Extracting Invariant Features From Images Using An Equivariant Autoencoder

Denis Kuzminykh KUZMINYKH@INSILICO.COM
Daniil Polykovskiy DANIIL@INSILICO.COM
Alexander Zhebrak ZHEBRAK@INSILICO.COM
Insilico Medicine, Rockville, USA

Editors: Jun Zhu and Ichiro Takeuchi

Abstract

Convolutional Neural Networks achieve state of the art results in many image recognition tasks. While their structure makes predictions invariant to small translations, some recognition tasks require invariance to other transformations, like rotation and reflection. We apply group convolutions to build an Equivariant Autoencoder with embeddings that change predictably under the specified set of transformations. We then introduce two approaches to extracting invariant features from these embeddings—Gram Pooling and Equivariant Attention. These two methods separate transformation-relevant information from all other image features. We use obtained embeddings in classification and clustering tasks and show an improvement of the classification quality on the learned embeddings compared to pure autoencoder and average pooling method. A visualization of the learned manifold shows that objects of the same class tend to cluster together, which was not observed for the pure autoencoder embeddings.

1. Introduction

Convolutional Neural Networks (CNNs) are extremely successful in computer vision tasks. Their hierarchical structure with convolutional weight sharing provides a strong prior capable of extracting relevant features from raw pixels. Due to the translation symmetry of the network, local patterns can be detected regardless of the spatial position in the image. However, while CNNs are invariant to local shifts, they do not exhibit rotation or reflection invariance. In many applications, we expect learned hidden representations and predictions of the model to be independent of the rotation angle. Medical, biological, aerial, astronomy images usually exhibit partial or full rotational symmetry. In this work, we explore multiple approaches to creating transformation invariant feature representations and validate our model on the augmented MNIST and Plankton (Cowen et al., 2015) datasets.

The most common method to promote the invariance is to augment the training dataset with randomly transformed images (Krizhevsky et al., 2012; Kuzminykh et al., 2018; Ronneberger et al., 2015). This approach, however, has multiple drawbacks. First, models have to extract implicit information about invariances directly from the data, leading to slower training. Also, there are no guarantees that learned features and predictions are the same across all transformations—invariance can be achieved only approximately. Another approach is averaging model outputs across all possible augmentations to make predictions invariant. While it is a viable approach for small invariance sets (like horizontal symmetries), this method significantly increases computational and memory costs for larger groups.

In this paper, we focus on creating transformation invariant image embeddings using convolutional autoencoders. Working with a dataset with natural invariances, we would expect transformed versions of the same object to have similar embeddings. However, this does not happen for standard models. Embeddings of transformed images tend to spread across the latent space. Consider, for example, an autoencoder trained on the augmented MNIST dataset. To visualize the embeddings, we apply a 3D Principal Component Analysis (PCA) to the embeddings obtained from an autoencoder and show the manifold in Figure 1. Notice that manifolds of images with a digit "one" form a circle, corresponding to different angles of this digit. This structure of the manifold is not optimal for classification. A more useful representation would map images of similar objects closer together, ignoring information like the degree of rotation. To get such embeddings, we focus on building invariant embeddings that remain constant across specified transformations.

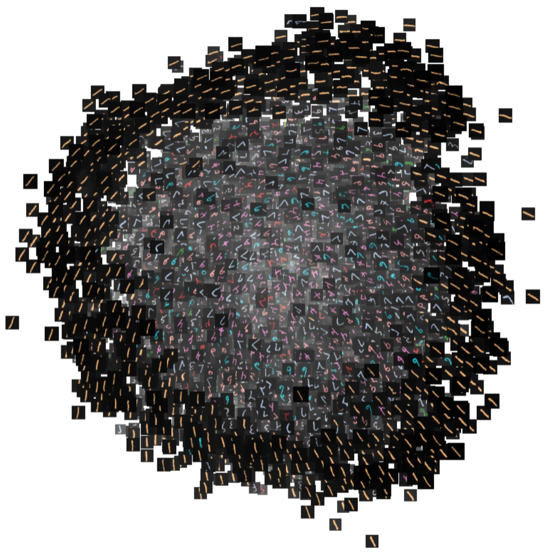
Figure 1: PCA of the augmented MNIST embeddings, obtained from an autoencoder. Notice how embeddings of ones form a circle.

We introduce a two-step process: first, we apply an Equivariant Autoencoder to extract equivariant features from the image. For any considered transformation of the input, we know exactly how the embeddings change. Then we apply a function that extracts invariant features from the equivariant embeddings (Figure 2). These features are transformation invariant and contain enough information to reconstruct an initial image up to a transformation. Our main contributions are as follows:

- We introduce two novel approaches to extracting transformation invariant features from equivariant ones, while preserving most of the relevant information—Gram Pooling and Equivariant Attention;

- We empirically evaluate our approach and show that learned latent space is better structured compared to the standard convolutional autoencoder in terms of the classification accuracy and visual assessment of the manifold.

