Restore Translation Using Equivariant Neural Networks

Yihan Wang[1,3], Lijia Yu[2,3], and Xiao-Shan Gao[1,3(✉)]

[1] Academy of Mathematics and Systems Science, Chinese Academy of Sciences,
Beijing, China
`xgao@mmrc.iss.ac.cn`
[2] SKLCS, Institute of Software, Chinese Academy of Sciences, Beijing, China
[3] University of Chinese Academy of Sciences, Beijing, China

Abstract. Invariance to spatial transformations such as translations is a desirable property and a basic design principle for classification neural networks. However, the commonly used convolutional neural networks (CNNs) are actually very sensitive to even small translations. There exist vast works to achieve exact or approximate transformation invariance by designing transformation-invariant models or assessing the transformations. These works usually make changes to the standard CNNs and harm the performance on standard datasets. In this paper, rather than modifying the classifier, we propose a pre-classifier restorer to recover translated inputs to the original ones which will be fed into any classifier for the same dataset. The restorer is based on a theoretical result which gives a sufficient and necessary condition for an affine operator to be translational equivariant on a tensor space.

Keywords: Equivariant network · Translation restorer

1 Introduction

Deep convolutional neural networks (CNNs) had outperformed humans in many computer vision tasks [9,12]. One of the key ideas in designing the CNNs is that the convolution layer is equivariant with respect to translations, which was emphasized both in the earlier work [5] and the modern CNN [12]. However, the commonly used components, such as pooling [7] and dropout [19,20], which help the network to extract features and generalize, actually make CNNs not equivariant to even small translations, as pointed out in [1,3]. As a comprehensive evaluation, Fig. 1 shows that two classification CNNs suffer the accuracy reductions of more than 11% and 59% respectively on CIFAR-10 and MNIST, when the inputs are horizontally and vertically translated at most 3 pixels.

Invariance to spatial transformations, including translations, rotations and scaling, is a desirable property for classification neural networks and the past few decades have witnessed thriving explorations on this topic. In general, there exist three ways to achieve exact or approximate invariance. The first is to design transformation-invariant neural network structures [2,6,8,10,15,16,18,21].

MNIST

Vertical \ Horizontal	-3	-2	-1	0	1	2	3
3	-34.1	-13.24	-10.49	-11.67	-16.28	-26.4	-41.21
2	-27.31	-6.11	-3.27	-3.31	-5.05	-10.65	-22.27
1	-22.88	-2.94	-0.82	-0.53	-1.2	-3.52	-10.09
0	-22.85	-2.3	-0.39	0	-0.32	-1.48	-5.2
-1	-28.85	-4.88	-1.79	-0.81	-0.95	-2.34	-6.28
-2	-42.35	-14.06	-7.51	-5.26	-5.54	-7.91	-13.68
-3	-59.89	-31.01	-21.66	-17.74	-17.64	-21.03	-27.78

CIFAR-10

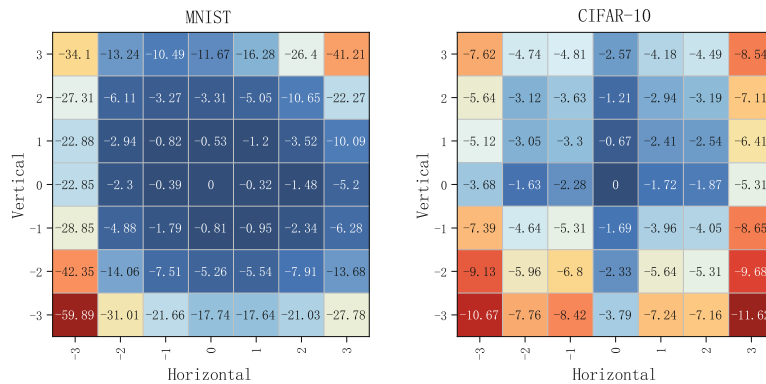| Vertical \ Horizontal	-3	-2	-1	0	1	2	3
3	-7.62	-4.74	-4.81	-2.57	-4.18	-4.49	-8.54
2	-5.64	-3.12	-3.63	-1.21	-2.94	-3.19	-7.11
1	-5.12	-3.05	-3.3	-0.67	-2.41	-2.54	-6.41
0	-3.68	-1.63	-2.28	0	-1.72	-1.87	-5.31
-1	-7.39	-4.64	-5.31	-1.69	-3.96	-4.05	-8.65
-2	-9.13	-5.96	-6.8	-2.33	-5.64	-5.31	-9.68
-3	-10.67	-7.76	-8.42	-3.79	-7.24	-7.16	-11.62

Fig. 1. The accuracy reduction after vertical and horizontal translations. The translation scope is $[-3, 3]$ pixels. Left: LeNet-5 on MNIST; Right: VGG-16 on CIFAR-10.

The second is to assess and approximate transformations via a learnable module [4,11] and then use the approximation to reduce the transformed inputs to "standard" ones. The third is data augmentation [1,3,17] by adding various transformations of the samples in the original dataset.

Those ad-hoc architectures to achieve invariance often bring extra parameters but harm the network performance on standard datasets. Moreover, the various designs with different purposes are not compatible with each other. Data augmentation is not a scalable method since the invariance that benefits from a certain augmentation protocol does not generalize to other transformations [1]. Including learnable modules such as the Spatial Transformer, all the three approaches require training the classifier from scratch and fail to endow existing trained networks with some invariance. It was indicated in [1] that "the problem of insuring invariance to small image transformations in neural networks while preserving high accuracy remains unsolved."

In this paper, rather than designing any in-classifier component to make the classifier invariant to some transformation, we propose a pre-classifier restorer to restore translated inputs to the original ones. The invariance is achieved by feeding the restored inputs into any following classifier. Our restorer depends only on the dataset instead of the classifier. Namely, the training processes of the restore and classifier are separate and a restore is universal to any classifier trained on the same dataset.

We split the whole restoration into two stages, transformation estimation and inverse transformation, see Fig. 2. In the first stage, we expect that standard inputs lead to standard outputs and the outputs of translated inputs reflect the translations. Naturally, what we need is a strictly translation-equivariant neural network. In Sect. 3, we investigate at the theoretical aspect the sufficient and necessary condition to construct a strictly equivariant affine operator on a tensor space. The condition results in *the circular filters*, see Definition 4, as the fundamental module to a strictly translation-equivariant neural network. We

give the canonical architecture of translation-equivariant networks, see Eq. (2). In Sect. 4, details of the restorer are presented. We define a translation estimator, the core component of a restorer, as a strictly translation-equivariant neural network that guarantees the first component of every output on a dataset to be the largest component, see Definition 5. For a translated input, due to the strict equivariance, the largest component of the output reflects the translation. Thus we can translate it inversely in the second stage and obtain the original image. Though the restorer is independent of the following classifier, it indeed depends on the dataset. Given a dataset satisfying some reasonable conditions, i.e. *an aperiodic finite dataset*, see Definition 6, we prove the existence of a translation estimator, i.e. a restorer, with the canonical architecture for this dataset. Moreover, rotations can be viewed as translations by converting the Cartesian coordinates to polar coordinates and the rotation restorer arises in a similar way.

