What Affects Learned Equivariance in Deep Image Recognition Models?

Bruintjes, Robert-Jan; Motyka, Tomasz; Gemert, Jan van

DOI
10.1109/CVPRW59228.2023.00512

Publication date
2023

Document Version
Final published version

Published in
Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)

2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) | 979-8-3503-0249-3/23/$31.00 ©2023 IEEE | DOI: 10.1109/CVPRW59228.2023.00512

What Affects Learned Equivariance in Deep Image Recognition Models?

Robert-Jan Bruintjes*[,1], Tomasz Motyka*[,2], Jan van Gemert[1]
[1]Computer Vision Lab, Delft University of Technology, [2]Synerise
{r.bruintjes,j.c.vangemert}@tudelft.nl, tomasz.motyka@synerise.com

Abstract

Equivariance w.r.t. geometric transformations in neural networks improves data efficiency, parameter efficiency and robustness to out-of-domain perspective shifts. When equivariance is not designed into a neural network, the network can still learn equivariant functions from the data. We quantify this learned equivariance, by proposing an improved measure for equivariance. We find evidence for a correlation between learned translation equivariance and validation accuracy on ImageNet. We therefore investigate what can increase the learned equivariance in neural networks, and find that data augmentation, reduced model capacity and inductive bias in the form of convolutions induce higher learned equivariance in neural networks.

1. Introduction

Equivariance in neural network features allows invariance to geometric transformations [6,23], making such networks more data efficient [25,38,40], parameter efficient [6] and robust to out-of-distribution transformations [1,13,33].

Equivariance with respect to specific geometric transformations can be designed into the neural network architecture [5, 6, 37]. However, even with careful design, it may happen that the resulting architecture is not as equivariant as intended [13, 17, 19, 45]. An example is the convolution operator in Convolutional Neural Networks (CNNs) for translation equivariance, which can be broken by border effects [17, 19] or pooling [45]. On the other hand, even if neural networks are not designed to be equivariant, they can still *learn* equivariance naturally. Existing works demonstrate qualitative examples of learned equivariant features [2,10,29]. However, how much equivariance is learned, and which factors affect equivariance, are open questions.

In this work, we quantify learned equivariance in image recognition neural networks that have and have not been explicitly designed for equivariance. Where existing works [3,15,18,27,46] typically only measure equivariance at the output of the network, we measure equivariance for all intermediate layers. To do so, we deviate from existing measures of learned equivariance which are inconsistent across network depths, and we design a consistent measure.

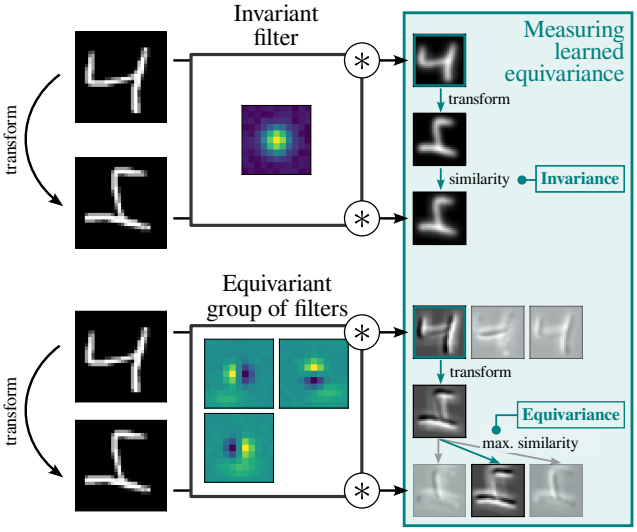


Figure 1. Neural networks can learn features that are invariant or equivariant w.r.t. a geometric transformation of the data, such as rotation. We measure learned equivariance w.r.t. translation and rotation in neural networks.

Using our measure for learned equivariance, we find evidence that learned translation equivariance in intermediate features of neural networks correlates with increased validation accuracy on ImageNet. We therefore investigate how we can increase learned equivariance by changing how we train neural networks. In particular, we find that 1) making the task equivariant does not increased learned equivariance; 2) data augmentations designed for invariance indeed increase learned equivariance, even in early and middle layers; 3) reducing model capacity increases learned equivariance, suggesting that equivariant features arise from a need to compress representations; 4) CNNs learn more translation and rotation equivariance in intermediate features than

*Equal contribution.

the Vision Transformers (ViTs).

We make the following contributions:

- We propose a new measure for learned equivariance that is allows comparing learned equivariance of features at different depths of the network.
- We show evidence for a positive correlation between learned translation equivariance in intermediate features and validation accuracy on ImageNet.
- We test how several aspects of neural network training affect learned equivariance. In summary, we find that data augmentation, reduced model capacity and the inductive biases of CNNs positively affect learned equivariance.