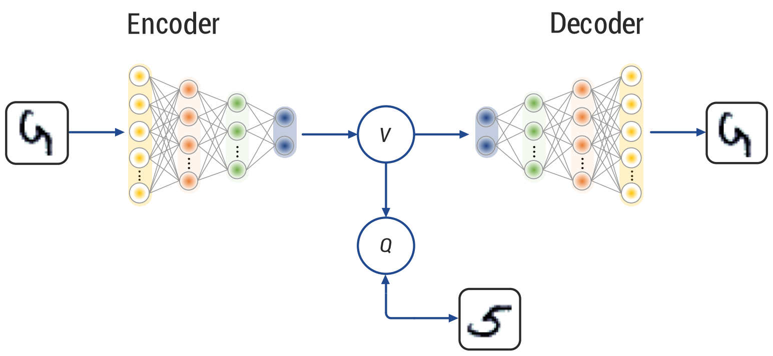

Figure 2: An overview of our model. We train an Equivariant Autoencoder that embeds input images into the latent code matrix V. Any considered transformation of an input image results in a permutation of rows of V. We then extract invariant features Q that are constant across all considered transformations.

2. Group Equivariance

In this section, we apply group convolutions (Cohen and Welling, 2016) to extract image features that are equivariant to the group D_{12} of 30° rotations and reflections along the vertical axis. We progressively build the D_{12} group, starting with 90° rotations, then adding 30° rotations and finally including reflections. Formally, the function $f(x)$ is called equivariant to a class of transformations T, if for all transformations $t \in T$ of x, its output changes in a predictable way:

$$f(tx) = t' f(x), \tag{1}$$

where t' is some transformation of the output. Similarly, we call function f invariant to this class of transformations, if all transformations $t \in T$ do not affect the output representation:

$$f(tx) = f(x). \tag{2}$$

2.1. Group p4: 90°-equivariant features

We start by building an equivariant encoder network $p4$-CNN that produces features equivariant to the group of 90° rotations. This network takes an image as an input and returns a feature matrix $V_{p4} \in \mathbb{R}^{4\times m}$. Rows of this matrix correspond to the features extracted at different 90° rotations of the initial image. The first row corresponds to features of the initial image, second—to 90° rotation of the image, third—to 180° rotation, and the fourth—to 270° rotation. If we feed a 90° rotated image to the $p4$-CNN, it produces a feature matrix with cyclically shifted rows of V_{p4}.

A naive method of building a $p4$-CNN would be to pass 90° rotated instances of an initial image through the same network and to concatenate the resulting features as rows of the matrix V_{p4}. Indeed, if we rotate the initial image by 90° and extract its features, corresponding representations in the matrix will shift one row upwards. For example, features of an initial image will now be stored in the 4-th row of V_{p4}, since this row is obtained as a 270° rotation of a 90° rotated initial image. However, in this approach, intermediate layers do not use features extracted at different angles, ignoring features from other branches. This encoder also tends to learn multiple rotated copies of the same filter, resulting in an inefficient use of learnable parameters.

Passing a rotated image through a filter is equivalent to passing an initial image through a rotated filter and then rotating the output in the opposite direction. This leads us to the idea that the number of learnable parameters can be reduced by adding rotated instances of filters. We use group convolutions (Cohen and Welling, 2016) with correlation operator as a building block of the network. This architecture encourages feature sharing between different computational branches corresponding to different rotation angles and reduces the number of learnable parameters. Correlation operator applied to an image X with filter k at shift Δ is defined as

$$[X * k](\Delta) = \sum_{y \in \mathbb{Z}^2} X[y]k[y - \Delta]. \tag{3}$$

Here, we compute a dot product between an image and a shifted filter. If we consider $\mathbb{Z}^2$ as a group of integer translations, the operation $y - \Delta$ would be an inverse shift of y by Δ. This can be written as $\Delta^{-1}(y)$, giving an equivalent representation to the equation 3:

$$[X * k](\Delta) = \sum_{y \in \mathbb{Z}^2} X[y]k[\Delta^{-1}(y)]. \tag{4}$$

For an arbitrary group G, we can extend the correlation operation by replacing transformations $\Delta \in \mathbb{Z}^2$ with elements $g \in G$:

$$[X * k](g) = \sum_{y \in \mathbb{Z}^2} X[y]k[g^{-1}(y)]. \tag{5}$$

Group $p4$ contains all combinations of integer translations and 90° rotations around the center of an image. We apply $p4$ group convolutions to the input images by computing the dot product with rotated and shifted filters. These transformations output four feature banks corresponding to different rotations of filters. By stacking group convolutions multiple times, we eliminate spatial dimensions and obtain a feature matrix V_{p4}. Notice that all layers after the first one should use the summation from Eq. 5 with respect to the group $p4$ and not to $\mathbb{Z}^2$:

$$[X * k](g) = \sum_{y \in p4} X[y]k[g^{-1}(y)]. \tag{6}$$

Figure 3 illustrates the transformation made by each layer of the $p4$-CNN. Here, we operate with four source feature banks, corresponding to different rotations of the initial image.

We apply rotated filters to different source banks and combine the results. Consider, for example, calculation of the forth target bank—270° rotated image. As one of the steps, we apply a filter bank rotated by 180° to the source bank corresponding to the 90°. These transformations combine features extracted at different angles, resulting in a more powerful embedding. Also, such layers require less parameters, since the network does not have to learn rotated filters anymore.

