Machine Learning (2023) 112:611–631
https://doi.org/10.1007/s10994-022-06288-4

Circular-symmetric correlation layer

Bahar Azari[1] · **Deniz Erdoğmuş**[1]

Received: 1 February 2022 / Revised: 24 October 2022 / Accepted: 23 November 2022 /
Published online: 28 December 2022

Abstract
Despite the vast success of standard planar convolutional neural networks, they are not the most efficient choice for analyzing signals that lie on an arbitrarily curved manifold, such as a cylinder. The problem arises when one performs a planar projection of these signals and inevitably causes them to be distorted or broken where there is valuable information. We propose a Circular-symmetric Correlation Layer (CCL) based on the formalism of roto-translation equivariant correlation on the continuous group $S^1 \times \mathbb{R}$, and implement it efficiently using the well-known Fast Fourier Transform (FFT) algorithm. We showcase the performance analysis of a general network equipped with CCL on various recognition and classification tasks and datasets.

Keywords Equivariant neural networks · Panoramic image processing · LiDAR segmentation · Fast Fourier transform (FFT)

1 Introduction

Planar convolutional neural networks, widely known as CNNs, are characterized by pattern-matching kernels that can identify motifs in the signal residing on a 2D plane. However, various applications exist in which signals lie on some curved planes, e.g., temperature and climate data on the surface of the (spherical) earth, and 360°−panoramic images and videos from panoramic cameras for 3D object classification, change detection and LiDAR segmentation. Analyzing signals in these applications is achievable by using the planar projection of them. Specifically, for 360°−panoramic image processing, which is the interest of this study, the image is usually unwrapped to a standard 2D image to be treated as an input feature map. However, the resulting arbitrary breakage of the signal at

Editors: Krzysztof Dembczynski and Emilie Devijver.

✉ Bahar Azari
azari.b@northeastern.edu

Deniz Erdoğmuş
erdogmus@ece.neu.edu

[1] Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115, USA

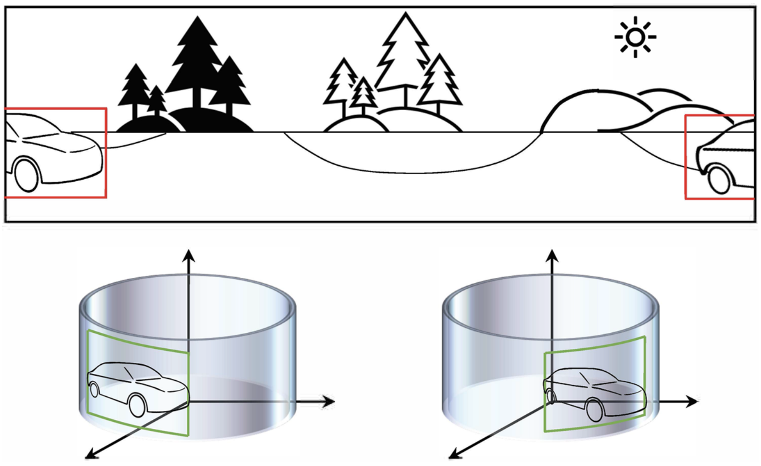


Fig. 1 Object breakage in 360°-panoramic image unwrapping. *Top:* The car has been subjected to image cut. *Bottom:* Cognition tasks should be invariant to shifting of the object on the surface the cylinder

the boundary may be destructive in object-detection tasks in terms of both information lost at the boundary and lack of equivariance to noticeable shifts. (see Fig. 1).

A convolution kernel produces a single value associated with the region in the image covered by it at a specific shift. However, the area at the boundaries of the image is neglected as the kernel shift needs to stop at a margin equal to half of the size of the kernel. This is detrimental for panoramic image processing because of potentially valuable information that exists in the border of the image (The car in Fig 1). In addition, the ever-shrinking size of the middle layer feature maps prevents the formation of deeper networks. Zero-padding applied to the out-of-image regions solves the latter issue, but the introduced distortion propagates inward from the boundaries as we go deeper in the CNN. Other proxy techniques trying to alleviate the border information loss problem exists, such as input padding [e.g., Shi et al. (2015)], but they increase the computational time and memory consumption.

Furthermore, a commonly neglected shortcoming of CNN becomes noticeable in the case of panoramic image processing where desired outputs should be immune to arbitrarily large rolling of the input image. This limitation is related to what is known as *invariance* and *equivariance* properties of the neural network as a function. For defining these properties, we consider a family, or a "group", of transformations (e.g., rotations, or translations) of input and output to a given layer of the neural network. The elements of the group can "act" on the input and output of each layer in some specific way. The neural network is invariant to the action of the group if transformations of the input do not change the output. Otherwise, it is equivariant if as we transform the input, the output is transformed according to some other action of the group. The convolution layers are empirically known to be invariant to small translations of their input image, but they are not completely immune to large shifts nonetheless (Goodfellow et al., 2009; Schmidt & Roth, 2012; He et al., 2015; Lenc & Vedaldi, 2015; Jaderberg et al., 2015; Cohen & Welling, 2016; Dieleman et al., 2016). In panoramic image processing, which can be achieved by a distortion-less projection of a cylindrical image onto a rectangular grid, arbitrary rotation around the principal axis of a cylindrical image manifests itself as a horizontal translation in a 2D grid.