In Sect. 5, the experiments on MNIST, 3D-MNIST and CIFAR-10 show that our restorers not only visually restore the translated inputs but also largely eliminate the accuracy reduction phenomenon.

2   Related Works

As a generalization of convolutional neural networks, group-equivariant convolutional neural networks [2,6] exploited symmetries to endow networks with invariance to some group actions, such as the combination of translations and rotations by certain angles. The warped convolutions [10] converted some other spatial transformations into translations and thus obtain equivariance to these spatial transformations. Scale-invariance [8,15,21] was injected into networks by some ad-hoc components. Random transformations [16] of feature maps were introduced in order to prevent the dependencies of network outputs on specific poses of inputs. Similarly, probabilistic max pooling [18] of the hidden units over the set of transformations improved the invariance of networks in unsupervised learning. Moreover, local covariant feature detecting methods [14,22] were proposed to address the problem of extracting viewpoint invariant features from images.

Another approach to achieving invariance is "shiftable" down-sampling [13], in which any original pixel can be linearly interpolated from the pixels on the sampling grid. This "shiftable" down-sampling exists if and only if the sampling frequency is at least twice the highest frequency of the unsampled signal.

The Spatial Transformer [4,11], as a learnable module, produces a predictive transformation for each input image and then spatially transforms the input to a canonical pose to simplify the inference in the subsequent layers. Our restorers give input-specific transformations as well and adjust the input to alleviate the poor invariance of the following classifiers. Although the Spatial Transformers and our restorer are both learnable modules, the training of the former depends not only on data but also on the subsequent layers, while the latter are independent of the subsequent classifiers.

3   Equivariant Neural Networks

Though objects in nature have continuous properties, once captured and converted to digital signals, these properties are represented by real tensors. In this section, we study the equivariance of operators on a tensor space.

3.1   Equivariance in Tensor Space

Assume that a map $\tilde{x} : \mathbb{R}^d \to \mathbb{D}$ stands for a property of some d-dimensional object where $\mathbb{D} \subseteq \mathbb{R}$. Sampling $\tilde{x}$ over a $(n_1, n_2, \cdots, n_d)$-grid results in a tensor x in a tensor space

$$\mathcal{H} := \mathbb{D}^{n_1} \otimes \mathbb{D}^{n_2} \otimes \cdots \otimes \mathbb{D}^{n_d}. \tag{1}$$

We denote $[n] = [0, 1, \ldots, n-1]$ for $n \in \mathbb{Z}_+$ and assume $k \bmod n \in [n]$ for $k \in \mathbb{Z}$. For an index $I = (i_1, i_2, \cdots, i_d) \in \prod_{i=1}^{d} [n_i]$ and $x \in \mathcal{H}$, denote $x[I]$ to be the element of x with subscript $(i_1, i_2, \cdots, i_d)$. For convenience, we extend the index of $\mathcal{H}$ to $I = (i_1, i_2, \cdots, i_d) \in \mathbb{Z}^d$ by defining

$$x[I] = x[i_1 \bmod n_1, \cdots, i_d \bmod n_d].$$

Definition 1 (Translation). *A translation $T^M : \mathcal{H} \to \mathcal{H}$ with $M \in \mathbb{Z}^d$ is an invertible linear operator such that for all $I \in \mathbb{Z}^d$ and $x \in \mathcal{H}$,*

$$T^M(x)[I] = x[I - M].$$

The inverse of T^M is clearly T^{-M}.

Definition 2 (Equivariance). *A map $w : \mathcal{H} \to \mathcal{H}$ is called* equivariant *with respect to translations if for all $x \in \mathcal{H}$ and $M \in \mathbb{Z}^d$,*

$$T^M(w(x)) = w(T^M(x)).$$

Definition 3 (Vectorization). *A tensor x can be vectorized to $X \in \overrightarrow{\mathcal{H}} = \mathbb{D}^N$ with $N = n_1 n_2 \cdots n_d$ such that*

$$X(\delta(I)) := x[I],$$

where $\delta(I) := (i_1 \bmod n_1) n_2 n_3 \cdots n_d + (i_2 \bmod n_2) n_3 n_4 \cdots n_d + \cdots + (i_d \bmod n_d)$, and we denote $X = \overrightarrow{x}$. Moreover, the translation T^M is vectorized as $T^M(X) := \overrightarrow{T^M(x)}$.

3.2   Equivariant Operators

When $\mathbb{D} = \mathbb{R}$, the tensor space $\mathcal{H}$ is a Hilbert space by defining the inner product as $x \cdot z := \overrightarrow{x} \cdot \overrightarrow{z}$ which is the inner product in vector space $\overrightarrow{\mathcal{H}}$. In the rest of this section, we assume $\mathbb{D} = \mathbb{R}$.

According to Reize's representation theorem, there is a bijection between continuous linear operator space and tensor space. That is, a continuous linear

operator $v : \mathcal{H} \to \mathbb{R}$ can be viewed as a tensor $v \in \mathcal{H}$ satisfying $v(x) = v \cdot x$. Now we can translate v by T^M and obtain $T^M(v) : \mathcal{H} \to \mathbb{R}$ such that $T^M(v)(x) = T^M(v) \cdot x$.

We consider a continuous linear operator $w : \mathcal{H} \to \mathcal{H}$. For $I \in \mathbb{Z}^d$ and $x \in \mathcal{H}$, denote $w_I(x) = w(x)[I]$. Then $w_I : \mathcal{H} \to \mathbb{R}$ is a continuous linear operator. An *affine operator* $\alpha : \mathcal{H} \to \mathcal{H}$ differs from a continuous linear operator w by a *bias tensor* c such that $\alpha(x) = w(x) + c$ for all $x \in \mathcal{H}$.

Theorem 1. *Let $\alpha(x) = w(x) + c : \mathcal{H} \to \mathcal{H}$ be an affine operator. Then, α is equivariant with respect to translations if and only if for all $M \in \mathbb{Z}^d$,*

$$w_M = T^M(w_\mathbf{0}) \text{ and } c \propto \mathbf{1},$$

where $\mathbf{0}$ is the zero vector in $\mathbb{Z}^d$ and $c \propto \mathbf{1}$ means that c is a constant tensor, *that is, all of its entries are the same.*

Proof of Theorem 1 is given in Appendix A. Recall that $\overrightarrow{\mathcal{H}} = \mathbb{R}^N$ is the vectorization of $\mathcal{H}$ and T^M also translates vectors in $\overrightarrow{H}$. Each continuous linear operator on $\mathcal{H}$ corresponds to a matrix in $\mathbb{R}^{N \times N}$ and each bias operator corresponds to a vector in $\mathbb{R}^N$. Now we consider the translation equivariance in vector space.

Definition 4 (Circular filter). *Let $W = (W_0, W_1, \cdots, W_{N-1})^T$ be a matrix in $\mathbb{R}^{N \times N}$. W is called a* circular filter *if $W_{\delta(M)} = T^M(W_0)$ for all $M \in \mathbb{Z}^d$.*

As the vector version of Theorem 1, we have

Corollary 1. *Let $A : \mathbb{R}^N \to \mathbb{R}^N$ be an affine transformation such that*

$$A(X) = W \cdot X + C,$$

in which $W \in \mathbb{R}^{N \times N}$, $C \in \mathbb{R}^N$. Then, A is equivariant with respect to translations in the sense that for all $M \in \mathbb{Z}^d$

$$A(T^M(X)) = T^M(A(X))$$

if and only if W is a circular filter and $C \propto \mathbf{1}$.