2. Related Works

Neural networks can learn equivariant features from data [24, 28, 29]. Particularly inspiring is the work by Olah *et al*. [29], that demonstrates by precise and meticulous manual investigations that learned equivariant features exist in networks that were not designed to be equivariant. Inspired by this work, we here investigate how to move beyond laborous manual qualitative investigations, and instead offer a quantitative approach, by giving an automatic measure for learned equivariance.

A number of existing works measure equivariance in neural networks. [15] study models from the pre-Deep Learning era which have since been superseded by the models we study. More recent works measure equivariance in Convolutional Neural Networks, with KL divergence on class probabilities [46], with Euclidean distance [18], or cosine similarity [3, 27] on feature maps. In our work, we show that the cosine similarity is not appropriate for measuring equivariance in intermediate feature maps, and offer a correlation-based measure.

Several works study how neural network hyperparameters and datasets affect learned translation equivariance in the final output of the network. The kernel and padding sizes of the architecture affect translation invariance [27], although data augmentation might have a bigger effect on translation invariance than the network architecture [18]. Similar conclusions about the importance of the data were found by others [3,4]. Here, we follow these investigations, and extend them by analyzing the impact on the intermediate layers.

There are some works that study equivariant properties of intermediate features. Recently, [26] proposed a method to detect invariance to any learned Lie group for intermediate features. However, they do not study equivariance, like we do. Other works study only the transformation group of translations ($\mathbb{Z}^2$). [45] measures the translation equivariance by computing cosine similarity between feature maps to show how max pooling violates the translation equivariance property. [17, 19] show that some padding methods disrupt the translation equivariance property in CNNs. [32] measure the invariance of intermediate representations using normalized cosine similarity to study the effect of pooling on deformation stability. Where these works diagnose issues with designed equivariance and test for their effects, we consider learned equivariance in a more general sense, including transformation groups not designed into the network, such as rotations.

3. Method

Neural networks can learn to be equivariant in two ways: either by learning invariant features or by learning equivariant groups of features, as shown in Fig 1. In this section we detail how we can measure the quantity of invariant features and equivariant groups of features. We discuss which similarity measure is appropriate for measurements of learned equivariance in features at different depths of a neural network. Finally, we verify our measures using artificially engineered equivariant CNNs.

In the following we will refer to invariant features and equivariant groups of features under the single predicate "learned equivariance", as invariance is a special case of equivariance.

3.1. Invariant features

We derive a measure of learned equivariance from inspecting the definition of equivariance [6] applied to a single neural network layer:

$$f(T_g(X)) = T'_g(f(X)), \tag{1}$$

where $X \in \mathbb{R}^{C_{\text{in}} \times H \times W}$ is an image or a feature map, $f(X) \in \mathbb{R}^{C \times H \times W}$ is the output of a neural network operation with C output features and T_g is the application of a transformation g from a transformation group G. For example, if $G = \mathbb{Z}^2$, then g is a translation with a particular integer-valued (x, y) offset. If T'_g is the identity function for all g, the layer f is *invariant* w.r.t. transformation T:

$$f(T(X)) = f(X). \tag{2}$$

Without designing invariance to T into neural network layer f, each individual feature in $f(X)$ can learn to behave invariant or not invariant with respect to T. We therefore define invariance for each feature $c \in C$ independently:

$$f(T(X))_c = f(X)_c. \tag{3}$$

In Fig. 1 we show an example where T is a 90° rotation.

To measure a feature's invariance w.r.t. g, we compute the similarity between $f(T(X))_c$ and $f(X)_c$:

$$\text{Invariance}(f_c, g) = S(f(T(X))_c, f(X)_c), \quad (4)$$

given a similarity function $S : \mathbb{R}^{\mathbb{H}\times\mathbb{W}} \times \mathbb{R}^{\mathbb{H}\times\mathbb{W}} \to [0, 1]$. Given invariance measures for each feature, we can average these measures for all features in a layer to compute a layer's invariance.

3.2. Equivariant features

A group of features $C_G \subseteq C$ in a neural network layer is equivariant with respect to a transformation group G if each feature $c' \in C_G$ activates for a different transformation g from the group. In other words, for a sample $T_g(X)$ transformed with any transformation from the group, the group of feature maps $f(X)_{c'\in C_G}$ will have one feature c' whose transformed feature map $T_g(f(X))_{c'}$ matches $f(T_g(X))_{c'}$:

$$f(T_g(X))_{c'} = T_g(f(X))_{c'}, \quad \exists c' \in C_T. \quad (5)$$

In Fig. 1 we show an example where g is a 90° rotation.