Therefore, utilizing planar CNN, with the aforementioned limitation, will not guarantee invariance to such transformation when required. Figure 1 (bottom) shows an example of this phenomenon. The object (car) identification task should be invariant to rotation around the principal axis. These issues have been heretofore addressed by creating more training data using multiple circularly shifted versions of the original data, i.e., data augmentation [see Lo et al. (2002)]. Although this approach seems adequate to some extent, it increases the training time by inflicting sample complexity, does not always guarantee invariance (Elesedy & Zaidi, 2021), and could have an adversary effect on the kernels' representational capacity as it exposes them to more border areas.

Nevertheless, because the building block of CNN (i.e., convolution or cross-correlation layer) has the potential equivariance property, we may exploit them to construct a network suitable for the translation-invariant tasks such as object detection in Fig. 1. Therefore, for a systematic treatment of analyzing the 360°−panoramic data, we propose a circular-symmetric correlation Layer (CCL) based on the formalism of roto-translation equivariant correlation on the continuous group $S^1 \times \mathbb{R}$—a group constructed of the unit circle and the real line. We implement this layer efficiently using the well-known Fast Fourier Transform (FFT) and discrete cosine transform (DCT) algorithms. We discuss how the FFT yields the exact calculation of the correlation along the panoramic direction due to its circular symmetry and guarantees the invariance to circular shift. The DCT provides an improved approximation to transnational symmetry compared to what we observe in CNNs. We showcase the performance analysis of a general network equipped with CCL on various recognition and classification tasks and datasets. The PyTorch package implementation of CCL is provided in the supplementary. Our contributions are as follows:

- Theoretical treatment of circular-symmetric correlation on the surface of a cylinder.
- Efficient implementation of CCL based on FFT and DCT.
- Experimental results showing competitive performance of neural networks equipped with CCL.

2 Related work

The outstanding ability of CNN in processing spatially and temporally correlated signals comes from the fact that it exploits the translational symmetry and equivariance property of its correlation layers. In other words, a trained kernel should be able to detect a particular pattern regardless of its specific location in the image. Due to this compelling property, there has been an increasing attempt to generalize the idea of CNN to other spaces and symmetry groups (Gens & Domingos, 2014; Olah, 2014; Dieleman et al., 2015; Guttenberg et al., 2016; Dieleman et al., 2016; Cohen & Welling, 2017; Ravanbakhsh et al., 2016; Zaheer et al., 2017; Ravanbakhsh et al., 2017; Worrall et al., 2017; Maron et al., 2020; Dym & Maron, 2021). Theoretical guarantee for generalization benefit of equivariant models was treated in Elesedy and Zaidi (2021).

Most of these studies focus on discrete groups. For example, the investigation of discrete 90° rotations acting on planar images in the work of Cohen and Welling (2016), permutations of nodes in graphs in Maron et al. (2019), or permutations of points in the point cloud in Zaheer et al. (2017). Recent works [such as Cohen et al. (2018), Cohen et al. (2019)] have been investigating equivariance to continuous groups and generalized the CNN to various spaces. Kondor and Trivedi (2018) and Cohen et al. (2018) use the generalized

Fourier transform for group correlation and provided a formalism to efficiently implement these layers. Circular symmetry, which is the interest of this paper, has also been empirically studied in Schubert et al. (2019), Papadakis et al. (2010) and Kim et al. (2020), but none of these works addressed the issue in a formal analytic way.

From another perspective, vision transformers (ViT) (Dosovitskiy et al., 2020) are models with good potential in this context. The vision transformer, which is used for image classification, employs a transformer architecture over fixed-size patches of the image. Linear embeddings of the patches are added to the positional embeddings and then are fed to the transformer encoder along with a learnable classification token for classification tasks. The transformer model can take into account both the locality in the data and also any other connection that may exist between patches.

3 Circular-symmetric correlation layer

To learn a function that predicts a quantity based on a spatially correlated signal such as an image, we need to perform cross-correlation (correlation, in short). Specifically, we slide a kernel (filter) *throughout* the signal and measure the similarity. We have the familiar case of a classical planar $\mathbb{R}^2$ correlation, in which the output value at translation $x \in \mathbb{R}^2$ is computed as an inner product between the input and a kernel, translated to x. However, correlation is not limited to signals on $\mathbb{R}^2$, and in our case, we are interested in images on the surface of a cylinder. We begin our discussion by introducing the correlation on the surface of a cylinder. To do so, we start with defining its mathematical building blocks.

3.1 Preliminaries and notation

Cylinder We consider the lateral surface of a cylinder, a manifold, which is constructed by the combination of two other manifolds—a circle and a line segment.[1] The unit circle S^1, defined as the set of points $z \in \mathbb{R}^2$ with norm 1, is a one-dimensional manifold that can be parameterized by polar coordinate $\varphi \in [0, 2\pi]$. Cartesian product of S^1 with a line $\mathbb{R}$ (or, a line segment $(-a, a)$) constructs a two-dimensional manifold, known as a cylinder $\mathbb{X} = S^1 \times \mathbb{R}$ (or, $S^1 \times (-a, a)$ in case of having a line segment). We characterize the set of points on the lateral surface of the cylinder by cylindrical coordinates $\varphi \in [0, 2\pi]$ and $z \in \mathbb{R}$ and define circular-symmetric signals and convolution kernels as continuous functions on this surface $f : \mathbb{X} \mapsto \mathbb{R}^K$, where K is the number of channels.

