual tasks. In this context, plane rotations of the input are a natural candidate. It is well-known that in CNNs, the first layer weights often converge into filterbanks with repeated filters at multiple orientations. Feature extraction in multiple orientations is preserved for at least the next few layers, as shown in Fig. 1a. On the contrary, higher-level features are dominated by directional features that either appear in one single orientation or are rotation-invariant (see Fig. 1b).

In this work, we evaluate rotation equivariance as a prior for the weights of the early layers of a CNN model. Our work builds on previous literature on equivariant representations for convolutional models. We also propose soft rotation-equivariant convolutional networks, which are inspired by the concept of "approximately equivariant." Finally, we evaluate


(a) Feature visualization for the fourth convolutional layer.

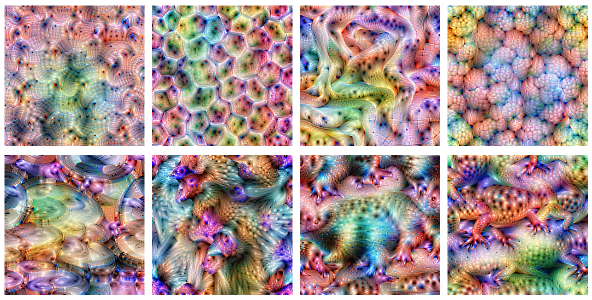

(b) Feature visualization for the last convolutional layer.

Fig. 1: Feature visualization for the VGG19 model trained on ImageNet. Each image corresponds to the input which maximizes the activations in a given channel (based on [1]). Many neurons on the fourth layer (a) encode the same feature in different orientations. On the contrary, the last layer's channels (b) encode for different features, some orientation invariant (top row) and others orientation specific (bottom row).

the effectiveness of different methods for the introduction of this prior in CNNs and its influence on the ability of the network to generalize to unseen data.

II. Related Work

Previous works in the field have addressed rotation equivariance and invariance in different ways. The most common technique to incorporate this prior knowledge in computer vision tasks is by doing rotation-based data augmentation [2], [3], in which data is rotated by a random angle before being fed to the training algorithm. This angle is usually drawn from a uniform distribution, whose interval is defined based on the specific problem at hand. This method can be seen as a prior on the data sampling scheme - e.g., the angle at which the image was captured was arbitrary. This technique allows an increase in the amount of training data, and thus, it usually leads to better generalization, most notably on rotation-invariant problems.

Another approach is to analyze the image at multiple orientations simultaneously. For this, rotated versions of the input are given to the model and processed in parallel. The intermediate representations are therefore available for different orientations and can be combined for decision. The works of [4] and [5] combine the outputs of these parallel streams at inference for increased robustness. Differently, [6] also employ this technique during training. A different approach also based on parallel processing is the rotation-invariant layer of [7]. This method penalizes in the loss function different intermediate representations for the same input at different orientations.

Some works address rotation equivariance and invariance using more sophisticated theoretical frameworks that make use of group theory to describe the symmetries expected in the data. This is the case of group equivariant convolutional networks [8], in which the authors define the group-equivariant convolution operation. This operation is more general and addresses other types of transformations, such as reflections of the plane. This work serves as a reference for the analysis done in our work. While the group-equivariant convolution works for 90° rotations, harmonic-networks [9] are equivariant under continuous groups of rotations. They replace the traditional filterbank parametrization in the network with circular harmonics, returning a maximal response and orientation. This notion of steerable filters, in which a finite set of coefficients can encode the response of the filter in all orientations, was also explored by [10], which proposed Steerable CNNs. Another interesting example of this is group-equivariant capsule networks [11]. While the original capsules aimed at viewpoint equivariance and invariance through the use of a pose vector, [11] make these pose vectors elements of a group, guarantying rotation equivariance.

Other works that are also relevant in this context provide insight into the equivariance properties in neural networks. This is the case of [12] that studies common CNN architectures and the equivalence and equivariance of their representations. Through their proposed method, they can detect at which point in the network a CNN loses its geometric symmetries. The work of [13] demonstrates the general link between convolution and equivariance under compact groups. In a similar fashion [14] describes the relationship between parameter sharing and equivariance. Finally, the work of [15] categorizes different previous works on this topic and provides a general theory for equivariant CNNs.