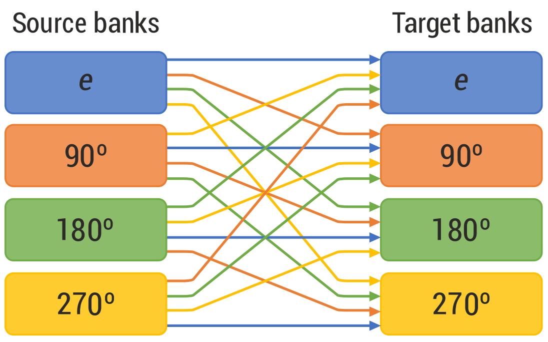

Figure 3: Single layer of $p4$-CNN. Rotated filter bank is applied to source banks, preserving the total degree of the rotation. One color corresponds to the same rotation degree.

2.2. Groups C_{12} and D_{12}—30° rotations and reflections

In this section, we build a neural network equivariant to the group C_{12} of 30° rotations, and then extend the model for the equivariance to the group D_{12} of 30° rotations and reflections along vertical axis. Note that D_{12} also contains reflections along the rotated axis. While the described approach can be used for arbitrary complex groups, we would have to interpolate the feature maps for all rotations except for the rotation multiple of 90°. This may lead to the loss of equivariance due to numerical errors, for example, for the interpolation of a 30° rotated 3×3 patch. Hoogeboom et al. (2018) solved this issue by introducing hexagonal feature maps. However, this approach requires an implementation of a new, less efficient type of convolutions. Here, we adopt the initial naive method and feed all transformed versions of an initial image through the network. We compute V_{p4} matrix for an initial image and images rotated by 30° and 60°. We then obtain an equivariant feature matrix $V_{C_{12}} \in \mathbb{R}^{12\times m}$ for a group of 30° rotations by stacking these three matrices. Any rotation that is multiple of 30° will cyclically shift rows of $V_{C_{12}}$. Finally, we apply group convolution layers to this matrix, mixing features from different rotations. As spatial dimensions at this point are already eliminated, interpolation of filters is no longer needed.

We can also add equivariance to reflections and create a group D_{12} of 30° rotations and reflections along vertical axis. We pass a mirrored image through a C_{12}-CNN to obtain a feature matrix $\overline{V}_{C_{12}}$, concatenate $\overline{V}_{C_{12}}$ with $V_{C_{12}}$, and pass the result through a few group convolution layers. These steps produce the final feature matrix $V_{D_{12}} \in \mathbb{R}^{24\times m}$. For this matrix, rotation of an initial image by 30° cyclically permutes first 12 and last 12 components of $V_{D_{12}}$ in the same direction. In the meantime, reflection along vertical axis inverts the order of components. We illustrate these permutations on a smaller group, D_4 of reflections and 90° rotations in Figure 4.

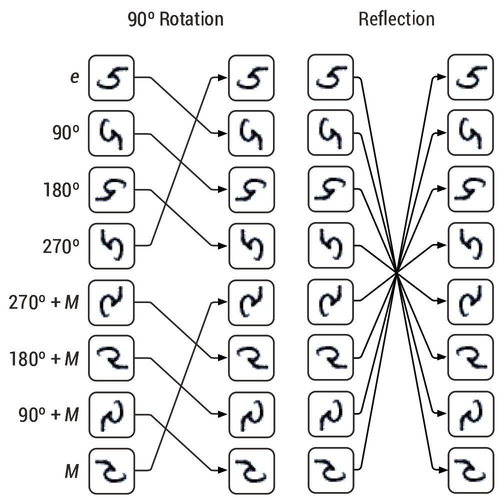

Figure 4: Permutations induced by the transformation from the group D_4. M corresponds to reflection along vertical axis. **Left:** 90° clockwise rotation. **Right:** reflection along vertical axis.

Finally, to build an autoencoder, we use a symmetric architecture and obtain initial images from $V_{D_{12}}$ with a decoder network. The final architecture is shown in Figure 5.

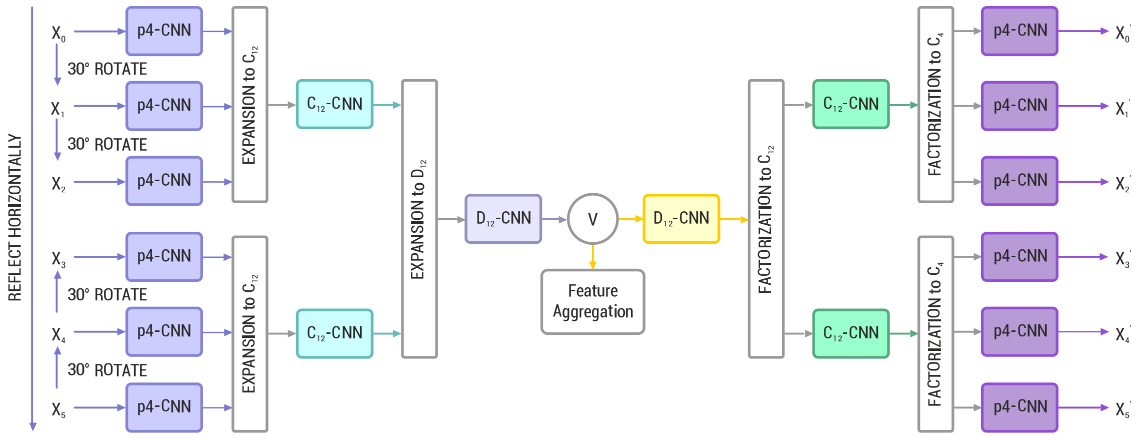

Figure 5: Architecture of an autoencoder producing D_{12}-equivariant features. Networks with the same color share the same weights.