Rotation and translation on cylinder surface The set of rotations around and translations along the z-axis is a subgroup of SE(3), the "special Euclidean group", denoted as $\mathcal{G} \leq \mathrm{SE}(3)$ and is isomorphic to $\mathbb{X}$, i.e., $\mathcal{G} = S^1 \times \mathbb{R}$. The action of an element ξ in $\mathcal{G}$ is a pair (R_ψ, ν), where R_ψ belongs to a subgroup of the "special orthogonal group" SO(3) representing a rotation by ψ around z-axis, and a translation by $\nu \in \mathbb{R}$ along z-axis. The representation of $\mathcal{G}$ corresponds to the set of all 4×4 transformation matrices of the form

$$\mathcal{G} = \left\{ \begin{pmatrix} \multicolumn{3}{c}{\multirow{3}{*}{R_ψ}} & 0 \\ & & & 0 \\ & & & \nu \\ 0 & 0 & 0 & 1 \end{pmatrix} \middle| \, \psi \in [0, 2\pi] \text{ and } \nu \in \mathbb{R} \right\}, \tag{1}$$

[1] It is either an infinite line or a line segment without its endpoints which is also a manifold.

where R_ψ is a 3D rotation matrix. In this study, we consider filters and functions on the cylindrical surface corresponding to applying the roto-translation operator L_ξ which takes a function $f : \mathbb{X} \mapsto \mathbb{R}^K$ and produces a shifted version by rotating it around and translating it along the principal axis:

$$[L_\xi f](x) = f(\xi^{-1}x). \tag{2}$$

As we explained earlier, since $\mathcal{G}$ is a group and groups contain inverses, for $\xi, \xi' \in \mathcal{G}$ we have $L_{\xi\xi'} = L_\xi L_{\xi'}$. We show this using inverse and associative properties of groups:

$$\begin{aligned}[L_{\xi\xi'}f](x) &= f\big((\xi\xi')^{-1}x\big) = f\big(\xi'^{-1}(\xi^{-1}x)\big)\\ &= [L_{\xi'}f]\big(\xi^{-1}x\big) = [L_\xi L_{\xi'}f](x).\end{aligned} \tag{3}$$

3.2 Correlation on cylinder

To define the correlation we begin with the established definition of the inner product. The inner product on the vector space of cylindrical signals is characterized:

$$\langle f, h\rangle = \int_{\mathbb{X}} \sum_{k=1}^{K} f_k(x)\, h_k(x) dx, \tag{4}$$

where the integration measure dx denotes the Haar measure (invariant integration measure) on the lateral surface of the cylinder and it is equal to $d\varphi dz$ in cylindrical coordinate. Due to the invariance of the measure, the value of the integral of a function affected by any $\xi \in \mathcal{G}$ remains the same, namely, $\int_{\mathbb{X}} f(\xi x)dx = \int_{\mathbb{X}} f(x)dx$ for all $\xi \in \mathcal{G}$. Using the inner product in (4), we define the correlation of signals and filters on the surface of the cylinder. Given a point on the cylinder $x \in \mathbb{X}$, a transformation on the subgroup of SE(3), $\xi \in \mathcal{G}$, and functions $f(x)$ and $h(x)$, the correlation is defined:

$$[f \star h](\xi) = \langle L_\xi f, h\rangle = \int_{\mathbb{X}} \sum_{k=1}^{K} f_k(\xi^{-1}x) h_k(x) dx. \tag{5}$$

Note that the correlation in (5) is also equivalent to $\langle f, L_{\xi^{-1}} h\rangle$ as the value of the correlation at a shift ξ is equal to the inner product of f and h, where either f is shifted by ξ, or h is shifted by the inverse of ξ (ξ^{-1}). Therefore, if we express the point x as $x = (\varphi, z)$, the transformation as $\xi = (\psi, \nu)$, and the Haar measure as $dx = d\varphi dz$, the correlation in (5) can be rewritten as:

$$\begin{aligned}[f \star h](\xi) &= \langle L_\xi f, h\rangle\\ &= \int_{\mathbb{R}} \int_0^{2\pi} \sum_{k=1}^{K} f_k(\varphi - \psi, z - \nu)\, h_k(\varphi - \psi, z - \nu) d\varphi dz,\end{aligned} \tag{6}$$

where the integral with respect to φ is the circular cross-correlation. It is worthwhile to mention that the resulting correlation function lies on the group $\mathcal{G}$ which is isomorphic to the space $\mathbb{X}$ that the initial functions have lied on, namely $S^1 \times \mathbb{R}$.

3.3 Equivariance of correlation layers

For the correlation in (6), defined in terms of the roto-translation operator L_ξ, we can show the crucial equivariance property known for all convolution and correlation layers. We express mathematically what we informally stated earlier.