III. Rotation-Equivariant CNNs

A layer is said to be *equivariant* under a set of transformations if applying them to the input changes the output in a predictable way (Fig. 2). If the output is unchanged, the layer is said to be *invariant*, which is a special case of equivariance. More formally, considering the function f and the group G, we say that f is *equivariant* if the group actions T and T' exist such that, for any x and any element $g \in G$:

$$f(T_g.x) = T'_g.f(x) \tag{1}$$

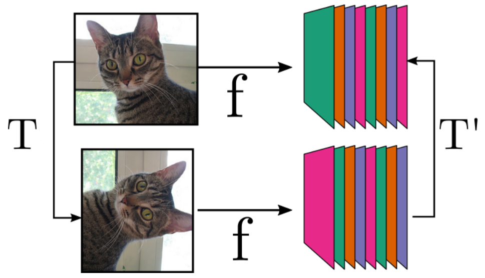


Fig. 2: Example of a rotation equivariant function. A rotation (T) on the input causes a shift (T') on the output.

Equivariance reflects the symmetries of function f. In Equation 1, T is a group action of the group G on set X, and T_g represents the transformation associated with element $g \in G$. Similarly, T' is a group action of the same group G on the codomain of f. Note, however, that T'_g need not be equal to T_g, and in general, they do not act on the same set. The only requirements for T to be a group action are: 1) that the transformation associated with the identity element maps to the same element $T_e.x = x$; and 2) that the composition of two transformations can be obtained by performing the group operation on the associated elements $T_g.T_h = T_{gh}$. From Equation 1 it is clear that invariance is a special case where $T'_g = T'_e \ \forall g \in G$. Notice, however, that invariance is not always a good prior as, in the case of rotation, it prevents the extraction of features in specific orientations, such as those in Fig. 1a and the bottom of Fig. 1b.

A. Plane Rotations

In this work, we considered the group composed of plane rotations multiple of $\frac{\pi}{2}$, $G = \{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\}$. Notice that the square grid is invariant to these rotations. As a consequence of this, we can define the group action h_θ on a point as that of first rotating the square grid by θ and then computing the coordinates of point x after the grid is rotated. Thus the result of applying h_θ to point $x = (u, v) \in \mathbb{Z}^2$, is given by:

$$h_\theta . \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} c_\theta & -s_\theta \\ s_\theta & c_\theta \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{cases} (u, v) & if \quad \theta = 0 \\ (-v, u) & if \quad \theta = \frac{\pi}{2} \\ (-u, -v) & if \quad \theta = \pi \\ (v, -u) & if \quad \theta = \frac{3\pi}{2} \end{cases} \tag{2}$$

with c_θ, s_θ the cosine and sine functions evaluated at θ. Notice that G is cyclic as it can be generated using the element $\frac{\pi}{2}$. The group operation of G is the sum of the rotation angles:

$$h_{\theta_1} + h_{\theta_2} = h_{(\theta_1 + \theta_2 \mod 2\pi)} \tag{3}$$

To see that h is indeed a group action notice that: 1) using Equation 2, $h_0.x = x$; 2) the composition of two transformations is given by the sum of the two angles.

$$h_{\theta_1}.h_{\theta_2}.x = \begin{bmatrix} c_{\theta_1} & -s_{\theta_1} \\ s_{\theta_1} & c_{\theta_1} \end{bmatrix} \begin{bmatrix} c_{\theta_2} & -s_{\theta_2} \\ s_{\theta_2} & c_{\theta_2} \end{bmatrix} = \tag{4}$$

$$\begin{bmatrix} c_{\theta_1+\theta_2} & -s_{\theta_1+\theta_2} \\ s_{\theta_1+\theta_2} & c_{\theta_1+\theta_2} \end{bmatrix} = h_{\theta_1+\theta_2}.x \tag{5}$$

B. The Convolutional Layer

In this work we will model layer representations (including the input) as $f : \mathbb{Z}^2 \to \mathbb{R}^C$ and each weight tensor, w, as a stack of K filters with the form $w_i : \mathbb{Z}^2 \to \mathbb{R}^C$ for $i \in \{0, ..., K-1\}$. In this formulation, C is the number of channels in representation f while K is the number of output channels in the layer parameterized by w. The convolution operation between the input image and one filter w_i is denoted by $*$ and given by:

$$[f * w_i](x) = \sum_{c=0}^{C-1} \sum_{y \in \mathbb{Z}^2} f_c(y) w_{i,c}(y - x) \tag{6}$$

If we want to refer to all output channels of that layer we will use the same symbol $f * w$, which can then be indexed to specify one specific channel ($[f * w]_i = f * w_i$).

Considering f^0 as the input of the network, we can define the group action T_θ which rotates the image: $[T_\theta.f^0](x) = f^0(h_{-\theta}.x)$. It follows from the properties of the transformation h and the fact that f^0 is a function, that this is indeed a group action of group G on the codomain of f^0. Similarly, this operation is also defined for w_i: $[T_\theta.w_i](x) = w_i(h_{-\theta}.x)$, which means a filter rotation.