3. Related Work

Over the recent years, many publications addressed the problem of obtaining invariant representations for 2D and 3D images. Cohen and Welling (2016) described group equivariant convolutional neural networks (G-CNNs) with G-convolution operator. By exploiting symmetries p4 (translations and 90° rotations) and p4m (p4 and reflections), convolutional layers can be used as a drop-in replacement of standard convolutions and can improve the results on image datasets without increasing the number of network parameters. Recent works by Kondor and Trivedi (2018) and Cohen et al. (2018b) generalized G-CNN framework for any compact group.

Dieleman et al. (2016) proposed four new cyclic operations (slice, roll, pool, and stack) for convolutional networks in order to achieve rotation equivariance for finite groups. Laptev et al. (2016) applied multiple convolutional Siamese networks (Koch, 2015) for each spatial transformation of an input followed by element-wise max-pooling. Gonzalez et al. (2016) performed convolutions on rotated filters with a global pooling over different orientations. Weiler et al. (2017) introduced roto-translational steerable filters on top of group convolutions. Novel SFCNN architecture with weight-sharing between learnable filters achieved state-of-the-art results on rotated MNIST and ISBI 2012 EM segmentation challenge.

Harmonic Networks, presented by Worrall et al. (2017b), use circular harmonics instead of convolutions, achieving patch-wise continuous 360° rotational equivariance. By applying encoder-decoder architecture with feature transform layer, Worrall et al. (2017a) was able to explicitly disentangle transformation features on the hidden representations.

Following previous works on group convolutions, Cohen et al. (2018a) designed spherical CNN architecture that can be applied to spherical signals and achieves equivariance on SO(3) manifold. 3D G-CNN model by Winkels and Cohen (2018) showed significant improvement over regular convolutions in the task of pulmonary nodule detection. Another extension of group convolutions for the hexagonal grid (Hoogeboom et al., 2018) was applied to the classification task on AID aerial dataset. Bekkers et al. (2018) proposed using SE(2) group convolutional layers for medical image analysis and obtained state-of-the-art results on three medical imaging tasks in histopathology, retinal imaging, and electron microscopy.

The proposed equivariant attention method distantly resembles a Spatial Transformer (Jaderberg et al., 2015) capable of learning standard image representations. Here, however, we produce a distribution over all possible transformations which allows the attention to learn in a more stable way.

4. Extracting invariant features

In this section, we introduce two novel approaches to extracting invariant features from equivariant ones obtained in the previous section. We validate the predictive quality of extracted features on the classification task. Trained fully-connected network shows improvement in prediction performance compared to the embeddings of a pure convolutional autoencoder (see section 5). To achieve a good performance, we try to retain as much information about the initial image as possible, while removing transformation-relevant information.

4.1. Gram Pooling

Consider a group of transformations T and Equivariant Encoder that maps an input image to an embedding matrix $V \in \mathbb{R}^{n\times m}$. Any transformation $t \in T$ permutes rows of V. With a slight abuse of notation, we denote corresponding permutations as elements of T.

The simplest way to extract invariant features from the matrix V is to average its rows: $w_{\text{mean}} = \frac{1}{n}\sum_{i=1}^{n} v_i$. Since permutation of rows will only change the summation order, w_{mean} is an invariant vector. However, this aggregation loses a lot of information about the initial image. While the embedding matrix V obtained from an encoder-decoder architecture contains all information about the input object, w_{mean} cannot be used to reconstruct an initial image, even if we specify transformation parameters. In this section, we introduce a *Gram Pooling* method that extracts an invariant embedding, keeping more information about an image than w_{mean}.

The proposed approach is based on the notion of Gram matrices. The Gram matrix of vectors $\{v_i\}_{i=1}^{m}$ is an $m \times m$ matrix G of pairwise scalar products: $G_{ij} = \langle v_i, v_j \rangle$. A Gram matrix $G = V^T V$ of columns of V forms an invariant set of features. If we denote columns of V as v_i, a scalar product $\langle v_i, v_j \rangle = \sum_{k=1}^{n} V_{ki} V_{kj}$ remains constant after any transformation from T, since a permutation only changes the order of the summation. Note that if the number of features is smaller than the size of the group, the initial set of vectors can be reconstructed up to isometry from its Gram matrix using Cholesky decomposition. The Gram matrix retains much more information about the initial image than simple averaging.