Group actions: for a set of points $\mathbb{X}$, we have a group $\mathcal{G}$ that acts on $\mathbb{X}$. This means that for each element $\xi \in \mathcal{G}$, there exist a transformation $T_\xi : \mathbb{X} \to \mathbb{X}$ corresponding to group action $x \mapsto T_\xi(x)$. We showed this simply as ξx to simplify notation. As we have seen earlier, the action of $\mathcal{G}$ on $\mathbb{X}$ extends to functions on $\mathbb{X}$ (induced action) and that is what we have denoted as the operator $L_\xi : f \mapsto f'$ which is $f'(x) = [L_\xi f](x) = f(\xi^{-1}x)$.

Equivariance: equivariance is the potential property of a map between functions on a pair of spaces with respect to a group acting on these spaces through the group action.

Definition 1 Let $\mathbb{X}_1$, $\mathbb{X}_2$ be two sets with group $\mathcal{G}$ acting on them. Consider V_1 and V_2 as the corresponding vector spaces of functions defined on these sets, and L_ω and L'_ω as the induced actions of $\mathcal{G}$ on functions. We say that a map $\Phi : V_1 \to V_2$ is $\mathcal{G}$–equivariant if

$$\Phi(L_\omega(f)) = L'_\omega(\Phi(f)) \quad \forall f \in V_1, \forall \omega \in \mathcal{G}.$$

Considering that the map in our case corresponds to the cross-correlation function we have defined on the cylindrical surface in (5), its equivariance with respect to the action of the group $\mathcal{G} = S^1 \times \mathbb{R}$ can be demonstrated as follows:

Theorem 1 *Cross-correlation on lateral surface of a cylinder is equivariant to the action of the group $S^1 \times \mathbb{R}$.*

Proof Given that the group G of transformations on the cylinder surface is isomorphic to the set of points on the cylindrical manifold, we have:

$$\begin{aligned}[h \star L_\omega f](\xi) &\overset{\text{by (5)}}{=} \langle L_\xi h, L_\omega f\rangle = \langle L_{\omega^{-1}} L_\xi h, f\rangle \\ &\overset{\text{by (3)}}{=} \langle L_{\omega^{-1}\xi} h, f\rangle = [h \star f](\omega^{-1}\xi) \\ &\overset{\text{by (2)}}{=} [L_\omega[h \star f]](\xi),\end{aligned}$$

□ where $[h \star .](\xi)$ is the cross-correlation function, and L_ω is a transformation operator. Note that in our case $L_\omega = L'_\omega$. Equivariance can be represented graphically by commutative diagram as:

$$\begin{tikzcd} f \arrow[r, "L_\omega"] \arrow[d, "{[h \star .](\xi)}"'] & L_\omega f \arrow[d, "{[h \star .](\xi)}"] \\ {[h \star f](\xi)} \arrow[r, "L_\omega"'] & {[L_\omega[h \star f]](\xi)} \end{tikzcd}$$

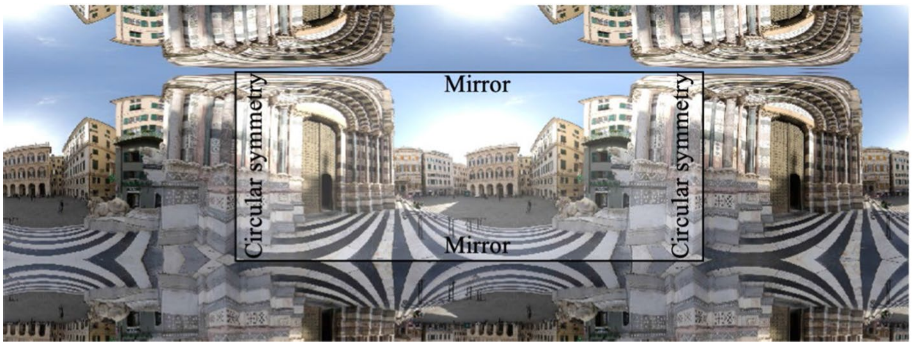


Fig. 2 360°−panoramic image with circular symmetry along the horizontal axis (unwrapped φ) and reflection symmetry along the vertical axis which are evoked by FFT and DCT, respectively

3.4 Implementing CCL using FFT

Computing cross-correlation and convolution using the Fast Fourier Transform (FFT) is known to be more efficient than their direct calculation. This is an important result of the Convolution theorem, according to which, the cross-correlation between two signals is equal to the product of the Fourier transform of one signal multiplied by the complex conjugate of Fourier transform of the other signal, or mathematically, $\widehat{f * g} = \hat{f} \odot \hat{g}$, where $\odot$ is the element-wise product. Fourier transform is a linear projection of a function onto a set of orthogonal basis functions. For the real line ($\mathbb{R}$) and the circle (S^1), these basis functions are the familiar complex exponentials $\exp(\imath n\theta)$, where $\imath = \sqrt{-1}$.