This affine transformation is very similar to the commonly used convolutional layers [5,12] in terms of shared parameters and similar convolutional operations. But the strict equivariance calls for the same in-size and out-size, and circular convolutions, which are usually violated by CNNs.

3.3   Equivariant Neural Networks

To compose a strictly translation-equivariant network, the spatial sizes of the input and output in each layer must be the same and thus down-samplings are not allowed. Though Corollary 1 provides the fundamental component of a

strictly translation-equivariant network, different compositions of this component lead to various equivariant networks. Here we give the *canonical architecture*. We construct the strictly translation-equivariant network F with L layers as

$$F(X) = F_L \circ F_{L-1} \circ \cdots \circ F_1(X). \tag{2}$$

The l-the layer F_l has n_l channels and for an input $X \in \mathbb{R}^{n_{l-1} \times N}$ we have

$$F_l(X) = \sigma(W[l] \cdot X + C[l]) \in \mathbb{R}^{n_l \times N}, \tag{3}$$

where

$$
\begin{aligned}
W[l] &= (W^1[l], \cdots, W^{n_l}[l]) \in \mathbb{R}^{n_l \times n_{l-1} \times N \times N}, \\
C[l] &= (C^1[l] \cdot \mathbf{1}, \cdots, C^{n_l}[l] \cdot \mathbf{1}), \\
W^k[l] &= (W^{k,1}[l], \cdots, W^{k,n_{l-1}}[l]) \in \mathbb{R}^{n_{l-1} \times N \times N}, \\
C^k[l] &= (C^{k,1}[l], \cdots, C^{k,n_{l-1}}[l]) \in \mathbb{R}^{n_{l-1}},
\end{aligned}
$$

σ is the activation, $W^{k,r}[l] \in \mathbb{R}^{N \times N}$ are circular filters, $C^{k,r}[l] \in \mathbb{R}$ are constant biases for $k = 1, \cdots, n_l$ and $r = 1, \cdots, n_{l-1}$, the $\cdot$ denotes the inner product and $\mathbf{1}$ is the vector whose all components are 1.

4   Translation Restorer

4.1   Method

In Sect. 3.3, we propose a strictly equivariant neural network architecture (2) such that any translation on the input will be reflected in the output. Generally speaking, once the outputs of an equivariant network on a dataset have some spatial structure, this structure shifts consistently as the input shifts. Thus, the translation parameter of a shifted input can be solved from its output. Finally, we can restore the input via the inverse translation. Figure 2 shows how a restorer works.

The whole restoration process splits into two stages, translation estimation and inverse translation. We first define the translation estimator which outputs a consistent and special structure on a dataset.

Definition 5. *Let $\mathcal{D} \subset \mathbb{D}^{P \times N}$ be a dataset with P channels. Then a translation-equivariant network*

$$F : \mathbb{R}^{P \times N} \to \mathbb{R}^N$$

is said to be a translation estimator for $\mathcal{D}$ if

$$F(X)[0] = \max_{i=0}^{N-1} F(X)[i],$$

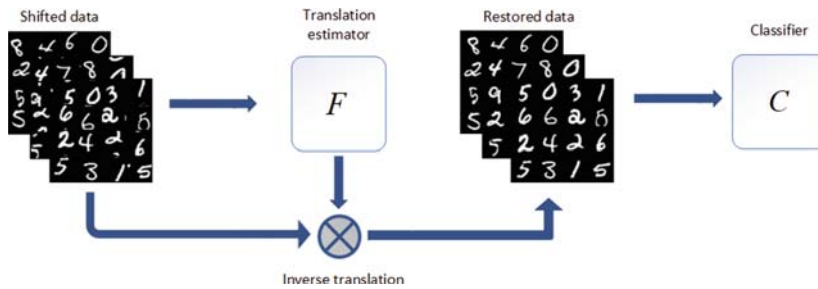*where $F(X)[i]$ is the i-th component of $F(X)$.*

Fig. 2. The pre-classifier translation restorer. For a shifted data $T^M(x)$ as the input, the translation estimator obtains the translation M and restore the original data $T^{-M}(T^M(x)) = x$, which is feed into a pre-trained classifier.

Given such a translation estimator for dataset $\mathcal{D}$ and a shifted input $X' = T^M(X)$ for some $X \in \mathcal{D}$, we propagate X' through F and get the output $F(X') \in \mathbb{R}^N$. Since the first component of $F(X)$ is the largest, the location of the largest component of $F(X')$ is exactly the translation parameter:

$$\delta(M) = \text{argmax}_{i=0}^{N-1} F_i(X').$$

Then we can restore $X = T^{-M}(X')$ by inverse translation. The restored inputs can be fed to any classifier trained on the dataset $\mathcal{D}$.

4.2 Existence of the Restorer

In this section, we show the existence of restorers, that is, the translation estimator. Note that our restorer is independent of the following classifier but dependent on the dataset. For translation, if a dataset contains both an image and a translated version of it, the estimator must be confused. We introduce aperiodic datasets to clarify such cases.

Definition 6 (Aperiodic dataset). *Let $\mathcal{D} \subset \mathbb{D}^{P \times N}$ be a finite dataset with P channels. We call $\mathcal{D}$ an aperiodic dataset if $\mathbf{0} \notin \mathcal{D}$ and*

$$T^M(X) = X' \iff M = \mathbf{0} \text{ and } X = X',$$

for $M \in \mathbb{Z}^{d+1}$ and $X, X' \in \mathcal{D}$. Here d is the spatial dimension and M decides the translation in the channel dimension in addition.

Let $\mathcal{D}$ be an aperiodic dataset. Given that $\mathbb{D} = [2^{Q+1}]$ which is the case in image classification, we prove the existence of the translation estimator for such an aperiodic dataset. The proof consists of two steps. The data are first mapped to their binary decompositions through a translation-equivariant network as Eq. (2) and then the existence of the translation-restorer in the form of Eq. (2) is proved for binary data.

Let $\mathbb{D} = [2^{Q+1}]$ and $\mathbb{B} = \{0, 1\}$. We denote $\eta : \mathbb{D} \rightarrow \mathbb{B}^Q$ to be the binary decomposition, such as $\eta(2) = (0, 1, 0)$ and $\eta(3) = (1, 0, 1)$. We perform the binary decomposition on $X \in \mathbb{D}^{P \times N}$ element-wisely and obtain $\eta(X) \in \mathbb{B}^{G \times N}$, where $G = PQ$ is the number of channels in binary representation. A dataset $\mathcal{D} \subseteq \mathbb{D}^{P \times N}$ can be decomposed into $\mathcal{B} \subset \mathbb{B}^{G \times N}$. Note that the dataset $\mathcal{D}$ is aperiodic if and only if its binary decomposition $\mathcal{B}$ is aperiodic.