However, reconstruction up to isometry still loses some relevant information. To find other invariant features, we look for them in a class of $q_{ij}^{\pi} = \langle u_i, \pi u_j \rangle$, where π is a permutation of vector components. We denote a set of permutations that generate invariant features as P. Note that this set is always non-empty, since features q_{ij}^{e} produce a Gram matrix. Formally, P can be defined as:

$$P = \{\pi \in S \mid \langle u, \pi v \rangle = \langle tu, \pi t v \rangle, \forall t \in T, \forall u, v \in \mathbb{R}^n\}, \tag{7}$$

where S is a set of all operators that permute vector elements. We now show that P consists of all elements that commute with T, i.e. $P = \{\pi \in S \mid \pi t = t\pi, \, \forall t \in T\}$. This is true, since if $\langle tv, \pi t u \rangle = \langle v, \pi u \rangle$, then $\langle v, t^{-1}\pi t u \rangle = \langle v, \pi u \rangle$ and $\langle v, t^{-1}\pi t u - \pi u \rangle = 0$. Since this equality holds for all vectors u and v, $t^{-1}\pi t = \pi$, which implies $t\pi = \pi t$. Note that it is sufficient to check that π commutes with all elements of a generating set of T. For the D_{12} group we have to check that π commutes with both 30° rotation and reflection. Also, note that P is a group, since inverse and composition of commuting elements also commutes. The constructed set P of all commuting elements is also known as a centralizer of a subgroup T of a group S.

Finding P in general requires to brute force over all permutations. However, if T is an abelian (commutative) group, any permutation of rows induced by T satisfies Eq. 7, meaning that $T \subset P$.

The D_4 group, generated by 90° rotations and reflections along vertical axes is not abelian. However, a non-identical permutation $\pi \in P$ exists. D_4 contains $n = 8$ elements. In a cycle notation, rotation performs permutation $R = (1234)(5678)$, while reflection performs $M = (87654321)$. Permutation $\pi = (15)(26)(37)(48)$ induces invariant features, since it commutes with both rotation and reflection.

An important note about the constructed set of features is that it is not required to store the whole $m \times m$ Gram matrix, since its elements are linearly dependent when $n < m$. In this case, it is sufficient to store elements of the first n diagonals.

Finally, we notice that diagonal features of constructed matrices are invariant for some permutations besides P. Because of this, we also add diagonal features $\widetilde{q}_{i,i}^{\pi} = \langle u_i, \pi u_i \rangle$ for permutations $\pi \in P'$, where

$$P' = \{\pi \in S \mid \langle u, \pi u \rangle = \langle tu, \pi tu \rangle, \forall t \in T, \forall u \in \mathbb{R}^n\} \tag{8}$$

For the group D_4, for example, permutation R, corresponding to the 90° rotation, lies in P' as well as its powers R^2 and R^3. In practice, sets P and P' can be very large, but the performance of predictive models already saturates when a few elements of P or P' are added. The final set of features consists of two parts, corresponding to P and P':

$$q^{\pi}_{i,j} = \langle u_i, u_j \rangle, \forall \pi \in P, i \in [1, n], j \in [i, \min(i + m, n)] \tag{9}$$

$$\widetilde{q}^{\pi}_{i,i} = \langle u_i, u_i \rangle, \forall \pi \in P', i \in [1, n] \tag{10}$$

4.2. Equivariant Attention

In this section, we propose an alternative attention-based approach to extracting invariant features. Consider equivariant features $V \in \mathbb{R}^{n \times m}$ such that any transformation $t \in T$ permutes rows of V with a permutation π_t, giving $\pi_t(V)$.

Attention mechanism (Vaswani et al., 2017; Xu et al., 2015) encourages neural network to focus on relevant parts of the data. For example, a neural network with attention applied to image pixels learns to focus on the object, ignoring the background. Attention is usually organized as a neural network that outputs a distribution over a set of objects. Representations of these objects are then averaged according to the obtained weights, resulting in a larger contribution from objects with larger weights. In our model, we apply attention to different permutations of V, i.e, we use $\{\pi_t(V)\}_{t \in T}$ as objects of attention. Attention network outputs weight w_{π_t} for every transformation $t \in T$. These weights are non-negative and sum up to one. We achieve this by passing the output of the network through a Softmax function. Permutations are then averaged, resulting in a feature matrix Q given as:

$$Q = \sum_{t \in T} w_{t^{-1}} \pi_t(V) \in \mathbb{R}^{n \times m}. \tag{11}$$

Attention weights should be consistent with transformations T and change accordingly when an input image is transformed. This is achieved when the attention network is also equivariant. As we show further, the matrix Q forms invariant features.

Note that permutations $\pi_t(V)$ in Eq. 11 are taken with the weights $w_{t^{-1}}$ corresponding to inverse permutations. If we transform an input image with $t' \in T$, components of w and rows of V are permuted with $\pi_{t'}$, resulting in

$$\sum_{t \in T} \left[\pi_{t'}(w)\right]_{t^{-1}} \pi_t\left(\pi_{t'}(V)\right) = \sum_{t \in T} \left[\pi_{t'}(w)\right]_{t^{-1}} \pi_{tt'}(V). \tag{12}$$

Note that $\left[\pi_{t'}(w)\right]_{t^{-1}}$ is the weight that we get after first permuting components of w and then taking a weight at position t^{-1}. The weight that ended up at position t^{-1} after permutation $\pi_{t'}$ was stored at position $[\pi^{-1}_{t'}]_{t^{-1}}$ before permutation. This results in the following equation:

$$\left[\pi_{t'}(w)\right]_{t^{-1}} = w_{(t')^{-1}t^{-1}} = w_{(tt')^{-1}}. \tag{13}$$