The input of the CCL is the spatial signal f on $\mathbb{X}$, sampled on a discrete grid of the cylindrical coordinate (φ, z). This signal is periodic in φ due to the 2D image being wrapped around a cylindrical manifold, and it is finite along z. Therefore, the convolution theorem holds for the dimension along unwrapped φ, and it is appropriate to use FFT for implementing the correlation in this dimension. However, we do not have the same periodicity in the z dimension. Hence, we use another set of basis functions (i.e., cosine waves), and as a consequence, we use discrete cosine transform (DCT) in the z dimension. As opposed to FFT, which is related to Fourier series coefficients of a periodically extended sequence, DCT (Muchahary et al., 2015) is associated with Fourier series coefficients of a periodically and *symmetrically* extended sequence, yields a continuous extension at the boundaries. As shown in Fig. 2 for a 360°−panoramic image, by applying FFT along unwrapped φ a circular symmetry is evoked along the horizontal axis, and by applying DCT along z dimension, a reflection symmetry is evoked along the vertical axis, which imply smooth boundaries in both dimensions. We will show in the experiments that the usage of DCT in this setting benefits the overall performance of the deep learning module in terms of vertical translation. We compute DCT by using N-FFT (Makhoul, 1980) in which a signal $f = \left[f_n \mid_{n=1}^{N}\right]$ is organized as $\hat{f} = \left[f_{2n-1} \mid_{n=1}^{N/2}, f_{N-2n+2} \mid_{n=1}^{N/2}\right]$ and FFT is applied to the resulting signal:

$$\mathrm{DCT}(f) = \Re\left(\left[2e^{\frac{-j\pi n}{2N}} \mid_{n=1}^{N}\right] \odot \mathrm{FFT}(\hat{f})\right),$$

where $\odot$ and $\Re$ denote element-wise multiplication and real part, respectively, and $e^{\frac{-j\pi n}{2N}}$ is a half-sample shift. The CCL class computation graph is summarized in Algorithm 1.

Algorithm 1 CCL class

```
Class CCL($C_{\mathrm{IN}}$, $C_{\mathrm{OUT}}$, $k_{\mathrm{H}}$, $k_{\mathrm{W}}$, $s_{\mathrm{H}}$, $s_{\mathrm{W}}$):
    Parameters: $W \in \mathbb{R}^{C_{\mathrm{OUT}} \times C_{\mathrm{IN}} \times k_{\mathrm{H}} \times k_{\mathrm{W}}}$, $b \in \mathbb{R}^{C_{\mathrm{OUT}}}$
    Input: $X \in \mathbb{R}^{C_{\mathrm{IN}} \times H \times W}$
    Output: $F \in \mathbb{R}^{C_{\mathrm{OUT}} \times \lceil \frac{H}{s_{\mathrm{H}}} \rceil \times \lceil \frac{W}{s_{\mathrm{W}}} \rceil}$
    Def Forward($X$):
        $X \leftarrow \mathtt{FFT}_h\big(\mathtt{DCT}_v(X)\big)$
        $W \leftarrow \mathtt{FFT}_h\big(\mathtt{DCT}_v(W)\big)$
        $F \leftarrow \sum_{C_{\mathrm{IN}}} X \odot W$
        $F \leftarrow \mathtt{FFT}_h^{-1}(F)$
        $F \leftarrow \mathtt{DCT}_v^{-1}(F)$
        return subsample($F, s_{\mathrm{H}}, s_{\mathrm{W}}$)
```

3.5 Computational complexity of CCL vs. CNN

The computational complexity of applying a Conv2d filter with kernel size K in a CNN to a panoramic image of size $H \times W$ is $O(WHK^2)$, whereas for CCL is $O(WH \log_2 W)$ and is independent of the kernel size. For a typical panoramic image of width $W = 1024$ with kernel size $K = 5$, $\log_2 W < K^2$ and therefore CCL performs relatively faster. This becomes more evident when using larger kernel sizes (e.g., in favor of a shallower network) or in deeper layers where kernel size becomes relatively larger w.r.t. image dimensions after applying pooling layers.

4 Experiments

We begin our experiments by investigating the effect of discretizing the continuous convolution in (6). We then demonstrate the accuracy and effectiveness of the CCL layer in comparison with the standard convolution layer by evaluating it over a couple of well-known datasets such as MNIST and CIFAR10. Finally, we provide application examples for adopting CCL in designing neural networks, e.g., 3D object classification using a 360°−panoramic projection of the object on a cylindrical surface, change detection, and LiDAR segmentation.

4.1 Discretization error of equivariant layer

In our attempt to design a group equivariant neural layer, we started by assuming the continuous group $S^1 \times \mathbb{R}$. However, for implementation, we need to discretize each function and group with a specific resolution. As a result of discretization of signal, correlation kernel, and continuous rotation group, the equivariance property does not hold exactly (i.e., we cannot exactly prove $[L_\xi f] \star g = L_\xi [f \star g]$). Furthermore, when using more than one layer, essentially a network, a moderate discretization error could propagate through the

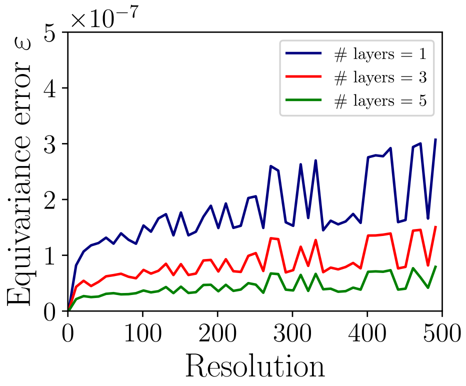


Fig. 3 Equivariance approximate error as a function of resolution and number of layer