The following Lemma 1 demonstrates the existence of a translation-equivariant network which coincides with the binary decomposition η on $[2^{Q+1}]^{P \times N}$. Proof details are placed in Appendix B.

Lemma 1. *Let $\eta : [2^{Q+1}] \rightarrow \mathbb{B}$ be the binary decomposition. There exists a $(2Q + 2)$-layer network F in the form of Eq. (2) with ReLU activations and width at most $(Q + 1)N$ such that for $X \in [2^{Q+1}]^{P \times N}$*

$$F(X) = \eta(X).$$

The following Lemma 2 demonstrate the existence of a 2-layer translation restorer for an aperiodic binary dataset. Proof details are placed in Appendix C.

Lemma 2. *Let $\mathcal{B} = \{Z_s | s = 1, 2, \cdots, S\} \subset \mathbb{B}^{G \times N}$ be an aperiodic binary dataset. Then there exists a 2-layer network F in the form of Eq. (2) with ReLU activations and width at most SN such that for all $s = 1, 2, \cdots, S$,*

$$F(Z_s)[0] = \max_{i=0}^{N-1} F(Z_s)[i].$$

Given a $(2Q + 2)$-layer network F' obtained from Lemma 1 and a 2-layer network F'' obtained from Lemma 2, we stack them and have $F = F'' \circ F'$ which is exactly a translation restorer. We thus have proved the following theorem.

Theorem 2. *Let $\mathcal{D} = \{X_s | s = 1, 2, \cdots, S\} \subset [2^{Q+1}]^{P \times N}$ be an aperiodic dataset. Then there exists a network $F : \mathbb{R}^{P \times N} \rightarrow \mathbb{R}^N$ in the form of Eq. (2) with ReLU activations such that for $s = 1, 2, \cdots, S$,*

$$F(X_s)[0] = \max_{i=0}^{N-1} F(X_s)[i],$$

of which the depth is at most $2Q+4$ and the width is at most $\max(SN, (Q+1)N)$. Namely, this network is a translation restorer for $\mathcal{D}$.

5   Experiments

The core component of the restorer is the translation estimator which outputs the translation parameter of the shifted inputs.

We use the architecture described in Eq. (2) with $L = 6$, $n_l = 1$ for $l = 1, \cdots, L$ and ReLU activations. The training procedure aims at maximizing the first component of the outputs. Thus the max component of the output indicates the input shift. The experimental settings are given in Appendix D. We report four sets of experiments below.
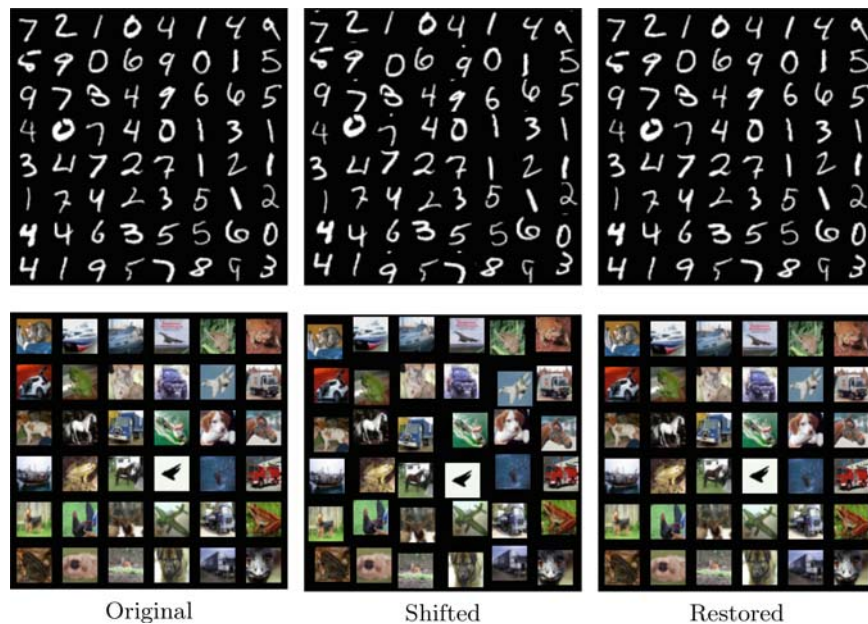
Fig. 3. The restorers for MNIST and CIFAR-10.

2D Images. We train translation restorers for MNIST and CIFAR-10. MNIST images are resized to 32×32 and CIFAR-10 images are padded 4 blank pixels at the edges.

In Fig. 3, the left column is the original images, the middle column is the randomly shifted images and the right column is the restored images. On both datasets, images are randomly shifted vertically and horizontally at most $\frac{1}{4}$ of its size. The shift is a circular shift where pixels shifted out of the figure appear on the other end. We can see that the shifted images are disorganized but the restored images are very alike the original images.

To evaluate the restoration performance of pretrained restorers, we train classifiers and test them on randomly shifted images and restored ones and the results are given in Table 1. When images are not shifted, the restorers lead to only 0.3% and 0.03% accuracy reduction on two datasets. Nevertheless, even if the translation scope is 1, restorers improve the accuracy. Moreover, no matter how the images are shifted, the restorer can repair them to the same status and result in the same classification accuracy, namely 98.59% and 88.18%, while the accuracies drop significantly without the restorer, and the larger the range of translation, the more obvious the restoration effect

Different Architectures. Our proposed restorer is an independent module that can be placed before any classifier. It is scalable to different architectures the subsequent classifier uses.

Table 1. Restoration performance on MNIST and CIFAR-10. Images are randomly shifted within the translation scope ranging from 0 to 8 in both vertical and horizontal directions. We use LeNet-5 on MNIST and ResNet-18 on CIFAR-10. "Res." and "Trans." stand for restorer and translation respectively.

	Res.\Trans.	0	1	2	3	4	5	6	7	8
MNIST	w/o	98.89	98.21	95.41	87.07	76.61	62.9	51.33	41.1	35.7
	w/	98.59	98.59	98.59	98.59	98.59	98.59	98.59	98.59	98.59
	Effect	-0.3	+0.38	+3.18	+11.52	+21.98	+35.69	+47.26	+57.49	+62.89
CIFAR-10	w/o	88.21	86.58	85.9	83.65	82.16	80.46	79.37	77.71	76.01
	w/	88.18	88.18	88.18	88.18	88.18	88.18	88.18	88.18	88.18
	Effect	-0.03	+1.6	+2.28	+4.53	+6.02	+7.72	+8.81	+10.47	+12.17

In Table 2, we evaluate the restoration performance on popular architectures including VGG-16, ResNet-18, DenseNet-121, and MobileNet v2. Translated mages (w/Trans.) are randomly shifted within scope 4 in both vertical and horizontal directions. The reduction of accuracy on original images is no more than 0.04% and the restorer improves the accuracy on shifted images by 1.66%–6.02%.

Table 2. Restoration performance on different architectures and CIFAR-10.