To fit this with the definition of equivariance (Eq. 1) we define T'_g:

$$f(T_g(X))_{c'\in C_G} = T'_g(f(X))_{c'\in C_G} \quad (6)$$

where T'_g transforms with T_g and selects feature c' that matches the transformation g. When this equation holds, the feature group C_G is equivariant w.r.t. T.

To measure the equivariance of a feature c we find the maximum similarity between $f(T_g(x))_c$ and $T_g(f(x))_{c'}$ over all features $c' \in C$, for a given transformation g:

$$\text{Equivariance}(f_c, g) = \max_{c'\in C} S(f(T_g(X))_c, T_g(f(X))_{c'}) \quad (7)$$

given a similarity function $S : \mathbb{R}^{\mathbb{H}\times\mathbb{W}} \times \mathbb{R}^{\mathbb{H}\times\mathbb{W}} \to [0, 1]$. Given equivariance measures for each feature, we can average these measures for all features in a layer to compute a layer's equivariance. Note that if a feature is invariant w.r.t. T_g, we will measure an equivariance score that is at least as high as the invariance score. As invariance is a special case of equivariance, this behavior of our measure is intended.

3.3. Measuring similarity

We need to choose a similarity measure $S : \mathbb{R}^{\mathbb{H}\times\mathbb{W}} \times \mathbb{R}^{\mathbb{H}\times\mathbb{W}} \to [0, 1]$ with which to compare feature maps to measure invariance and equivariance. In existing works, cosine similarity is commonly used as a similarity measure used to compute invariance or equivariance of the networks representations [4, 27, 45]. However, cosine similarity is sensitive to the mean values of its input vectors. This behaviour is depicted in Figure 2a. As different layers in a neural network have different mean activation values (see Fig. 2b), this biases the similarity measure.

We propose to use Pearson correlation [14] instead. Pearson correlation, also called centered cosine similarity, is a similarity measure that does not suffer from sensitivity to the mean of the inputs, as it computes the covariance of the inputs normalized by their standard deviations. It is the basis of many methods for comparing network representations [20, 22, 31].

To motivate our choice, we visualize the difference between using cosine similarity and correlation for measuring equivariance in the following example. We train ResNet-44 [16] model on CIFAR-10 [21] dataset and compute invariance w.r.t. to 90° rotation after each residual block. In Figure 2 we show the qualitative comparison between the scores, computed using cosine similarity and correlation, and the mean of the activations. Additionally, we compute a correlation between the magnitude of the activations and equivariance scores computed with cosine similarity (**0.63**) and correlation (**0.11**). Scores computed using cosine similarity correlate visibly with the mean of the activation while, for the scores computed using correlation, this effect is less prevalent. In our experiments, we therefore use correlation as a measure to quantify equivariance.

4. Experiments

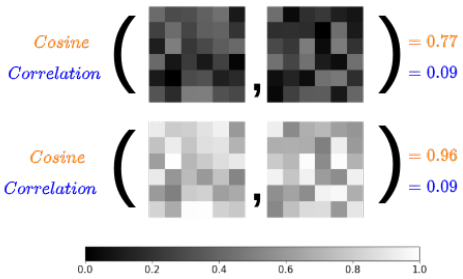
4.1. Controlled experiments

To verify that our method captures equivariance, we apply it to two controlled toy settings. We create two 3-layer CNNs with hand-crafted filters such that we expect to measure perfect learned rotation invariance and equivariance respectively. For the invariant model, we set all the filters to be rotationally symmetric, using a 2D isotropic Gaussian function, and measure the invariance after each layer (Fig. 3a). For the equivariant model, we cut out corners of the filters from the invariant model such that all the filters are rotations of one another (Fig. 3b). Our measure finds both models capture exactly the intended learned equivariances, demonstrating the validity of our measure.

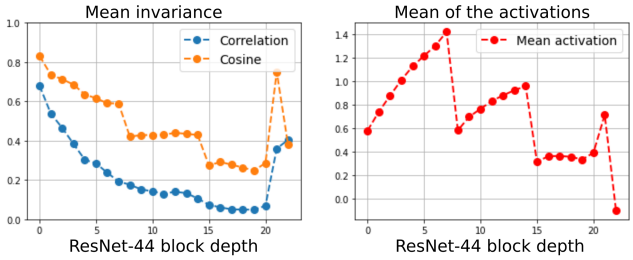
4.2. Does learned equivariance improve accuracy?

We study the relationship between the validation accuracy and the amount of learned equivariance in large-scale seminal models. For each part of each trained model we compute Spearman's rank correlation between the amount of invariance or equivariance and the ImageNet validation accuracy of the model.