Combining it all together, we get

$$\begin{aligned}\sum_{t\in T}\left[\pi_{t'}\left(w\right)\right]_{t^{-1}}\pi_{tt'}\left(V\right) &= \sum_{t\in T} w_{(tt')^{-1}}\pi_{tt'}\left(V\right)\\ &= \sum_{t\in T} w_{t^{-1}}\pi_{t}\left(V\right)\\ &= Q,\end{aligned} \tag{14}$$

indicating that the total sum is invariant. Having weights w and the invariant feature matrix Q, we can invert the linear transformation in Eq. 11 to reconstruct initial equivariant weights V.

When attention weights w are concentrated in one permutation (one-hot), i.e. only one permutation of V is selected, attention weights can be inverted using

$$V = \sum_{t\in T} w_t \pi_t(Q). \tag{15}$$

Notice that when w_t is close to one-hot, Eq. 15 holds approximately. A case of one-hot attention weights has an important theoretical advantage, since the selected permutation corresponds to a standardizing transformation. This results in decomposition of an equivariant feature matrix into two entities: invariant feature matrix and transformation required to standardize the image. Such decomposition extracts all invariant information about the initial image, removing only transformation-relevant part. To promote concentrated weights, we add an entropy regularizer to the model:

$$\min \mathcal{H}(w) = -\sum_{t\in T} w_t \log w_t. \tag{16}$$

Finally, we apply Equivariant Attention to build an autoencoder, shown in Figure 6. The model first extracts equivariant features using an Equivariant Encoder and computes equivariant attention weights w. These weights are used to obtain an invariant matrix using Eq. 11. Since the embedding no longer contains transformation-relevant information, we can further compress the embedding using a small fully-connected neural network, resulting in a final set of invariant features. During the decoding procedure, a symmetric architecture is used to reconstruct an initial image. We use Eq. 15 to invert equivariant attention, reusing attention weights from the encoder.

5. Experiments

In this section, we compare the proposed models with a standard autoencoder in terms of classification performance on embeddings and visual assessment of the manifold. We validate our models on augmented MNIST and Plankton datasets.

5.1. Augmented MNIST

For the experiment on the MNIST, we constructed an augmented version of the original data. MNIST is a standard image classification dataset with 70,000 handwritten digits. We augmented images of the initial dataset with random rotations and reflections.

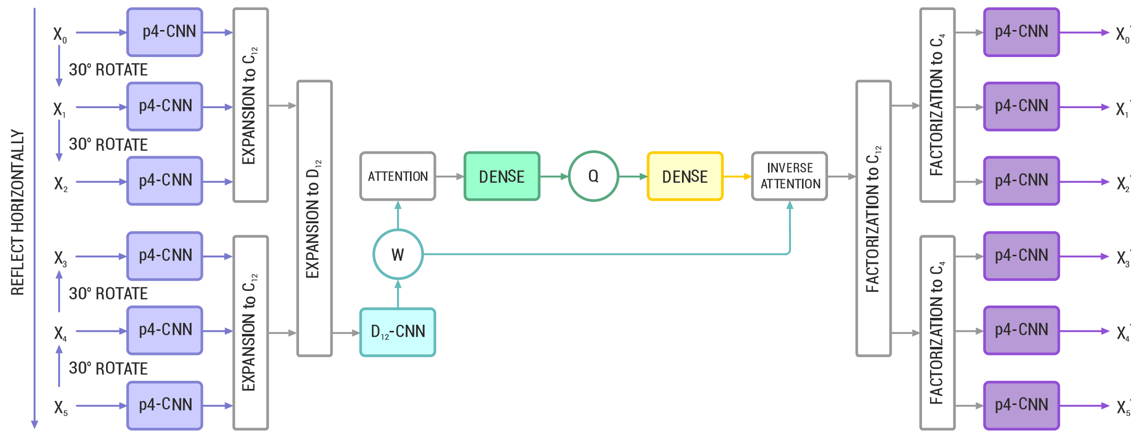

Figure 6: An autoencoder with Equivariant Attention. Networks with the same color share the same weights.

We trained an unsupervised Equivariant Autoencoder and extracted an invariant representation from the embeddings. Then, we trained a multinomial logistic regression and a fully-connected neural network to predict a class label from these features. When the classifier has a higher performance, we assume that similar images are better localized in the latent space.

We compared our model with two baselines. As a first trivial baseline, we built a convolutional autoencoder and trained a classifier on its embeddings. These embeddings are not invariant and may contain irrelevant information about rotations and reflections. As a second baseline, we used a simple mean pooling to extract invariant features from equivariant embeddings, instead of using Gram Pooling or Equivariant Attention. This method by design loses some relevant information about the initial image. For each model, we tuned hyperparameters independently, based on the performance on the validation set. Experimental results are provided in Table 1.