Res.\ Trans.	VGG-16		ResNet-18		DenseNet-121		MobileNet v2	
	w/o	w/	w/o	w/	w/o	w/	w/o	w/
w/o	89.27	83.40	88.21	82.16	92.14	90.46	88.10	83.36
w/	89.23	89.23	88.18	88.18	92.12	92.12	88.09	88.09
Effect	-0.04	+5.83	V0.03	+6.02	-0.02	+1.66	-0.01	+4.73

Translation Augmentation. Training with translation augmentation is another approach to improving the translational invariance of a model. However, translation augmentation is limited in a certain scope and thus cannot ensure the effectiveness for test images shifted out of the scope.

In Fig. 4, we compare the restoration performance on models not trained with translation augmentation (dash lines) and models trained with translation augmentation (solid lines). The augmentation scope is 10% of the image size, that is, 3 pixels for MNIST and 4 pixels for CIFAR-10. Translation augmentation indeed improves the translational invariance of the classifier on images shifted in the augmentation scope. However, when the shift is beyond the augmentation scope, the accuracy begins to degrade. In such a case, the pre-classifier restorer is still able to calibrate the shift and improve the accuracy of the classifier trained with augmentation.
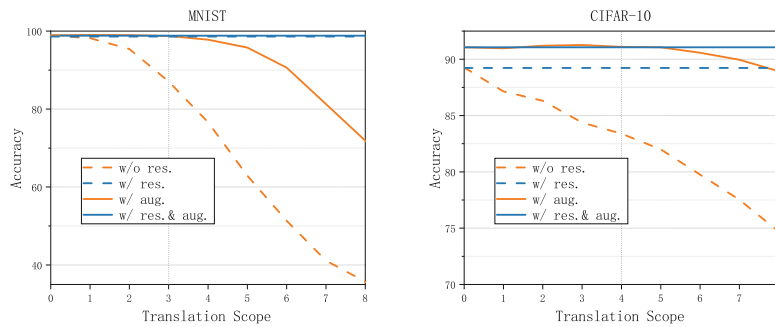
Fig. 4. Restoration performance on classifiers trained with or without translation augmentations. The models are LeNet-5 for MNIST and VGG-16 for CIFAR-10. "res. and "aug" stand for restorer and augmentation, respectively.

3D Voxelization Images. 3D-MNIST contains 3D point clouds generated from images of MNIST. The voxelization of the point clouds contains grayscale 3D tensors.

Figure 5 visualizes the restoration on 3D-MNIST. In the middle of each sub-figure, the 3-dimensional digit is shifted in a fixed direction. This fixed direction is detected by the translation estimator and the restored digit is shown on the right.
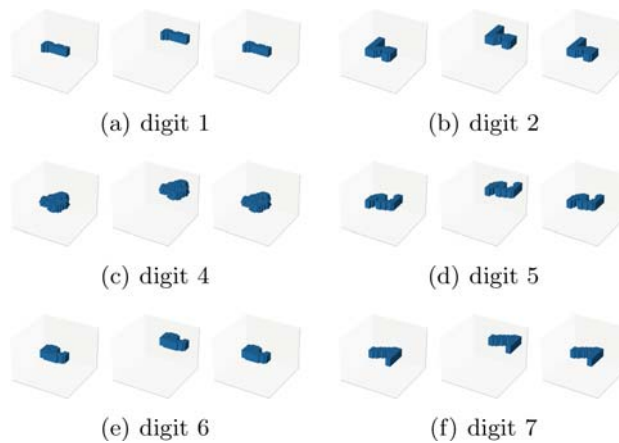

Fig. 5. The restorer on 3D-MNIST. In each sub-figure, the left is the original digit, the middle is the shifted digit, and the right is the restored digit.

6   Conclusion

This paper contributes to the equivalent neural networks in two aspects. Theoretically, we give the sufficient and necessary conditions for an affine operator $Wx+b$ to be translational equivariant, that is, $Wx+b$ is translational equivariant on a tensor space if and only if W has the high dimensional convolution structure and b is a constant tensor. It is well known that if W has the convolution structure, then Wx is equivariant to translations [5,9], and this is one of the basic principles behind the design of CNNs. Our work gives new insights into the convolutional structure used in CNNs in that, the convolution structure is also the necessary condition and hence the most general structure for translational equivariance. Practically, we propose the translational restorer to recover the original images from the translated ones. The restorer can be combined with any classifier to alleviate the performance reduction problem for translated images. As a limitation, training a restorer on a large dataset such as the ImageNet is still computationally difficult.

A   Proof of Theorem 1

We first prove a lemma.

Lemma 3. *Let $v : H \to \mathbb{R}$ be a continuous linear operator. We have*

$$v(T^M(x)) = T^{-M}(v)(x),$$

for all $x \in H$ and all $M \in \mathbb{Z}^d$.

Proof. A continuous linear operator v can be viewed as a tensor in H. We have

$$
\begin{aligned}
v(T^M(x)) &= v \cdot T^M(x) \\
&= \sum_{I \in \prod_{i=1}^d [n_i]} v[I] \cdot T^M(x)[I] \\
&= \sum_{I \in \prod_{i=1}^d [n_i]} v[I] \cdot x[I - M] \\
&= \sum_{I \in \prod_{i=1}^d [n_i]} v[I + M] \cdot x[I] \\
&= \sum_{I \in \prod_{i=1}^d [n_i]} T^{-M}(v)[I] \cdot x[I] \\
&= T^{-M}(v) \cdot x \\
&= T^{-M}(v)(x).
\end{aligned}
$$

Theorem 1. *Let $\alpha(x) = w(x) + c : \mathcal{H} \to \mathcal{H}$ be an affine operator. Then, α is equivariant with respect to translations if and only if for all $M \in \mathbb{Z}^d$,*

$$w_M = T^M(w_0) \ and \ c \propto \mathbf{1},$$

where $\mathbf{0}$ *is the zero vector in* $\mathbb{Z}^d$ *and* $c \propto \mathbf{1}$ *means that* c *is a* constant tensor, *that is, all of its entries are the same.*

Proof. Here we denote the index by $I = (i_1, i_2, \cdots, i_d)$. On the one hand,

$$
\begin{aligned}
& T^M(\alpha(x))[I] \\
&= T^M(w(x))[I] + T^M(c)[I] \\
&= w(x)[I - M] + c[I - M] \\
&= w_{I-M}(x) + c[I - M].
\end{aligned}
$$

On the other hand,

$$
\begin{aligned}
& \alpha(T^M(x))[I] \\
&= w(T^M(x))[I] + c[I] \\
&= w_I(T^M(x)) + c[I] \\
&= T^{-M}(w_I)(x) + c[I],
\end{aligned}
$$

in which the last equation is from Lemma 3.
Sufficiency. Assume for all $M \in \mathbb{Z}^d$,

$$
w_M = T^M(w_{\mathbf{0}}) \text{ and } c \propto \mathbf{1}.
$$

We have

$$
\begin{aligned}
T^{-M}(w_I) &= T^{-M}(T^I(w_{\mathbf{0}})) \\
&= T^{I-M}(w_{\mathbf{0}}) \\
&= w_{I-M}, \\
c[I - M] &= c[\mathbf{0}] \\
&= c[I], \\
T^M(\alpha(x))[I] &= \alpha(T^M(x))[I].
\end{aligned}
$$