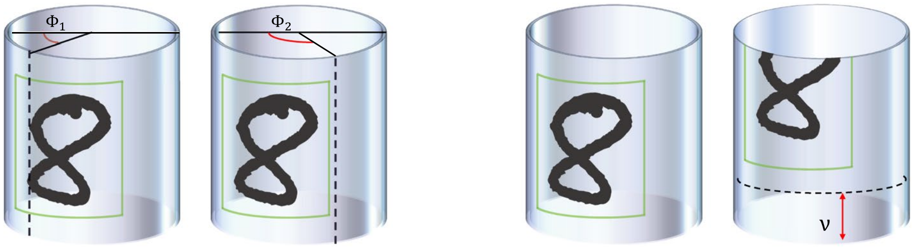


Fig. 4 $\mathcal{R}$MNIST. For the dataset to be representative of all the define transformations mentioned in the paper, namely, rotation around the z-axis and translation along the z-axis, we randomly generated the discretised rolls ($\varphi_i \in [0, 2\pi]$ with step size of 1/28). *Left*: panoramic boundary cut is rolled from φ_1 to φ_2. *right*: the image is translated north by ν

network. Therefore, it is necessary to examine the impact of discretization on the equivariance properties of the network.

To this aim, we test the equivariance of CCL by sampling $N = 1000$ random input feature maps f_n with 10 input channels, and for each feature map we sample a rotation $\varphi \in [0, 2\pi]$. To create and compare both sides of the above-mentioned equivariance equality, we first shift the input feature map before passing it through the network, and then we shift the output feature map of the intact input. We compute the discretization error as $\varepsilon = \frac{1}{N}\sum_{n=1}^{N} \operatorname{std}\big(L_{\xi_n}\Phi(f_n) - \Phi(L_{\xi_n} f_n)\big)/\operatorname{std}\big(\Phi(f_n)\big)$, where $\Phi(.)$ is a composition of CCL correlation layers with randomly initialized filters interleaved with ReLU non-linearity. As we can see in Fig. 3, the approximation error is in the order of 10^{-7} and although it grows with the resolution, it decreases when we add more layers with ReLU. Furthermore, the increase rate with resolutions of the image seems to be quite low and saturating.

4.2 Invariance analysis of networks built with CCL

We first evaluate the equivariance performance of a neural network equipped with CCL to rotations of the input along the z-axis. We propose a version of MNIST and CIFAR10 datasets called Rolled MNIST ($\mathcal{R}$MNIST) and Rolled CIFAR10 ($\mathcal{R}$CIFAR10), respectively, wrapped around a cylindrical surface as shown in Fig. 4. In these datasets, we augment the actual MNIST and CIFAR10 datasets with the horizontally rolled version of the

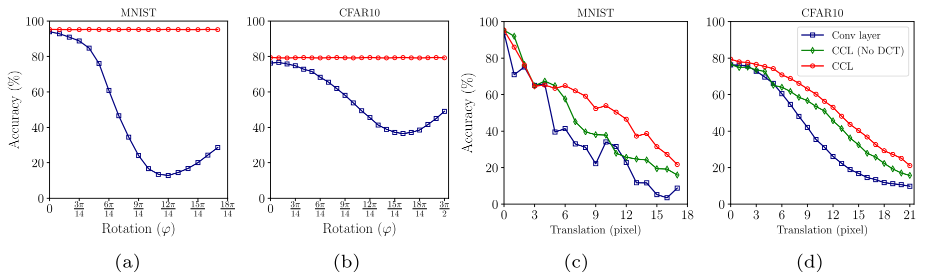


Fig. 5 Accuracy of the neural networks trained on MNIST and CIFAR10 and tested on the rolled and translated version. The red, blue, and green curves correspond to the accuracy results of CNN, the network adopting CCL, and the network adopting a CCL variant with FFT in both directions denoted by CCL (No DCT), respectively. The two left figures show the accuracy performance of the models versus the rotation of the images around the principal axis (z-axis). The two right figures show the accuracy performance of the models versus translation along the z-axis. We observe that the CCL layer is exactly equivariant to S^1, and it demonstrates a greater degree of equivariance to translation along the z-axis compared to its counterparts (conv2d) and (CCL-No DCT) (Color figure online)

original images using random samples of $\varphi \in [0, 2\pi]$ (see Fig. 4). Therefore, for a standard image size of 28×28, the rotation by $\pi/2$ is equivalent to shifting the image horizontally by $\pi/2 \times 28/2\pi = 7$. Consequently, the boundary cut of the image can pass through and destruct the consistency of the object (e.g., the digits in the MNIST dataset or the animal in the CIFAR10 dataset).