C. Rotation-Equivariant Convolution

We now show how a CNN may be parameterized in order to preserve rotation-equivariance throughout its intermediate representations. Our first observation is that the convolution of the transformed image $T_\theta.f^0$ with filter w_k can alternatively be obtained by rotating the filter and the resulting feature map:

$$[[T_\theta.f^0] * w_i](x) = \sum_{c=0}^{C-1} \sum_{y \in \mathbb{Z}^2} f^0_c(h_{-\theta}.y) w_{i,c}(y - x) \tag{7}$$

$$= \sum_{c=0}^{C-1} \sum_{z \in \mathbb{Z}^2} f^0_c(z) w_{i,c}((h_\theta.z) - x) \tag{8}$$

$$= \sum_{c=0}^{C-1} \sum_{z \in \mathbb{Z}^2} f^0_c(z) w_{i,c}(h_\theta.(z - h_{-\theta}.x)) \tag{9}$$

$$= [f^0 * [T_{-\theta}.w_i]](h_{-\theta}.x) \tag{10}$$

$$= [T_\theta.[f^0 * [T_{-\theta}.w_i]]](x) \tag{11}$$

In step 9 we assumed the group action was distributive over the sum, which is true given that the action can be computed with matrix multiplication (Equation 2).

Based on this result, we can parameterize the first convolutional layer so that it produces an equivariant response. For this, we define each filter of the new weight tensor, w', to be:

$$w'_{4i+k} = T_{-\frac{k\pi}{2}}.w_i \tag{12}$$

with $k \in \{0,1,2,3\}$. This definition means that the same filter is included four times, one for each angle multiple of $\frac{\pi}{2}$. As such, the number of output channels increases by four if the same number of parameters is kept. Conversely, if one wishes to keep the number of convolutions the same, the number of base filters (w_i) should be reduced four-fold. It also means that each rotation on w' causes a circular shift between the filters that correspond to the same base filter:

$$T_\theta.w'_{4i+k} = T_\theta.T_{-\frac{k\pi}{2}}.w_i = T_{\frac{(b-k)\pi}{2}}.w_i = w'_{4i+((k-b) \mod 4)} \tag{13}$$

with $b = \frac{2\theta}{\pi}$ and $\mod$ the modulo operator. The last step is based on fact that $\frac{k\pi}{2} = \frac{(k \mod 4)\pi}{2}$.

We now show that this parametrization guarantees an equivariant response for the first layer of the model. Based on the previous result (Equation 11), the result of the convolution of $T_\theta.f^0$ with the $(4i+k)th$ filter in w' is given by:

$$[[T_\theta.f^0] * w']_{4i+k} = [T_\theta.f^0] * [T_{-\frac{k\pi}{2}}.w_i] \tag{14}$$
$$= T_\theta.[f^0 * [T_{-\theta}.[T_{-\frac{k\pi}{2}}.w_i]]] \tag{15}$$
$$= T_\theta.[f^0 * [T_{-\frac{(k+b)\pi}{2}}.w_i]] \tag{16}$$
$$= T_\theta.[f^0 * w']_{4i+((k+b) \mod 4)} \tag{17}$$

Notice that a rotation of the input causes a rotation of the output along with a circular shift. The combination of these two transformations is a group action of group G on the codomain of $[f^0 * w']$. To see this, notice that there are four possible circular shifts, which can be accessed by successively applying the first element ($\frac{\pi}{2}$ or $b=1$) zero to three times, just like rotation. We will call this new group action L_θ and so:

$$[L_\theta.f^1]_{4i+k} = L_\theta.[f^0 * w']_{4i+k} = T_\theta.[f^0 * w']_{4i+((k+b) \mod 4)} \tag{18}$$

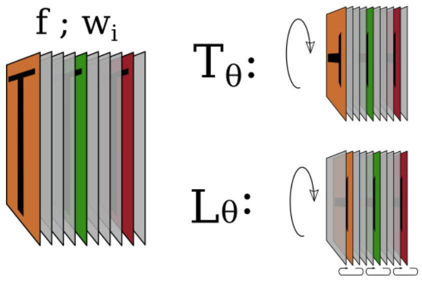


Fig. 3: Visualization of the group actions defined in section III.

This parametrization for w' ensures that there is a correspondence between each input rotation, T_θ, and output transformation, L_θ. Because layers are applied sequentially, the next layer should preserve equivariance for an input acted upon by L_θ (rather than T_θ). Only then is equivariance preserved across the composition of the two layers, a fact that follows from the definition in Equation 1. A parametrization that ensures this is given by:

$$w'_{4i+k} = L_{-\frac{k\pi}{2}}.w_i \tag{19}$$

Note that L_θ acts on the input-channels dimension of the tensor:

$$L_\theta.w_{i,c}(x) = w_{i,c'}(h_{-\theta}.x) \tag{20}$$

with $c' = (c//4) \times 4 + ((c-b) \mod 4)$, with $//$ denoting the integer division operation.