Thus,

$$
T^M(\alpha(x)) = \alpha(T^M(x)).
$$

Necessity. Assume α is equivariant with respect to translations in the sense that

$$
T^M(\alpha(x)) = \alpha(T^M(x)).
$$

We have

$$
w_{I-M}(x) - T^{-M}(w_I)(x) = c[I] - c[I - M].
$$

Fix indices $I = \mathbf{0}$ and obtain that for all $M \in \mathbb{Z}^d$,

$$
w_M(x) - T^M(w_{\mathbf{0}})(x) = c(\mathbf{0}) - c(M).
$$

Recall that a continuous linear operator can be viewed as a tensor where the operation is the inner product in tensor space. We have

$$c(\mathbf{0}) - c(M) = (w_M - T^M(w_\mathbf{0})) \cdot x = \overrightarrow{w_M - T^M(w_\mathbf{0})} \cdot \overrightarrow{x}.$$

For each fixed M, the left side is a constant scalar and the right side is a linear transformation on all vector $\overrightarrow{x} \in \overrightarrow{H}$. Thus, both sides are equal to zero tensor and we have

$$c(\mathbf{0}) = c(M),$$
$$\overrightarrow{w_M} = \overrightarrow{T^M(w_\mathbf{0})}.$$

That is, for all $M \in \mathbb{Z}^d$,

$$c \propto \mathbf{1},$$
$$w_M = T^M(w_\mathbf{0}).$$

B   Proof of Lemma 1

We first prove a lemma.

Lemma 4. *Let $\eta : [2^{Q+1}] \to \mathbb{B}$ be the binary decomposition. There exists a $(2Q+2)$-layer network $f : \mathbb{R} \to \mathbb{R}^Q$ with ReLU activations and width at most $Q+1$ such that for $x \in [2^{Q+1}]$*

$$f(x) = \eta(x).$$

Proof. We decompose $x \in [2^{Q+1}]$ as $x = x_0 + 2x_1 + \cdots + 2^Q x_Q$. Then for $q = 0, \cdots, Q$, we have

$$x_q = \sigma(1 - \sigma(2^q + 2^{q+1}x_{q+1} + \cdots 2^Q x_Q - x)).$$

Thus, for $q = 0, \cdots, Q-1$, we construct

$$f_{2q+1}\left(\begin{pmatrix} x \\ x_Q \\ \vdots \\ x_{Q-q+1} \end{pmatrix}\right) = \sigma\left(\begin{pmatrix} 0 \\ \vdots \\ 0 \\ 2^{Q-q} \end{pmatrix} + \begin{pmatrix} 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 \\ -1 & 2^Q & \cdots & 2^{Q-q+1} \end{pmatrix}\begin{pmatrix} x \\ x_Q \\ \vdots \\ x_{Q-q+1} \end{pmatrix}\right) \in \mathbb{R}^{q+2},$$

$$f_{2q+2}\left(\begin{pmatrix} x \\ x_Q \\ \vdots \\ x_{Q-q} \end{pmatrix}\right) = \sigma\left(\begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 1 & 0 \\ 0 & \cdots & 0 & -1 \end{pmatrix}\begin{pmatrix} x \\ x_Q \\ \vdots \\ x_{Q-q} \end{pmatrix}\right) \in \mathbb{R}^{q+2}.$$

The last 2 layers

$$f_{2Q+1}(\begin{pmatrix} x \\ x_Q \\ \vdots \\ x_1 \end{pmatrix}) = \sigma(\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \begin{pmatrix} -1 & 2^Q & \cdots & 2 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} x \\ x_Q \\ \vdots \\ x_1 \end{pmatrix})\mathbb{R}^{Q+1},$$

$$f_{2Q+2}(\begin{pmatrix} x_0 \\ x_Q \\ \vdots \\ x_1 \end{pmatrix}) = \sigma(\begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} + \begin{pmatrix} -1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_Q \\ \vdots \\ x_1 \end{pmatrix}) \in \mathbb{R}^{Q+1}.$$

Let $f = f_{2Q+2} \circ \cdots \circ f_1$. For $x \in [2^{Q+1}]$ and $x = x_0 + 2x_1 + \cdots + 2^Q x_Q$ we have

$$f(x) = \begin{pmatrix} x_0 \\ x_Q \\ \vdots \\ x_1 \end{pmatrix}.$$

Lemma 1. *Let $\eta : [2^{Q+1}] \to \mathbb{B}$ be the binary decomposition. There exists a $(2Q + 2)$-layer network F in the form of Eq. (2) with ReLU activations and width at most $(Q + 1)N$ such that for $X \in [2^{Q+1}]^{P \times N}$*

$$F(X) = \eta(X).$$

Proof. From Lemma 4, there exists a network f such that for $x \in [2^{Q+1}]$, $f(x) = \eta(x)$. We denote the l-the layer of f by f_l for $l = 1, \cdots, 2Q + 2$. Without loss of generality, we assume for $z \in \mathbb{R}^{K_{l-1}}$

$$f_l(z) = \sigma(w_l \cdot z + b_l),$$

where σ is ReLU activation and $w_l \in \mathbb{R}^{K_l \times K_{l-1}}, b_l \in \mathbb{R}^{K_l}$.

Now we construct a $(2Q + 2)$-layer network F in the form of Eq. (2). For $l = 1, \cdots, 2Q + 2$, let $n_l = K_l \times P$. We construct F_l in the form of Eq. (3) as

$$W^{k_l \times p, k_{l-1} \times r}[l] = \begin{cases} \text{diag}(w_l[k_l, k_{l-1}]) & \text{if } p = r \\ 0 & \text{otherwise} \end{cases},$$

$$C^{k_l \times p, k_{l-1} \times r}[l] = \begin{cases} \dfrac{b_l[k_l]}{K_{l-1}} & \text{if } p = r \\ 0 & \text{otherwise} \end{cases},$$

for $k_l = 1, \cdots, K_l$, $k_{l-1} = 1, \cdots, K_{l-1}$ and $p, r = 1, \cdots, P$.

We can verify that for $X \in \mathbb{R}^{K_{l-1} \times P \times N}$, $k_l = 1, \cdots, K_l$, $p = 1, \cdots, P$ and $i = 0, \cdots, N-1$

$$
\begin{aligned}
F_l(X)[k_l, p, i] &= \sigma(W^{k_l \times p}[l] \cdot X + C^{k_l \times p}[l] \cdot \mathbf{1})[i] \\
&= \sigma(\sum_{k_{l-1}=1}^{K_{l-1}} W^{k_l \times p, k_{l-1} \times p} \cdot X[k_{l-1}, p, :] + C^{k_l \times p, k_{l-1} \times p})[i] \\
&= \sigma(\sum_{k_{l-1}=1}^{K_{l-1}} \mathrm{diag}(w_l[k_l, k_{l-1}]) \cdot X[k_{l-1}, p, :] + \frac{b_l[k_l]}{K_{l-1}})[i] \\
&= \sigma(b_l[k_l] + \sum_{k_{l-1}=1}^{K_{l-1}} w_l[k_l, k_{l-1}] \cdot X[k_{l-1}, p, i]) \\
&= \sigma(w_l[k_l, :] \cdot X[:, p, i] + b_l[k_l]) \\
&= f_l(X[:, p, i])[k_l].
\end{aligned}
$$