Table 1: Classification accuracy on the extracted embeddings for the augmented MNIST dataset

Model	Group	Accuracy, %	
		Logistic regression	Neural network
Autoencoder	—	47.8	58.3
Average Pooling	D_4	58.2	60.8
Average Pooling	D_{12}	80.7	82.5
Equivariant Attention	D_4	81.6	83.8
Equivariant Attention	D_{12}	72.6	76.5
Gram Pooling	D_4	65.2	67.5
Gram Pooling	D_{12}	**87.9**	**88.8**

5.2. Plankton dataset

For experiments on the Plankton data, we used a publicly available part of the Plankton dataset from the National Data Bowl competition 2015 Cowen et al. (2015). The data contain approximately 160,000 images of small sea creatures from 127 different classes. However, labels are available only for approximately 30,000 images. As a preprocessing step, we padded all images to have a square shape and rescaled them to the resolution of 32×32 pixels. The type of the plankton present on the image is invariant to rotations and reflections. Since the goal of this paper is not to achieve the state of the art classification results, but to introduce a general framework for extracting invariant features, we restrict ourselves to rather shallow models and ignore important features such as the original image aspect. Classification results are provided in Table 2.

Table 2: Classification accuracy on the extracted embeddings for the Plankton dataset

Model	Group	Accuracy, %	
		Logistic regression	Neural network
Autoencoder	—	44.3	53.3
Average Pooling	D_4	48.0	56.1
Average Pooling	D_{12}	53.9	58.3
Equivariant Attention	D_4	49.1	58.0
Equivariant Attention	D_{12}	49.5	57.8
Gram Pooling	D_4	55.2	57.8
Gram Pooling	D_{12}	**55.7**	**62.2**

Provided results suggest that forcing features to be invariant to rotation and reflection improves the classification accuracy. Also, it is profitable to increase the number of angles in a group of transformations. Compared to simple mean pooling, both Equivariant Attention and Gram Pooling methods show better performance, suggesting that extraction of a larger number of invariant features helps the classifier. Comparing Gram Pooling and Equivariant Attention, we see that Gram Pooling, in general, outperforms Equivariant Attention. We hypothesize that it is due to a more complex optimization problem stated for the attention. However, the Equivariant Attention approach guarantees to keep all invariant information in extracted features if attention weights are one-hot, which may be profitable for more complex classifiers.

5.3. Manifold structure

In this experiment, we visually assess the quality of the latent space, extracted by the best model (Gram Pooling, D_{12}) for the Plankton dataset. We show Principal Components and t-SNE Maaten and Hinton (2008) embeddings in Figure 7. Results suggest that the proposed method brings objects from the same class closer together. For example, in the t-SNE plot of the simple Autoencoder embeddings, the blue class was separated into 5 clusters, corresponding to different angles of the similar image. Invariant embeddings produced by the Gram Pooling, however, formed a single cluster with these images.

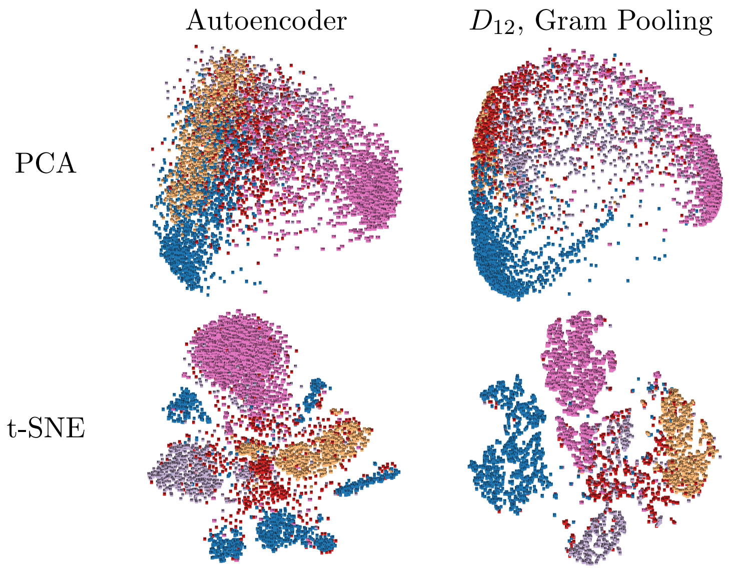

Figure 7: Manifold of the Plankton dataset extracted by the autoencoder and by the Equivariant Autoencoder with invariant features computed using Gram Pooling. Same colors denote the same classes.

5.4. Intrinsic dimensionality

In this experiment, we investigate the intrinsic dimensionality of the embeddings, i.e., the number of dimensions that are effectively used. To do so, we computed invariant embeddings of the augmented MNIST images and calculated the explained variance ratio of the first principal components. All considered models had almost perfect reconstruction quality, indicating that the equivariant embedding kept almost all information about the initial image. Also, Equivariant Attention models produced attention weights with almost zero entropy, indicating that invariant features also kept all the required information. Results are provided in Figure 8. The highest compression is achieved for the Gram Pooling approach, where most of the features turn out to be linearly dependent. Comparing D_4 and D_{12} curves, we notice that when information about finer rotations is eliminated, much less information is retained in the embedding. Finally, comparing these curves for the Equivariant Attention, combined with the results from Table 1, we conclude that the Equivariant Attention was not able to adapt to the group D_{12}. However, the Equivariant Attention for the D_4 group uses much less dimensions than the corresponding model with Gram Pooling, even though the latter could miss some information about the initial image.