We perform three testing experiments using the actual datasets or their rolled versions and report the results in Table 1. Note that in our experiments, we do not aim at optimizing the architectures for the best accuracy. Our goal is to demonstrate how a neural network equipped with CCL can outperform regular CNN in terms of accuracy for tasks requiring equivariance. In the case of training and testing with the original MNIST and CIFAR10, the performance of a neural network using CCL is comparable to its CNN counterpart, although the CCL network slightly outperforms. However, if we train these two neural networks on the original non-augmented datasets and test them on the $\mathcal{R}$MNIST and $\mathcal{R}$CIFAR10, we see a considerable performance drop for CNN. The reason is that CNNs cannot handle a considerable degree of image translation. The accuracy of the CNN improves by training on the augmented version of the datasets, however, it is still considerably lower than that of CCL. Also, note that training with the augmented dataset is significantly slower as it contains several times more samples, i.e., for each rotation. Additionally, we compared the CCL network with circular convolutional neural network (CCNN) (Schubert et al., 2019) which uses circular padding for panoramic images. Although circular padding helps remedy the broken horizontal equivariance, CCL still performs better because using FFT and DCT promotes smooth rotation and translation equivariance. Also, CCL performs 38% faster on average. To see the adopted architectures refer to Table 2. To make the learned representation invariant to the rotation around the z-axis, a global average pooling layer is used between the correlation and fully-connected layers [see Lin et al. (2013)].

We show another set of results comparing the equivariance of neural networks adopting CCL layers and regular CNN. We adopt similar network architectures described in Table 2. CCL(M) corresponds to the usage of the CCL layer with an output channel size of M. For the regular CNN, we replace the CCL with the Conv2d layer and keep everything else the same. Figure 5 shows the accuracy of the CCL neural network (red) and CNN (blue) trained on MNIST and CIFAR10 datasets and tested on their rolled and translated versions.

The two left figures show the accuracy performance of the models versus different degrees of rotation of the images around the principal axis (z-axis). It is obvious that the CCL neural network trained only on the unperturbed data generalizes quite well in all the rotations of the test data, hence the flat red line. Nonetheless, CNN performance drops as the rotation value increases to the point where the image begins to roll back to its original position, hence the sharp drop of the blue line. The two right figures show the accuracy performance of the models versus translation along z-axis. For a finite signal, the equivariance property does not hold for the translations along the z-axis. Therefore, although the CCL layer is exactly equivariant to S^1, it is not completely equivariant to vertical translation. However, networks equipped by the CCL layer demonstrate a greater degree of equivariance compared to their counterpart (conv2d), which is the consequence of using DCT in implementing the CCL layer. Specifically, because DCT exploits an even reflection symmetry of the images, objects remain more consistent along the upper and lower edges of the image (see Fig. 2). To further investigate the effect of DCT along the vertical dimension, we also implemented a version of CCL layer adopting FFT in *both* directions (green). It can be observed that adopting DCT better preserve translation equivariance (z-axis) and consequently results in higher accuracy.

4.3 Application to 3D object classification

We evaluate the effectiveness of our model in an image-based 3D object classification task against standard CNN and spherical CNN ($\mathcal{S}^2$-CNN) (Cohen et al., 2018), which is a convolutional neural network for spherical data. Specifically, we took three different architectures VGG16, ResNet18, and a self-designed architecture to compare the classification results of various networks adapting CCL layers versus Conv2d. Here, we adopt continuous panoramic views of 3D objects that describe the position of the object's surface with respect to a cylindrical surface in the 3D space (Yavartanoo et al., 2018; Sfikas et al., 2018; Shi et al., 2015; Papadakis et al., 2010). We use ShapeNetCore, a subset of the ShapeNet dataset, (Chang et al., 2015), with 51,300 unique 3D models that covers 55 common object categories. This dataset consists of two subsets: a regular subset of consistently aligned 3D models and another subset where 3D models are perturbed by random rotations in 3D. For this study, we aim to use 360°−panoramic images with rotations along a specific axis (e.g. z-axis). We construct this dataset by rolling the 360°−panoramic images in Fig. 6b using random samples of $\varphi \in [0, 2\pi]$. According to the Table 1, the classification accuracy of our model is higher than that of CNN, CCNN, and $\mathcal{S}^2$-CNN for this dataset. See Table 2 for information regarding the architecture.

4.4 Application to change detection

We further evaluated the CCL in image-based detection of temporal scene changes. We used the Tsunami change detection dataset (Sakurada & Okatani, 2015), which consists of one hundred aligned panoramic image pairs of scenes in tsunami-damaged areas of Japan and the corresponding ground-truth masks of scene changes. We generated a randomly rolled version of this dataset ($\mathcal{R}$TSUNAMI) in which the first and second views are relatively rolled. This task demonstrates the absolute need for an equivariant layer for designing an efficient neural network architecture. Since the inputs are two panoramic images, a regular network architecture requires the two images to be precisely aligned. Hence the network performance declines when this requirement is not satisfied.

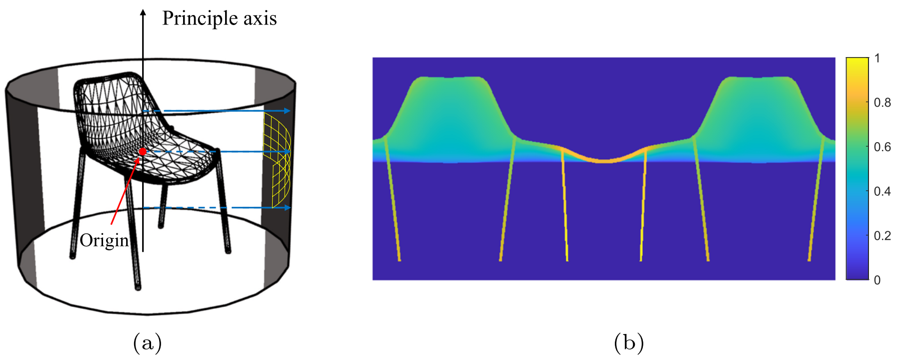


Fig. 6 360°–panoramic view of a 3D object: **a** we cast a ray from each pixel on the cylinder surface to the 3D object and measure the depth. **b** Depth measurements form a 360°–panoramic image