An analogous result to Equation 11, showing the equivalence between performing the operation on the feature representation or on the filters, can be obtained for L_θ:

$$[[L_\theta.f^1] * w_i](x) = \sum_{c=0}^{C-1} \sum_{y \in \mathbb{Z}^2} f^1_{c'}(h_{-\theta}.y) w_{i,c}(y-x) \tag{21}$$
$$= \sum_{c'=0}^{C-1} \sum_{z \in \mathbb{Z}^2} f^1_{c'}(z) w_{i,c}(h_\theta.(z - h_{-\theta}.x)) \tag{22}$$
$$= [f^1 * [L_{-\theta}.w_i]](h_{-\theta}x) \tag{23}$$
$$= [T_\theta.[f^1 * [L_{-\theta}.w_i]]](x) \tag{24}$$

in the above result we considered $c' = (c//4) \times 4 + ((c-b) \mod 4)$ which means $c = (c'//4) \times 4 + ((c'+b) \mod 4)$, the shift corresponding to $L_{-\theta}$. Using this, we can show that the parametrization shown in Equation 19 indeed guaranties equivariance:

$$[[L_\theta.f^1] * w']_{4i+k} = T_\theta.[f^1 * [L_{-\theta}.w'_{4i+k}]] \tag{25}$$
$$= T_\theta.[f^1 * w'_{4i+((k+b) \mod 4)}] \tag{26}$$
$$= T_\theta.[f^1 * w']_{4i+((k+b) \mod 4)} \tag{27}$$
$$= L_\theta.[f^1 * w']_{4i+k} \tag{28}$$

In this case there is a correspondence between the input transformation L_θ and the output transformation L_θ. Thus, using the same parametrization for the following layers is enough to preserve equivariance across the composition of layers.

IV. Rotation Equivariance Regularization

The introduction of the rotation equivariance prior in early layers can make learning more robust, and thus act as a regularization method. Following the analysis of the previous section, one way to do this is by hard constraining the weights in the convolutional layers. However, it is not clear that this is the best method. For instance, in the work of [12], authors show how the internal representations become progressively less equivariant to rotation as we move from the input to the output of the model. Thus, a fixed structure on the convolutional filters might not be optimal, as it does not allow for this progressive loss of equivariance.

In this context, we considered three families of methods to incorporate the rotation equivariance prior. The first one uses a fixed structure on the weights, as discussed in the previous section. The remaining two consist of additive terms which when added to the loss function promote (but do not enforce) equivariance and thus must be balanced with other minimization objectives. Because they capture this notion of *approximately equivariant*, we name the models regularized by these strategies as *Soft Rotation-Equivariant CNNs*. Each method was given a short name to facilitate the comprehension of the experimental section.

A. Constraining the Weights (hard)

The equivariance prior can be implemented directly based on the previous analysis by setting the filters of the first and subsequent layers according to the formulas:

$$w'_{4i+k} = T_{\frac{k\pi}{2}}.w_i \tag{29}$$

$$w'_{4i+k} = L_{\frac{k\pi}{2}}.w_i \tag{30}$$

This formulation is a particular case of the group equivariant convolution proposed by [8]. While the authors validate their method as a new convolution operation, in our work, we use it as a prior on the weights of particular layers. This allows us to understand the importance of rotation equivariance at different points in the network. Also, differently from [8], in our experimental setting we compare models with the same time complexity (number of operations), instead of parameter complexity (number of parameters). Both comparisons are valid, and its usefulness depends on the considered setting.

B. Soft priors on the Weights

An alternative strategy to achieve rotation equivariance is to penalize parametrizations that do not guarantee rotation equivariance in the loss function. Thus, this strategy takes the form of:

$$\mathcal{L} = \mathcal{L}_{task} + \lambda \sum_{l \in L} \mathcal{L}_{reg}(w^l) \tag{31}$$

with L the set of layers we want to regularize.

In the above formulation, the design of $\mathcal{L}_{reg}$ is not an easy task as equivariance depends on the group action on the layer's input and the group action on the layer's output. For instance, in the first layer of the network, there are many possible rotation equivariant parametrizations which would induce different group actions on the output. Each of these actions would make the first layer equivariant while defining a different transformation to which the second layer's regularization would need to account for. To address this, in this work, we settle for parametrizations, which approximate the one described in the previous section (*hard*). One problem of approximating a fixed structure is that if at a particular internal representation of the network, a rotation of the input, T_θ, does not correspond to the output transformation L_θ, the method loses its regularization property. Also, note that although these methods contain more parameters for the same number of channels, when compared to the *hard* strategy, the fact that they approximate it can be used for model compression, by encoding the small differences to the *hard* structure (residues) with less bits. These methods are listed below.