We test four CNNs (EfficientNet-B6 & EfficientNet-B7 [35], ResNeXT-101 [41] and Inception-V3 [34]) and two Vision Transformer variants (Vision Transformer [11] and MLP-Mixer [36]). We measure invariance and equivariance for both translation and rotation for 2000 images from the ImageNet validation set. We do not train the models ourselves but instead use available checkpoints from

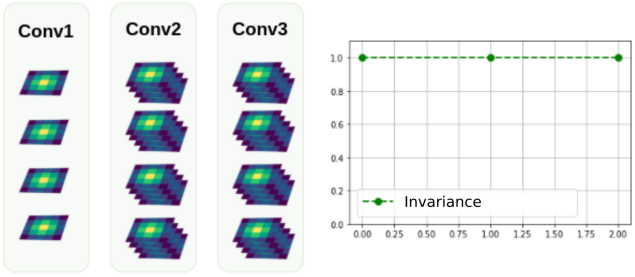


(a) Comparison of different similarity measures on random feature maps. Feature maps on the bottom are shifted by 0.5 with respect to the top ones. Cosine similarity is sensitive to such shifts while correlation is invariant.

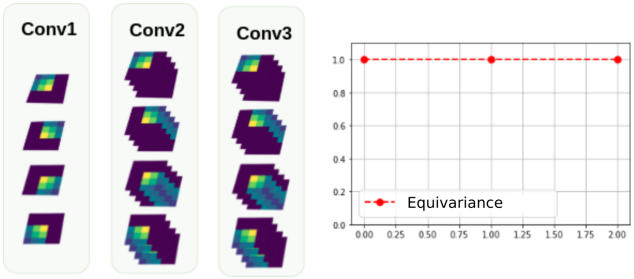


(b) Comparison between magnitude of the activations (red) and equivariance computed using correlation (blue) and cosine similarity (orange). We compute correlation between the magnitude of the activations and equivariance scores computed with correlation (**0.11**) and cosine similarity (**0.63**).

Figure 2. Analysis of the influence of magnitude of weights on different similarity measures.



(a) Rotation invariance



(b) Rotation equivariance

Figure 3. We create two controlled toy CNNs, each designed to be perfectly invariance and equivariant respectively, to test if our method measures equivariance correctly, which it does.

`torchvison` [30] or `timm` [39]. Since the studied model families do not have the same exact number of layers, we divide each model into depth-wise parts and report the average equivariance measures over all layers in each part. Feature maps from the beginning of the network until the global average pooling (GAP) layer are uniformly partitioned into *Early*, *Middle* and *Late* parts. *Pool* captures the feature maps directly after the GAP layer and *Final* is the feature map directly before the softmax layer. We discriminate between the *Pool* part and the *Final* part to identify what role in achieving equivariance the global pooling and final classifier have.

Figure 4 shows there is some correlation between translation equivariance in *Early* and *Middle* layers and accuracy on ImageNet, while attaining almost perfect correlation in the *Final* part. In contrast, for rotations there is little correlation between the equivariance in the representation before global pooling and the validation accuracy.

Even though the sample size (six models) for this correlation test is small, we conclude that there is some evidence for the benefit of learning translation equivariance in intermediate features of neural networks trained on ImageNet. In the following we therefore study what can increase the learned equivariance in such networks.

4.3. Equivariance in the data

On tasks where invariant responses are beneficial to solve the task, e.g. translation invariance in image recognition, one may wonder how this invariance is achieved. We study how learned equivariance in intermediate features is affected by adding transformations to the data and therefore into the task. We choose to study rotation transformations on CNNs, as rotation equivariance is not designed into CNNs. We study whether there is a difference if the task is invariant or equivariant with respect to introduced transformations.

We train a 7-layer CNN taken from [6], consisting of 7 layers of 3×3 convolutions, 20 channels in each layer, ReLU activation functions, batch normalization, and max-pooling after layer 2, on 3 different datasets. We test on three different datasets. The first dataset is *MNIST6*, which is the regular MNIST [9] without $\{0, 1, 6, 8\}$ classes, to get rid of rotational transformations that these classes have. For example, digit 8 is very similar to its $180°$ rotation, so, by default, this class would introduce some rotation invariance, which is undesirable as we want to control for rotation invariance in this setting. Second is the *MNIST6-Rot-Inv* where every digit in *MNIST6* is randomly rotated by $r \in \{0°, 90°, 180°, 270°\}$ upfront. This dataset imposes invari-

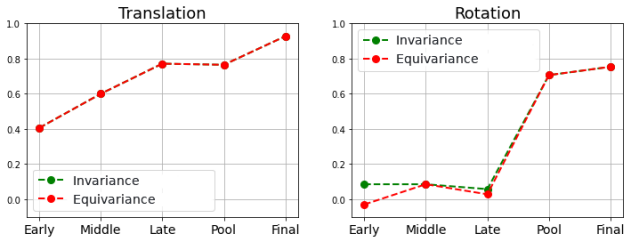


Figure 4. Spearman's rank correlation between learned equivariance and ImageNet validation accuracy. Translation equivariance in intermediate features correlates with increased accuracy on ImageNet, while rotation equivariance does not.