6. Conclusion

In this paper, we introduced an Equivariant Autoencoder that extracts equivariant features from images using group convolutions. We also proposed two approaches for invariant feature extraction from equivariant embeddings—Gram Pooling and Equivariant Attention. Comparing these two methods, we noticed that Gram Pooling based method shows better classification performance. However, this approach does not necessarily preserve all rele-

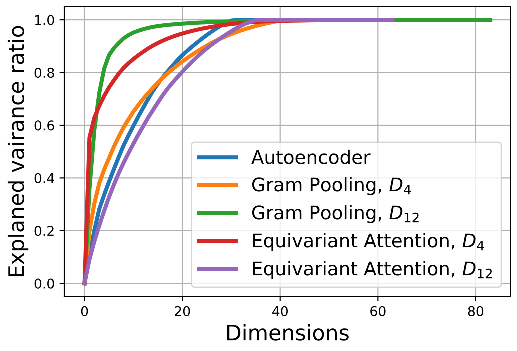

Figure 8: Explained variance ratio of the first principal components of the embeddings.

vant information about the image. Equivariant Attention, on the other hand, decomposes features into transformation-relevant and irrelevant parts. The second part is guaranteed to keep all invariant features from the image if attention weights are one-hot. The proposed approach can be further used for classification with bigger groups without increasing the number of learnable parameters significantly.

In the future work, we want to compare the invariant latent space of the proposed Equivariant Attention method with Variational (Kingma and Welling, 2013) and Adversarial (Makhzani et al., 2015) Autoencoders, since both architectures force the embedding to form a structured manifold. For the case of generative models, we can assess how invariant features affect the properties of generated samples. As we noted in the paper, the Equivariant Attention has some similarities with the Spatial Transformer (Jaderberg et al., 2015) architecture. These models can also be compared in terms of their ability to learn invariant image representations.

References

Erik Johannes Bekkers, Maxime W. Lafarge, Mitko Veta, Koen A. J. Eppenhof, Josien P. W. Pluim, and Remco Duits. Roto-translation covariant convolutional networks for medical image analysis. *CoRR*, abs/1804.03393, 2018.

Taco Cohen and Max Welling. Group equivariant convolutional networks. In *ICML*, 2016.

Taco Cohen, Mario Geiger, Jonas Köhler, and Max Welling. Spherical cnns. *CoRR*, abs/1801.10130, 2018a.

Taco Cohen, Mario Geiger, and Maurice Weiler. Intertwiners between induced representations (with applications to the theory of equivariant neural networks). *CoRR*, abs/1803.10743, 2018b.

Robert K. Cowen, S. Sponaugle, K.L. Robinson, J. Luo, Oregon State University, and Hatfield Marine Science Center. Planktonset 1.0: Plankton imagery data collected from f.g. walton smith in straits of florida from 2014-06-03 to 2014-06-06 and used in the 2015 national data science bowl (ncei accession 0127422), 2015.

Sander Dieleman, Jeffrey De Fauw, and Koray Kavukcuoglu. Exploiting cyclic symmetry in convolutional neural networks. In *ICML*, 2016.

Diego Marcos Gonzalez, Michele Volpi, and Devis Tuia. Learning rotation invariant convolutional filters for texture classification. *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2012–2017, 2016.

Emiel Hoogeboom, Jorn W. T. Peters, Taco Cohen, and Max Welling. Hexaconv. *CoRR*, abs/1803.02108, 2018.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in neural information processing systems*, pages 2017–2025, 2015.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

G. R. Koch. Siamese neural networks for one-shot image recognition. 2015.

Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. *CoRR*, abs/1802.03690, 2018.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60:84–90, 2012.

Denis Kuzminykh, Daniil Polykovskiy, Artur Kadurin, Alexander Zhebrak, Ivan Baskov, Sergey Nikolenko, Rim Shayakhmetov, and Alex Zhavoronkov. 3d molecular representations based on the wave transform for convolutional neural networks. *Molecular pharmaceutics*, 2018.

Dmitry Laptev, Nikolay Savinov, Joachim M. Buhmann, and Marc Pollefeys. Ti-pooling: Transformation-invariant pooling for feature learning in convolutional neural networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 289–297, 2016.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian J. Goodfellow. Adversarial autoencoders. *CoRR*, abs/1511.05644, 2015.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 6000–6010, 2017.

Maurice Weiler, Fred A. Hamprecht, and Martin Storath. Learning steerable filters for rotation equivariant cnns. *CoRR*, abs/1711.07289, 2017.

Marysia Winkels and Taco Cohen. 3d g-cnns for pulmonary nodule detection. 2018.

Daniel Worrall, Stephan Garbin, Daniyar Turmukhambetov, and Gabriel Brostow. Interpretable transformations with encoder-decoder networks. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5737–5746, 2017a.

Daniel E. Worrall, Stephan J. Garbin, Daniyar Turmukhambetov, and Gabriel J. Brostow. Harmonic networks: Deep translation and rotation equivariance. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7168–7177, 2017b.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, pages 2048–2057, 2015.