Difference Decay (decay) For each weight tensor the loss term is given by:

$$\sum_i \sum_{k_1=k_2+1}^{3} \sum_{k_2=0}^{3} \left\| w_{4i+k_1} - L_{\frac{(k_2-k_1)\pi}{2}} w_{4i+k_2} \right\|_2^2 \tag{32}$$

It is easy to check that $\mathcal{L}_{decay}$ reaches a minimum when the filters mimic the parametrization seen in the previous section.

Alignment (align) The second strategy uses the inner product to capture the idea of alignment. The following quantity is minimized:

$$\sum_i \frac{1}{16} \sum_{k_1=0}^{3} \sum_{k_2=0}^{3} (1 - w_{4i+k_1}^T L_{\frac{(k_2-k_1)\pi}{2}}.w_{4i+k_2})^2 \tag{33}$$

In this case, minimizing $\mathcal{L}_{align}$ requires mimicking the equivariant parametrization but also that the filters have norm 1. This requirement can also be found on other regularization methods such as weight orthonormality regularization [16].

Rotation Variant δ (comp) In this strategy we parameterize filters as a combination between two components, one using the hard structure and one conventional filter.

$$w'_{4i+k} = L_{\frac{k\pi}{2}}.w_i + \delta_{4i+k} \tag{34}$$

A stronger L2 penalty is applied to the rotation variant component δ. The δ's can be seen as residues and are initialized as zeros.

C. Soft priors on the Activations

Alternatively, to achieve rotation equivariance one can penalise non-equivariant intermediate representations in the network. This strategy is similar to that of [7] but focusses on equivariance instead of invariance. In this case, the regularization term takes the form of:

$$\mathcal{L} = \mathcal{L}_{task} + \lambda \sum_{g \in G} \left[T'_g.F(x) - F(T_g.x) \right] \tag{35}$$

Different regularization methods can be designed based on this general term, depending on the group G and the transformations T and T' considered. In this work, we are dealing with 90º rotations of the input, which define both T and G. Thus, the proposed strategies only change T'_g. One important consideration for these methods is that it requires additional computation to obtain $F(T_g.x)$. As such, contrary to weight regularization strategies, its impact on optimization time is not negligible. Also, it does not allow for model compression as the weights are not similar to each other, but only the activations.

Fixed T' (fixed) This strategy defines T' to be the L_θ from the previous section. As such, for each activation, the following term is minimized:

$$\sum_{\theta \in G} [L_\theta.F(x)] - F(h_{-\theta}.x) \tag{36}$$

Note that there are other parametrizations, different from the *hard* strategy, that minimize this term. For instance, one could reorder the filters and their coefficients in an orderly way and maintain the equivariance property. For this strategy, the filters are initialized with the rotation equivariance structure and allowed to diverge from there.

Learned T' (learn) An alternative strategy is to learn the transformation along with the model. For this we define the matrix M whose value $M_{m,n}$ expresses the equivalence between the feature extracted by w_n from the input and w_m from the same input after a $\frac{\pi}{2}$ rotation. Formally:

$$\sum_{\theta \in G} [M.F(h_{-\theta-\frac{\pi}{2}}x)] - F(h_{-\theta}.x) \tag{37}$$

Note that, contrary to all the previous techniques, here the group action on the output is not defined. The only requirement is that it exists and that F is equivariant under plane rotations of the input and $M-$multiplication on the output. Although M is not defined for the other elements of a group it could be obtained by computing M^b. In practice however, the minimization of Equation 37 requires that M can be optimized efficiently by gradient descent. Initialization of the weights is done similarly to the previous method. M is initialized as a "shifted" identity matrix to account for the channel shifting effect of L_θ. In this way we guarantee that at the start the regularization term is zero.

V. Experiments

In this work, we study the rotation equivariance prior for feature learning in CNNs. Specifically, we assess if this is a good prior for modeling conventional image data, and its use is not limited to problem formulations with rotation symmetries (e.g., aerial images). Also, we evaluate the effectiveness of the proposed strategies for introducing this prior and their effect on the model's ability to generalize to unseen data.

To this end, the task of image classification was focussed. The following data was used for the proposed validation:

- CIFAR [17] - The CIFAR10 and CIFAR100 are well-known datasets for classification tasks. They are divided into a standard train and test split with 50k and 10k images, respectively. The dataset was augmented using random cropping after 4-pixel padding on each side of the image, along with random horizontal flipping.
- SVHN [18] - Street View House Numbers is another well-known classification dataset for digit recognition. No data augmentation was used in this case.
- SINS10 [19] - The Scaled ImageNet Subset dataset is composed of 100k colored images of ten classes. The images have a side dimension of 96px. The dataset contains ten folds of equal size. The first eight were used for training while the last two were used for testing.