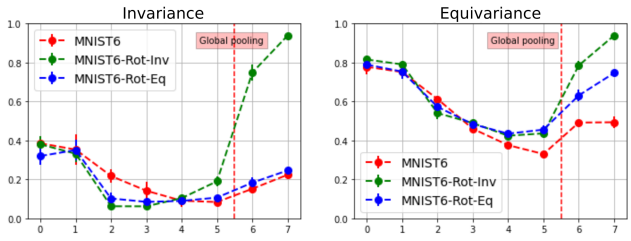


Figure 5. Learned rotation equivariance for rotation invariance/equivariance in the data. Invariance or equivariance in the task does not induce learning more equivariant features up until the late part of the network. Also there is no visible difference, up until the late part of the network, in the learned equivariance between the invariant and equivariant tasks.

ance into the task as, for every transformation, the predicted class should be the same. The last dataset, *MNIST6-Rot-Eq*, is created in the same way as *MNIST6-Rot-Inv*, but now the classes are made up of all combinations of digit number and rotation (e.g. class 2 is $(\text{digit}0°, 180°)$). This dataset imposes equivariance into the task. We compute in- and equivariance of the trained model for 2000 images from the validation set. We average the score over $90°, 180°, 270°$ rotations. Each experiment is repeated three times using different random seed. We train for 100 epochs using Adam with a batch size of 128 and a learning rate of 0.01, L2 regularization at 0.0005 and weight decay at epochs 25 and 50 with a factor of 10.0.

In Figure 5 we show the learned rotation equivariance. Firstly, we observe that the equivariance decreases with the depth, up to the final part after global average pooling (GAP), regardless of the task. For the tasks where the equivariance or invariance is imposed in the task, we see an increase in the final part, which suggests that GAP plays a significant role in achieving equivariance. Secondly, we do not see any significant differences between *MNIST6*, *MNIST6-Rot-Inv* and *MNIST6-Rot-Eq*, up to a later stage, which may indicate that early convolutional layers learn features with some amount of rotation equivariance regardless of the rotation invariance of the task. Finally, we observe that rotation equivariance is much larger then rotation invariance in the early and middle layers, which shows that CNNs do learn more rotated versions of the same feature in different channel rather then learning invariant, symmetrical features in a single channel. We conclude that introducing equivariance into the task does not significantly affect the learned equivariance of intermediate features.

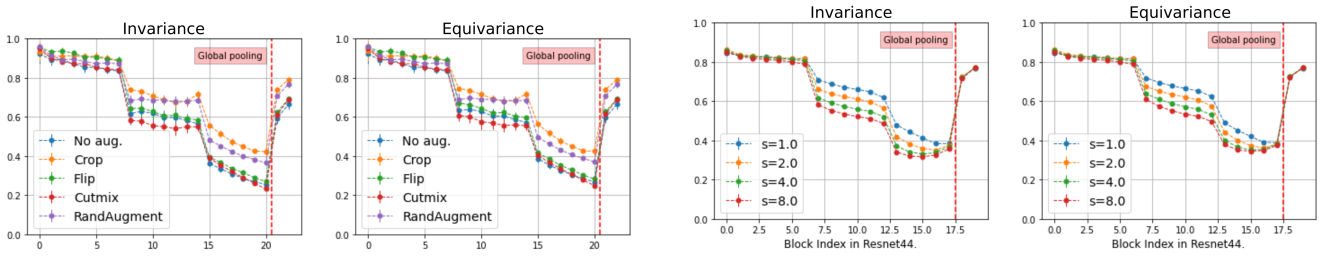
4.4. Data augmentations

By duplicating input samples under some transformation, data augmentation can induce invariance in the neural network. We study what the effect of data augmentation is on learned equivariance in intermediate representations: does data augmentation result in more invariant or equivariant intermediate features? For example, we know that the random crops data augmentation method, which essentially introduces random translations into the data, increases the model performance and translation invariance at the last layer [18]. The question is whether random crops increase the learned equivariance of intermediate features as well. In this experiment we study how translation equivariance is affected by different data augmentations.

We train ResNet-44 [16], adapted for CIFAR-10 [21], on the CIFAR-10 dataset using one of the following augmentations: random crops, horizontal flips, CutMix [43], RandAugment [7]. In each experiment we compute equivariance of the trained model over 2000 images from the validation set and average the score over diagonal shifts from one to 16 pixels. Each experiment is repeated three times by training the network different random seeds. We train for 200 epochs using SGD with a batch size of 128 and a learning rate of 0.1 and a momentum of 0.9, L2 regularization at 0.0001 and weight decay at epochs 100 and 150 with a factor of 10.0.