That is,

$$
F_l(X)[:, p, i] = f_l(X[:, p, i]).
$$

Thus, for $X \in \mathbb{R}^{P \times N}$ and $p = 1 \cdots P$, $i = 0, \cdots, N-1$

$$
\begin{aligned}
F(X)[:, p, i] &= f_{2Q+2}(F_{2Q+1} \circ \cdots \circ F_1(X)[:, p, i]) \\
&\vdots \\
&= f_{2Q+2} \circ \cdots \circ f_1(X[p, i]) \\
&= f(X[p, i]).
\end{aligned}
$$

For $Z \in [2^{Q+1}]^{P \times N}$,

$$
F(Z) = \eta(Z).
$$

C   Proof of Lemma 2

Assume a network $F = F_2 \circ F_1$ in the form of Eq. (2) with ReLU activations and $n_0 = G$, $n_1 = S$, $n_2 = 1$ satisfies that for $X \in G \times N$

$$
F(X) = \frac{1}{S} \sum_{s=1}^{S} \sigma(W^s[1] \cdot X + C^s[1] \cdot \mathbf{1}). \tag{4}
$$

Here, the weights and biases in F_2 degenerate as

$$
\begin{aligned}
W[2] &= W^1[2] = (\frac{1}{S}I, \cdots, \frac{1}{S}I), \\
C[2] &= 0.
\end{aligned}
$$

For convenience, in the rest of this section we simplify $W^s[1], C^s[1]$ to W^s, C^s.
    The following result is well known.

Lemma 5. *Let $\mathcal{B} = \{Z_s | s = 1, 2, \cdots, S\} \subset \mathbb{B}^{G \times N}$ be a G-channel aperiodic binary dataset. Let $\|Z\|$ be the L_2-norm of Z.*

1) $T^{\mathbf{0}}(Z_s) \cdot Z_s = \|Z_s\|^2$.
2) $T^M(Z_s) \cdot Z_t \leq \|Z_s\|^2 \leq (GN)^2$ for any $M \in \mathbb{Z}^d$.
3) For any $M \in \mathbb{Z}^d$ that $M \bmod (n_1, n_2, \cdots, n_d) \neq \mathbf{0}$, $T^M(Z_s) \cdot Z_s \leq \|Z_s\|^2 - 1$.
4) If $\|Z_s\| = \|Z_t\|$, $T^M(Z_s) \cdot Z_t \leq \|Z_t\|^2 - 1$ for any $M \in \mathbb{Z}^d$.
5) If $\|Z_s\| > \|Z_t\|$, $\|Z_s\| \geq \sqrt{\|Z_t\|^2 + 1} \geq \|Z_t\| + \frac{1}{2GN}$.

The i-th component of $F(Z)$ in Eq. (4) is

$$F(Z)[i] = \frac{1}{S} \sum_{s=1}^{S} \sigma(W_i^k \cdot Z + C^k \cdot \mathbf{1}),$$

where $W_i^k = (W_i^{k,1}, \cdots, W_i^{k,G}) \in \mathbb{R}^{G \times N}$ and $W_i^{k,r}$ is the i-th row of $W^{k,r}$. Recall that each circular filter $W^{k,r} \in \mathbb{R}^{N \times N}$ in Eq. (4) is determined by its first row $W_0^{k,r} \in \mathbb{R}^N$ and $W_{\delta(M)}^{k,r} = T^M(W_0^{k,r})$. And the biases $C^{k,r}$ are actually scalars.

Lemma 6. *Let $\mathcal{B} = \{Z_s | s = 1, 2, \cdots, S\} \subset \mathbb{B}^{G \times N}$ be a G-channel aperiodic binary dataset. Endow $\mathcal{B}$ with an order that*

$$s \geq t \iff \|Z_s\| \geq \|Z_t\|.$$

Construct the filters and biases in Eq. (4) as

$$W_0^{s,r} = \frac{Z_s^r}{\|Z_s\|},$$

$$C^{s,r} = \frac{1}{G(2GN + 1)} - \frac{\|Z_{s-1}\|}{G},$$

for $s = 1, \cdots S, r = 1 \cdots, G$ and set $\|Z_0\| = \frac{1}{2GN+1}$.
    Then,

a) if $t < s$, then $\sigma(W_i^s \cdot Z_t + C^s \cdot \mathbf{1}) = 0, i = 0, 1, \cdots, GN - 1$;
b) if $t = s$, then $\sigma(W_0^s \cdot Z_s + C^s \cdot \mathbf{1}) - \sigma(W_i^s \cdot Z_s + C^s) > \frac{1}{2GN+1}, i = 1, 2, \cdots, GN - 1$;
c) if $t > s$, then $\sigma(W_i^s \cdot Z_t + C^s \cdot \mathbf{1}) < GN$.

Proof. This proof uses Lemma 5.

a) Assuming $t < s$, we have

$$W_{\delta(M)}^s \cdot Z_t + C^s \cdot \mathbf{1} = T^M(Z_s) \cdot Z_t / \|Z_s\| + \frac{1}{2GN + 1} - \|Z_{s-1}\|,$$

$$\leq T^M(Z_s) \cdot Z_t / \|Z_s\| + \frac{1}{2GN + 1} - \|Z_t\|.$$

If $\|Z_s\| = \|Z_t\|$,

$$T^M(Z_s) \cdot Z_t/\|Z_s\| \leq \|Z_t\| - \frac{1}{\|Z_t\|},$$

$$W^s_{\delta(M)} \cdot Z_t + C^s \cdot \mathbf{1} \leq \frac{1}{2GN+1} - \frac{1}{\|Z_t\|} < 0.$$

If $\|Z_s\| > \|Z_t\|$,

$$T^M(Z_s) \cdot Z_t/\|Z_s\| \leq \frac{\|Z_t\|^2}{\|Z_t\| + \frac{1}{2GN}},$$

$$\begin{aligned}
W^s_{\delta(M)} \cdot Z_t + C^s \cdot \mathbf{1} &\leq \frac{\|Z_t\|^2}{\|Z_t\| + \frac{1}{2GN}} + \frac{1}{2GN+1} - \|Z_t\| \\
&= \frac{1}{2GN+1} - \frac{1}{2GN + 1/\|Z_t\|} \\
&< 0.
\end{aligned}$$

Thus, for all $M \in \mathbb{Z}^d$,

$$\sigma(W^s_{\delta(M)} \cdot Z_t + C^s \cdot \mathbf{1}) = 0.$$

b) We have

$$\begin{aligned}
W^s_0 \cdot Z_s + C^s \cdot \mathbf{1} &= \|Z_s\| - \|Z_{s-1}\| + \frac{1}{2GN+1}, \\
W^s_{\delta(M)} \cdot Z_s + C^s \cdot \mathbf{1} &= T^M(Z_s) \cdot Z_s/\|Z_s\| + \frac{1}{2GN+1} - \|Z_{s-1}\| \\
&\leq \frac{\|Z_s\|^2 - 1}{\|Z_s\|} + \frac{1}{2GN+1} - \|Z_{s-1}\| \\
&= \|Z_s\| - \|Z_{s-1}\| + \frac{1}{2GN+1} - \frac{1}{\|Z_s\|}.
\end{aligned}$$