As baseline, a VGG19 [20] architecture was optimized from scratch in each dataset, using stochastic gradient descent with momentum. The network was adapted by reducing the number of neurons in the fully-connected layers. The batch size was set to 128 for all datasets. Batch normalization and weight decay were used, along with dropout for the fully connected layers. The learning rate and the number of epochs were adapted for each dataset. Initialization was done using LSUV [21].

A. Measuring Rotation-Equivariance

One way to measure how similar two features are is to estimate their correlation coefficient (ρ). Based on this, we define the following measure to assess how equivariant are the features extracted by a CNN up to a certain layer:

$$\mathbb{E}_{x \in \mathcal{D}} \left[\frac{1}{16} \sum_{\theta_1, \theta_2 \in G} \frac{1}{K} \sum_{i=0}^{K-1} \max_{0 \leq j < K} \rho \left[F_i(h_{\theta_1}.x), F_j(h_{\theta_2}.x) \right] \right] \tag{38}$$

This function will return a value close to one if, for each data point, the extracted features are well-correlated for different orientations of the input.

To evaluate the ability of each strategy to approximate equivariant representations, the first two convolutional blocks (4 convolutional layers) of the VGG19 model were regularized at different values of λ. For the activation-based strategies, the output of the second block was used to compute the regularization loss. Each model was trained on the first 40k images of the CIFAR10 dataset. Then, for each, the measure described above was computed for the validation data (the last 10k images of the training set) and random data.

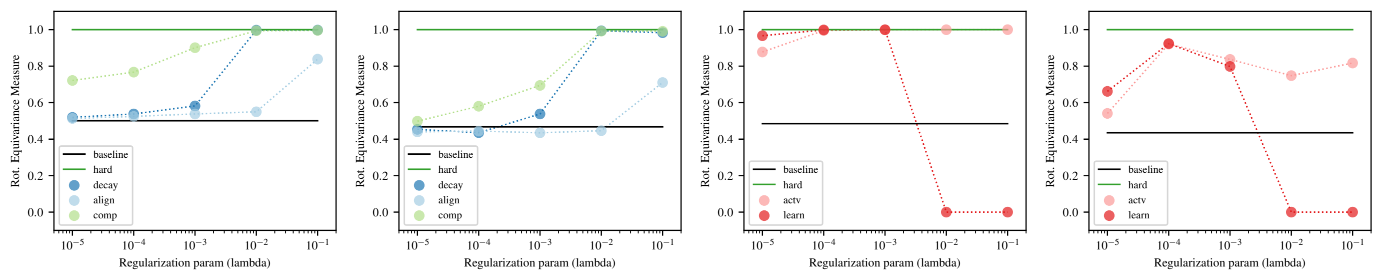
As shown in Figure 4, weight-based methods progressively become more equivariant as λ is increased. The difference between the validation data and the random data is small, revealing that they maintain this property even for patterns that are not frequent in the training data, similar to the use of *hard* structured filters. This is to be expected as their weights are close to the hard structure, which is necessarily equivariant. Interestingly, the parametrization of *comp* leads to a model more equivariant than the baseline, even for small values of λ.

Activation-based methods are also able to learn equivariant representations for the validation. However, their behavior changes for random data. The fact that they were not exposed to some of these visual patterns during training means they were not optimized to recognize them at multiple orientations. The zero value for high values of λ using the learn strategy signifies non-convergence, indicating instability of the model for high values of λ.

B. Generalization

To assess how each method affects the ability of the model to generalize to unseen data, the baseline network was regularized in the first two convolutional blocks for all four datasets. For the SVHN and SINS10 datasets, the experiment was repeated while regularizing the first four convolutional blocks. For the activation-based methods, the output of the last layer of these blocks was used. The regularization parameter λ was optimized for the CIFAR10 on the training data, by training on the first 40k images and leaving the last 10k for validation. Each experiment was repeated five times, and average accuracy and standard deviations are reported. Due to the high number of classes in CIFAR100 we also report the top-5 accuracy. The results are shown in Table I.