In principle we expect the in- and equivariance to be the same since translation equivariance should be provided by the convolution. However, we include equivariance in our experiments since there are works showing that the information about location can be encoded in different channels [17, 19].

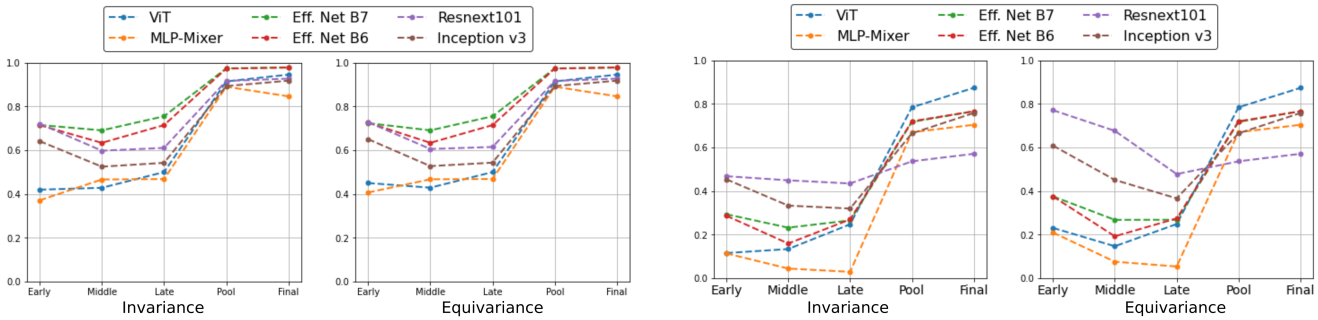
In Figure 6a we show learned translation equivariance for the tested data augmentations. Random crops and RandAugment increase the equivariance of learned features in the *Middle*, *Late* and *Final* parts, while the other data augmentation methods do not have any significant effect, with CutMix even having less equivariance than the baseline in the *Middle* part. We complement the finding of [18] by showing that random crops increase not only translation invariance but also translation equivariance in the intermediate layers. Also, we do not see any difference between invariance and equivariance for any data augmentation, which means that any equivariance learned is just invariance.



(a) Data augmentations

(b) Model capacity: number of channels s

Figure 6. Measuring learned translation equivariance for (a) data augmentations and (b) model capacity. For data augmentations (a), random crops and RandAugment increase channel equivariance the most, while other strategies have no discernible improvements. For model capacity (b), smaller models learn more in- and equivariance, although the amount of in- and equivariance in the end is similar.



(a) Learned translation equivariance

(b) Learned rotation equivariance

Figure 7. Measuring learned equivariance for inductive biases. For translation (a), the CNN variants exhibit more equivariance in the intermediate representation then the Vision Transformer variants. Global pooling seems to play an important role in achieving invariance. For rotation (b), the CNN variants exhibit more equivariance in the intermediate representation then the Vision Transformer variants. The *Early* and *Middle* parts have more equivariance than invariance.

4.5. Model capacity

We hypothesize that a smaller model in principle benefits from a more efficient representation and hence may learn more equivariant features. We therefore study whether model capacity influences learning translation equivariant representations. We train WideResNet-40 (WRN-40) [44] models, where we scale the number of channels (the "width") in each layer by a factor $s \in \{1, 2, 4, 8\}$. We train on the CIFAR-10 dataset and measure learned translation equivariance. The hyperparameters used for training are the same as in the data augmentation experiment of Sec 4.4.

In Figure 6b we show learned translation equivariance for different model capacities. We observe that the amount of translation equivariance is lower for the wider models, even though the amount of invariance in the final part is the same, which matches our hypothesis: an efficient representation learns to be equivariant.

4.6. Architectures

The architecture of a neural network determines which biases can be learned in training. Vision Transformers (ViTs) [11] lack certain inductive biases present in CNNs, which has been linked to their reduced data efficiency [12, 42]. We are interested to what extent the difference in inductive bias between CNNs and Vision Transformers (ViTs) affects learned equivariance.

We test architectures as they were designed for the ImageNet dataset [8], to faithfully represent their intended inductive bias. We use the same architectures and pretrained model weights as tested in Sec. 4.2: four CNNs (EfficientNet-B6 & EfficientNet-B7 [35], ResNeXT-101 [41] and Inception-V3 [34]) and two Vision Transformer variants (Vision Transformer [11] and MLP-Mixer [36]). We measure both translation and rotation equivariance on trained models for 2000 images from the ImageNet validation set. We also use the same depth-wise partitioning of feature maps into parts as used in Sec. 4.2. We measure translation equivariance over diagonal shifts of size 1 to 32 and rotation equivariance for $90, 180, 270$ rotations.