Since

$$\|Z_s\| - \|Z_{s-1}\| \geq 0 \text{ and } \frac{1}{2GN+1} - \frac{1}{\|Z_s\|} < 0,$$

we have

$$\sigma(\|Z_s\| - \|Z_{s-1}\| + \frac{1}{2GN+1}) - \sigma(\|Z_s\| - \|Z_{s-1}\| + \frac{1}{2GN+1} - \frac{1}{\|Z_s\|}) \geq \frac{1}{2GN+1}.$$

c) Assuming $t > s$, we have

$$W_{\delta(M)}^s \cdot Z_t + C^s \cdot \mathbf{1} = T^M(Z_s) \cdot Z_t / \|Z_s\| + \frac{1}{2GN+1} - \|Z_{s-1}\|$$

$$\leq \|Z_s\| - \frac{1}{\|Z_s\|} + \frac{1}{2GN+1} - \|Z_{s-1}\|$$

$$< \|Z_s\|$$

$$\leq GN,$$

$$\sigma(W_{\delta(M)}^s \cdot Z_t + C^s \cdot \mathbf{1}) < GN.$$

Lemma 2. *Let $\mathcal{B} = \{Z_s | s = 1, 2, \cdots, S\} \subset \mathbb{B}^{G \times N}$ be an aperiodic binary dataset. Then there exists a 2-layer network F in the form of Eq. (2) with ReLU activations and width at most SN such that for all $s = 1, 2, \cdots, S$,*

$$F(Z_s)[0] = \max_{i=0}^{N-1} F(Z_s)[i].$$

Proof. Without loss of generality, we assign an order to the dataset that

$$s \geq t \iff \|Z_s\| \geq \|Z_t\|.$$

We set $\alpha \geq 1 + GN + 2G^2N^2$ and construct F as Eq. (4) such that

$$W_0^{s,r} = \frac{\alpha^{s-1}Z_s^r}{\|Z_s\|},$$

$$C^{s,r} = \frac{\alpha^{s-1}}{G(2GN+1)} - \frac{\alpha^{s-1}\|Z_{s-1}\|}{G},$$

for $s = 1, \cdots S, r = 1 \cdots, G$ and set $\|Z_0\| = \frac{1}{2GN+1}$.

From Lemma 6, we have for $i = 1, 2, \cdots, GN - 1$,

$$S(F(Z_t)[0] - F(Z_t)[i])$$

$$= \sum_{s=1}^{S} \alpha^{s-1} [\sigma(W_0^s \cdot Z_t + C^s) - \sigma(W_i^s \cdot Z_t + C^s)]$$

$$= \sum_{s=1}^{t} \alpha^{s-1} [\sigma(W_0^s \cdot Z_t + C^s) - \sigma(W_i^s \cdot Z_t + C^s)]$$

$$\geq \frac{\alpha^{t-1}}{2GN+1} + \sum_{s=1}^{t-1} \alpha^{s-1} [\sigma(W_0^s \cdot Z_t + C^s) - \sigma(W_i^s \cdot Z_t + C^s)]$$

$$> \frac{\alpha^{t-1}}{2GN+1} - GN \sum_{s=1}^{t-1} \alpha^{s-1}$$

$$= \frac{\alpha^{t-1}}{2GN+1} - \frac{GN(1 - \alpha^{t-1})}{1 - \alpha}$$

$$= \frac{(\alpha - 2G^2N^2 - GN - 1)\alpha^{t-1} + 2G^2N^2 + GN}{(2GN+1)(\alpha - 1)}$$

$$> 0.$$

D   Experimental Settings

For CIFAR-10, we constantly pad 4 pixels with values 0 around images. For MNIST, we resize images to 32×32. For 3D-MNIST, we voxelize this dataset and constantly pad 8 pixels with value 0 around images.

We leverage restorers with 6 layers. In each layer, we use a sparse circular filter, for example, its kernel size is 9. Each layer outputs only one channel and has no bias parameter.

References

1. Azulay, A., Weiss, Y.: Why do deep convolutional networks generalize so poorly to small image transformations? J. Mach. Learn. Res. **20**, 1–25 (2019)
2. Cohen, T., Welling, M.: Group equivariant convolutional networks. In: International Conference on Machine Learning, pp. 2990–2999. PMLR (2016)
3. Engstrom, L., Tran, B., Tsipras, D., Schmidt, L., Madry, A.: A rotation and a translation suffice: fooling CNNs with simple transformations. arXiv preprint (2018)
4. Esteves, C., Allen-Blanchette, C., Zhou, X., Daniilidis, K.: Polar transformer networks. arXiv preprint arXiv:1709.01889 (2017)
5. Fukushima, K.: Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biol. Cybern. **36**, 193–202 (1980)
6. Gens, R., Domingos, P.M.: Deep symmetry networks. Adv. Neural Inf. Process. Syst. **27** (2014)
7. Gholamalinezhad, H., Khosravi, H.: Pooling methods in deep neural networks, a review. arXiv preprint arXiv:2009.07485 (2020)
8. Ghosh, R., Gupta, A.K.: Scale steerable filters for locally scale-invariant convolutional neural networks. arXiv preprint arXiv:1906.03861 (2019)
9. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
10. Henriques, J.F., Vedaldi, A.: Warped convolutions: efficient invariance to spatial transformations. In: International Conference on Machine Learning, pp. 1461–1469. PMLR (2017)
11. Jaderberg, M., Simonyan, K., Zisserman, A., et al.: Spatial transformer networks. Adv. Neural Inf. Process. Syst. **28** (2015)
12. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
13. Lenc, K., Vedaldi, A.: Understanding image representations by measuring their equivariance and equivalence. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 991–999 (2015)
14. Lenc, K., Vedaldi, A.: Learning covariant feature detectors. In: Hua, G., Jégou, H. (eds.) ECCV 2016. LNCS, vol. 9915, pp. 100–117. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-49409-8_11
15. Marcos, D., Kellenberger, B., Lobry, S., Tuia, D.: Scale equivariance in CNNs with vector fields. arXiv preprint arXiv:1807.11783 (2018)
16. Shen, X., Tian, X., He, A., Sun, S., Tao, D.: Transform-invariant convolutional neural networks for image classification and search. In: Proceedings of the 24th ACM International Conference on Multimedia, pp. 1345–1354 (2016)

17. Shorten, C., Khoshgoftaar, T.M.: A survey on image data augmentation for deep learning. J. Big Data **6**(1), 1–48 (2019)
18. Sohn, K., Lee, H.: Learning invariant representations with local transformations. arXiv preprint arXiv:1206.6418 (2012)
19. Srivastava, N.: Improving neural networks with dropout. Univ. Toronto **182**(566), 7 (2013)
20. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
21. Xu, Y., Xiao, T., Zhang, J., Yang, K., Zhang, Z.: Scale-invariant convolutional neural networks. arXiv preprint arXiv:1411.6369 (2014)
22. Zhang, X., Yu, F.X., Karaman, S., Chang, S.F.: Learning discriminative and transformation covariant local feature detectors. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6818–6826 (2017)