As shown, *hard* constraining the weights or regularizing them using weight-based methods leads to an increase in

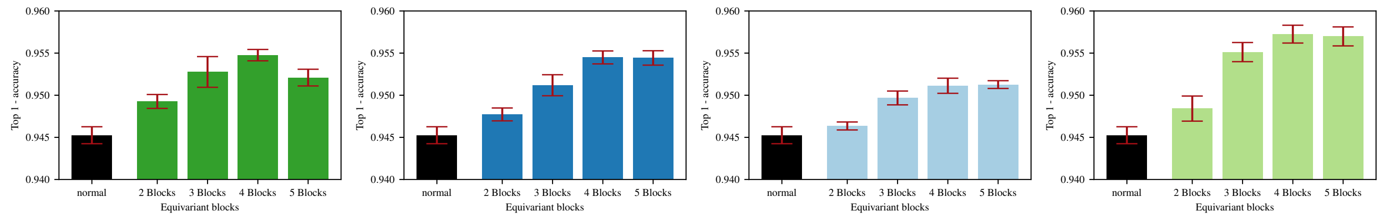


(a) Weight-based on val. data (b) Weight-based on rnd. data (c) Activation-based on val. data (d) Activation-based on rnd. data

Fig. 4: Rotation equivariance measure for different methods of regularization. The horizontal lines correspond to the baseline and the hard constrained methods and were empirically obtained for each setting. The value of zero in the measure for the *learn* strategy indicates non-convergence.

TABLE I: Classification accuracies for the four datasets for each regularization method. Results are in %

	CIFAR10	CIFAR100		SVHN	SINS10	SVHN	SINS10
	2 Rot. Equiv. Blocks					4 Rot Equiv. Blocks	
Strategy	Top1 - Acc.	Top1 - Acc.	Top5 - Acc.	Top1 - Acc.	Top1 - Acc.	Top1 - Acc.	Top1 - Acc.
Baseline	91.88±0.23	69.45±0.18	89.19±0.15	94.53±0.10	93.20±0.15	94.53±0.10	93.20±0.15
Hard	92.04±0.17	69.99±0.19	89.63±0.17	**94.93±0.08**	93.54±0.07	95.48±0.07	93.86±0.13
Decay	**92.43±0.24**	**70.48±0.33**	**89.64±0.18**	94.77±0.08	93.40±0.09	95.45±0.08	94.01±0.11
Align	92.07±0.11	69.86±0.18	89.43±0.16	94.64±0.05	93.28±0.11	95.11±0.09	93.55±0.08
Comp	92.10±0.14	70.38±0.23	**89.64±0.14**	94.84±0.15	**93.57±0.10**	**95.73±0.11**	**94.03±0.11**
Actv	91.66±0.16	69.63±0.38	89.08±0.29	93.99±0.06	92.54±0.15	93.76±0.15	91.37±0.52
Learn	91.86±0.18	69.64±0.19	89.05±0.24	94.12±0.17	92.66±0.16	93.92±0.20	92.17±0.27



(a) Hard (b) Decay (c) Align (d) Comp

Fig. 5: Effect of regularization for different numbers of convolutional blocks on SVHN dataset.

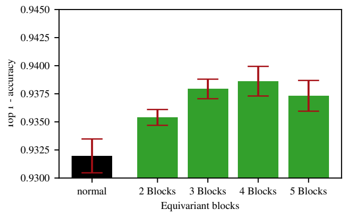
classification accuracy for all datasets. This increase is more noticeable when regularization is applied for four convolutional blocks on the SVHN and SINS10 datasets. Regarding the activation methods, they always lead to comparable or worse results than the baseline. The three strategies that consistently lead to better generalization, when compared to all the remaining ones, were *hard*, *decay*, and *comp*.

The fact that different methods of encoding the same prior lead to a consistent increase in the test set accuracy strongly suggests that rotation equivariance is an important factor for generalization in these settings. This was observed for datasets composed of images with different characteristics and mostly without rotational symmetries, suggesting that its usability is not limited to a narrow set of problems.

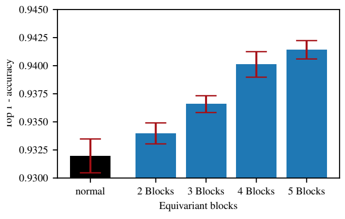
The relatively low accuracy of activation-based methods across different datasets, combined with the previous set of experimental results leads to the conclusion that even though these methods produce equivariant representations on unseen data, this does not equate to better generalization. Possible reasons for this include the fact that applying this regularization might lead to optimization problems or that the model is trivially minimizing the objective (e.g., by learning rotation-invariant features). After inspecting the activations of the layer where the activation difference was minimized, we verified that they were smaller on average when compared to the baseline. We also verified that initializing the baseline's weights with an equivariant structure (the one used for these models) does not affect the final test set accuracy.

Finally, we note that regularizing the filters of the intermediate layers (blocks 3 and 4) also leads to a significant increase in test set accuracy, as these were the best models in both the SVHN and SINS10 datasets. This shows that the usefulness of the proposed prior is not limited to the early features of the network. To investigate this, we run the experiments for each weight-based strategy for different numbers of regularized blocks, from two up to five. The results for the SVHN and SINS10 are shown in Figures 5 and 6.