In Figure 7a we present the results for learned translation equivariance. We can see that ViT and MLP-Mixer have less translation equivariance than CNNs in *Early* and *Middle* layers. This is not unexpected, as convolutions directly integrate translation equivariance, whereas Vision

Transformers have to learn position embeddings that are translation equivariant. This reduced translation equivariance could be the reason for the poor data efficiency of ViT and MLP-Mixer [11, 36] since translation equivariance improves data efficiency [19]. Finally, we note that learned invariance and equivariance are identical for the tested models, meaning that these networks do not learn to represent different translations in different channels.

In Figure 7b we present the results for learned rotation equivariance. We observe that the ViT and MLP-Mixer have lower rotation equivariance than the CNNs in intermediate features, while after the GAP layer the ViT exhibits the most rotation equivariance out of all the models. Secondly, we note that early parts of all networks learn equivariant features that are not invariant, more so than in late parts of the networks. In contrast to the results for translation equivariance, we see that models with low rotation equivariance throughout *Early*, *Middle* and *Late* parts (ViT, Efficient-Net B6/B7) have the highest rotation equivariance in the *Final* part, while the models with highest equivariance in *Early*, *Middle* and *Late* parts (ResNeXT-101, Inception-v3) have the least equivariance in the *Final* part. This shows that high learned equivariance in the final model representation does not imply that intermediate representations are also highly equivariant.

5. Conclusion

We conduct a quantitative study on learned equivariance in intermediate features of CNNs and Vision Transformers trained for image recognition, using an improved measure of equivariance. We find evidence that translation equivariance in intermediate representations correlates with ImageNet validation accuracy. We show that data augmentations and reduced model capacity can increase learned equivariance in intermediate features. Also, the CNNs we test learn more translation and rotation equivariance in intermediate features than the ViTs we test.

Limitations. Our method allows to measure equivariance w.r.t. affine transformations only. The reason for that is the transformation g with respect to which we measure the equivariance has to be a map from and to an identical discrete domain, e.g. feature maps. This restriction disqualifies continuous transformations such as rotations with any other resolution than 90 degrees, or scaling with non-integer scaling factors.

Future work. Learned equivariance benefits image recognition models. However, applying equivariant priors usually adds additional cost in terms of memory or computation. Future work could study whether one can apply equivariant priors selectively within a neural network, saving computing cost where networks already learn to be equivariant. Additionally, we show that Vision Transformers learn less translation equivariance than CNNs. Future work could explore methods to increase translation invariance in Vision Transformers, to aid in their data efficiency.

Acknowledgements

Robert-Jan Bruintjes and Jan van Gemert are financed by the Dutch Research Council (NWO) (project VI.Vidi.192.100). All authors sincerely thank everyone involved in funding this work.

References

[1] Thomas Altstidl, An Nguyen, Leo Schwinn, Franz Köferl, Christopher Mutschler, Björn Eskofier, and Dario Zanca. Just a matter of scale? reevaluating scale equivariance in convolutional neural networks. *arXiv preprint arXiv:2211.10288*, 2022. 1

[2] James Bergstra, Aaron Courville, and Yoshua Bengio. The statistical inefficiency of sparse coding for images (or, one gabor to rule them all). *arXiv preprint arXiv:1109.6638*, 2011. 1

[3] Valerio Biscione and Jeffrey Bowers. Learning translation invariance in cnns. *arXiv preprint arXiv:2011.11757*, 2020. 1, 2

[4] Diane Bouchacourt, Mark Ibrahim, and Ari S. Morcos. Grounding inductive biases in natural images: invariance stems from variations in data. *arXiv preprint arXiv:2211.10288*, 2021. 2, 3

[5] Taco Cohen, Maurice Weiler, Berkay Kicanaoglu, and Max Welling. Gauge equivariant convolutional networks and the icosahedral cnn. In *International conference on Machine learning*, pages 1321–1330. PMLR, 2019. 1

[6] Taco S. Cohen and Max Welling. Group equivariant convolutional networks. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 2990–2999. JMLR.org, 2016. 1, 2, 4

[7] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2020. 5

[8] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. 6

[9] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. 4

[10] Sander Dieleman, Jeffrey De Fauw, and Koray Kavukcuoglu. Exploiting cyclic symmetry in convolutional neural networks. In *International conference on machine learning*, pages 1889–1898. PMLR, 2016. 1

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Trans-

formers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 3, 6, 7

[12] Stéphane d'Ascoli, Hugo Touvron, Matthew L Leavitt, Ari S Morcos, Giulio Biroli, and Levent Sagun. Convit: Improving vision transformers with soft convolutional inductive biases. In *International Conference on Machine Learning*, pages 2286–2296. PMLR, 2021. 6