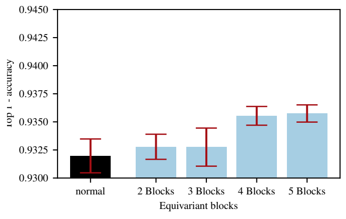
The results generally show that, as we increase the number of regularized layers, the test-set accuracy is increased or maintained. The only exception to this is the *hard* strategy.



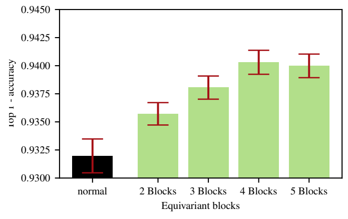
(a) Hard



(b) Decay



(c) Align



(d) Comp

Fig. 6: Effect of regularization for different numbers of convolutional blocks on SINS10 dataset.

Also, there is no clear winner strategy between the three best strategies (*hard*, *comp* and *decay*), with the "number of layers regularized" being the variable that most affects the generalization ability.

C. The Rotation Equivariance Prior

The experimental component of this work shows that different strategies for incorporating the notion that "features in one orientation should be useful in other orientations" makes learning more robust and thus serves as a regularization strategy. The usability of such methods is not limited to applications which contain rotational symmetries and its implementation is straight-forward in most scenarios. It is important to mention that the regularization proposed was not used in spite of other methods (weight decay, batch norm and dropout) but alongside them.

Equivariance plays a key role in neural network optimization and their ability to generalize to unseen data. This idea is central in the design of CNNs. Image models, such as these, should be adapted for rotation equivariance in the generality of computer vision tasks.

VI. Conclusion

Rotation equivariant features are learned by CNNs from data in the generality of computer vision tasks, for early layers. Thus, incorporating this idea into the training process or the architecture of the model has the potential to make learning more robust in the general setting.

In this work, we frame rotation equivariance as a prior on the distribution of the weights of CNNs. We propose different regularization techniques that capture the notion of *approximate equivariance*, leading to the optimization of soft rotation equivariant CNNs. We quantitatively evaluate the equivariance and generalization properties of the proposed models. Finally, we demonstrate that incorporating this prior in CNNs leads to better generalization in different image datasets.

Equivariance plays a central role in the generalization ability of CNNs. These models should become better at modeling natural data, as we increase the number of useful equivariance properties these models have. Rotation equivariance is one of such properties.

References

[1] C. Olah, A. Mordvintsev, and L. Schubert, "Feature visualization," *Distill*, 2017, https://distill.pub/2017/feature-visualization.

[2] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, Jul 2019.

[3] L. Taylor and G. Nitschke, "Improving deep learning with generic data augmentation," in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, Nov 2018, pp. 1542–1547.

[4] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. van den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

[5] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, *Mitosis Detection in Breast Cancer Histology Images with Deep Neural Networks*. MICCAI, 2013.

[6] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys, "Ti-pooling: Transformation-invariant pooling for feature learning in convolutional neural networks," *CVPR*, 2016.

[7] G. Cheng, P. Zhou, and J. Han, "Learning rotation-invariant convolutional neural networks for object detection in vhr optical remote sensing images," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 12, pp. 7405–7415, Dec 2016.

[8] T. S. Cohen and M. Welling, "Group equivariant convolutional networks," in *ICML 2016*.

[9] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow, "Harmonic networks: Deep translation and rotation equivariance," in *CVPR*, 2017, pp. 7168–7177.

[10] T. Cohen and M. Welling, "Steerable CNNs," *ICLR*, 2017.

[11] J. E. Lenssen, M. Fey, and P. Libuschewski, "Group equivariant capsule networks," in *NIPS*, 2018, pp. 8844–8853.

[12] K. Lenc and A. Vedaldi, "Understanding image representations by measuring their equivariance and equivalence," *International Journal of Computer Vision*, vol. 127, no. 5, pp. 456–476, May 2019.

[13] R. Kondor and S. Trivedi, "On the generalization of equivariance and convolution in neural networks to the action of compact groups," in *ICML*, 2018, pp. 2752–2760.

[14] S. Ravanbakhsh, J. Schneider, and B. Póczos, "Equivariance through parameter-sharing," in *ICML*, 2017, pp. 2892–2901.

[15] T. Cohen, M. Geiger, and M. Weiler, "A general theory of equivariant cnns on homogeneous spaces," *CoRR*, vol. abs/1811.02017, 2018.

[16] N. Bansal, X. Chen, and Z. Wang, "Can we gain more from orthogonality regularizations in training deep networks?" in *NeurIPS*, 2018, pp. 4261–4271.

[17] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.

[18] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

[19] T. U. of Waikato, "Sins-10 dataset," https://www.cs.waikato.ac.nz/ ml/sins10/.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.

[21] D. Mishkin and J. E. S. Matas, "All you need is a good init," *CoRR*, vol. abs/1511.06422, 2015.