[13] Tom Edinxhoven. Using and abusing equivariance. Master's thesis, Delft University of Technology, 2023. 1

[14] David Freedman, Robert Pisani, and Roger Purves. Statistics (international student edition). *Pisani, R. Purves, 4th edn. WW Norton & Company, New York*, 2007. 3

[15] Ian Goodfellow, Honglak Lee, Quoc Le, Andrew Saxe, and Andrew Ng. Measuring invariances in deep networks. *Advances in neural information processing systems*, 22, 2009. 1, 2

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3, 5

[17] Md Amirul Islam, Matthew Kowal, Sen Jia, Konstantinos G Derpanis, and Neil DB Bruce. Global pooling, more than meets the eye: Position information is encoded channel-wise in cnns. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 793–801, 2021. 1, 2, 5

[18] Eric Kauderer-Abrams. Quantifying translation-invariance in convolutional neural networks, 2017. 1, 2, 5

[19] Osman Semih Kayhan and Jan C van Gemert. On translation invariance in cnns: Convolutional layers can exploit absolute spatial location. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14274–14285, 2020. 1, 2, 5, 7

[20] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International Conference on Machine Learning*, pages 3519–3529. PMLR, 2019. 3

[21] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009. 3, 5

[22] Malte Kuss and Thore Graepel. The geometry of kernel canonical correlation analysis. 2003. 3

[23] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995. 1

[24] Karel Lenc and Andrea Vedaldi. Understanding image representations by measuring their equivariance and equivalence. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 991–999, 2015. 2

[25] Attila Lengyel and Jan van Gemert. Exploiting learned symmetries in group equivariant convolutions. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 759–763. IEEE, 2021. 1

[26] Artem Moskalev, Anna Sepliarskaia, Ivan Sosnovik, and Arnold Smeulders. Liegg: Studying learned lie group generators. *arXiv preprint arXiv:2210.04345*, 2022. 2

[27] Johannes C Myburgh, Coenraad Mouton, and Marelie H Davel. Tracking translation invariance in cnns. In *Artificial Intelligence Research: First Southern African Conference for AI Research, SACAIR 2020, Muldersdrift, South Africa, February 22-26, 2021, Proceedings 1*, pages 282–295. Springer, 2020. 1, 2, 3

[28] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. An overview of early vision in inceptionv1. *Distill*, 5(4):e00024–002, 2020. 2

[29] Chris Olah, Nick Cammarata, Chelsea Voss, Ludwig Schubert, and Gabriel Goh. Naturally occurring equivariance in neural networks. *Distill*, 5(12):e00024–004, 2020. 1, 2

[30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. 4

[31] Maithra Raghu, Justin Gilmer, Jason Yosinski, and Jascha Sohl-Dickstein. Svcca: Singular vector canonical correlation analysis for deep learning dynamics and interpretability. *Advances in neural information processing systems*, 30, 2017. 3

[32] Avraham Ruderman, Neil C Rabinowitz, Ari S Morcos, and Daniel Zoran. Pooling is neither necessary nor sufficient for appropriate deformation stability in cnns. *arXiv preprint arXiv:1804.04438*, 2018. 2

[33] Mateus Sangalli, Samy Blusseau, Santiago Velasco-Forero, and Jesus Angulo. Scale equivariant u-net. *arXiv preprint arXiv:2210.04508*, 2022. 1

[34] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 3, 6

[35] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019. 3, 6

[36] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272, 2021. 3, 6, 7

[37] Maurice Weiler and Gabriele Cesa. General e (2)-equivariant steerable cnns. *Advances in Neural Information Processing Systems*, 32, 2019. 1

[38] Maurice Weiler, Fred A Hamprecht, and Martin Storath. Learning steerable filters for rotation equivariant cnns. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 849–858, 2018. 1

[39] Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019. 4

[40] Marysia Winkels and Taco S Cohen. 3d g-cnns for pulmonary nodule detection. *arXiv preprint arXiv:1804.04656*, 2018. 1

[41] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017. 3, 6

[42] Yufei Xu, Qiming Zhang, Jing Zhang, and Dacheng Tao. Vitae: Vision transformer advanced by exploring intrinsic inductive bias. *Advances in Neural Information Processing Systems*, 34:28522–28535, 2021. 6

[43] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6023–6032, 2019. 5

[44] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016. 6

[45] Richard Zhang. Making convolutional networks shift-invariant again. *CoRR*, abs/1904.11486, 2019. 1, 2, 3

[46] Allan Zhou, Fahim Tajwar, Alexander Robey, Tom Knowles, George J Pappas, Hamed Hassani, and Chelsea Finn. Do deep networks transfer invariances across classes? *arXiv preprint arXiv:2203.09739*, 2022. 1, 2