We relax this cumbersome condition, by designing a neural network that is equivariant to the roll of the first view and invariant to the shift of the second view. The network consists of two CCL(50)-MaxPool-ReLU layers followed by two CCL(50)-Transpose-ReLU and a final CCL(1)-Sigmoid layer which regresses the change mask. The second layer is followed by a global AvgPool across feature maps of the second view (to make it invariant) and is then summed with feature maps of the first view. CCL-Transpose layer is implemented by up-sampling feature maps with zeros of appropriate stride. This network architecture is visualized in Fig. 7 and summarized in Table 2. We tested our model against the CNN and CCNN versions of the same architecture and reported the results in Table 1. In Fig. 8 we show that our model generalizes better to the test set due to its equivariance to S^1.

4.5 Application to LiDAR semantic segmentation

Lastly, we evaluated the CCL in LiDAR data semantic segmentation. For this purpose, we used the nuScenes (Caesar et al., 2020) dataset, including 40, 000 LiDAR point clouds (34, 000 train and 6000 test) with 32 highly imbalanced semantic labels. We merged these labels into six dominant classes of background, pedestrian, bicycle/motorcycle, movable object, bus/truck, and car, and projected each point cloud onto a panoramic image (see supplementary for more details). We adopted the architecture shown in Table 2 for CCL, CNN, and CCNN with a weighted cross-entropy loss to account for class imbalance. We also compared with $\mathcal{S}^2$-CNN (Cohen et al., 2018) by projecting each point cloud on the surface of a sphere. We report balanced accuracy (average of the recall for all classes) in Table 1. The confusion matrix for networks equipped with CCL and Conv2d CNN are depicted in Fig. 10. The CCL network has done a better job classifying the labels specifically regarding the bicycle/motorcycle class. In Fig. 9, we show an instance of the segmentation performance for CCL and Conv2d CNN along with ground truth (GT). CCL performs better particularly on the vertical borders. Note that $\mathcal{S}^2$-CNN is not an effective choice for LiDar dataset in the autonomous driving application. This is because the vertical field of view is often as low as 40° and as a result, a large portion of pixels in the sphere (top and bottom regions) is left unused.

Table 1 Accuracy/F-score results for network using CCL, Conv2d, circular Conv2d, and spherical convolution layers

Train set	MNIST	MNIST	$\mathcal{R}$MNIST	CIFAR10	CIFAR10	$\mathcal{R}$CIFAR10	ShapeNet	TSUNAMI	LiDAR
Test set	MNIST	$\mathcal{R}$MNIST	$\mathcal{R}$MNIST	CIFAR10	$\mathcal{R}$CIFAR10	$\mathcal{R}$CIFAR10	$\mathcal{R}$ShapeNet	$\mathcal{R}$TSUNAMI	$\mathcal{R}$LiDAR
CCL	**95.27**	**95.15**	**95.35**	**79.40**	**79.24**	**79.02**	**82.14**	**88.91***	**76.43**
Conv2d	93.96	16.81	46.64	76.29	49.11	66.07	68.08	76.46*	73.54
CCNN	94.23	93.01	93.77	79.18	77.59	78.90	77.34	82.62*	74.67
$\mathcal{S}^2$-CNN	–	–	–	–	–	–	71.06	–	69.47
CCL (ResNet18)	99.03	**98.97**	**99.03**	–	–	–	–	–	–
Conv2d (ResNet18)	**99.05**	19.51	58.17	–	–	–	–	–	–
CCL (VGG16)	–	–	–	**93.46**	**93.17**	**93.50**	–	–	–
Conv2d (VGG16)	–	–	–	93.44	55.79	74.23	–	–	–

The best performance results are highlighted in bold

*These values are F-scores

Table 2 Network architectures

Layer	MNIST	CIFAR10	Panoramic ShapeNet	Panoramic TSUNAMI	nuScenes LiDAR
Input	$f \in \mathbb{R}^{1\times28\times28}$	$f \in \mathbb{R}^{3\times32\times32}$	$f \in \mathbb{R}^{1\times48\times100}$	$f \in \mathbb{R}^{1\times28\times128}$	$f \in \mathbb{R}^{4\times40\times360}$
1	CCL(8), ReLU	CCL(128), ReLU	CCL(64), BN, ReLU	CCL(50), ReLU	CCL(32), ReLU
2	CCL(8), ReLU	CCL(128), ReLU	MaxPool(2, 2)	MaxPool(2, 2)	CCL(32), ReLU
3	MaxPool(2, 2)	MaxPool(2, 2)	CCL(64), BN, ReLU	CCL(50), ReLU	CCL(32), ReLU
4	CCL(8), ReLU	CCL(128), ReLU	CCL(128), BN, ReLU	MaxPool(2, 2)	CCL(32), ReLU
5	CCL(8), ReLU	CCL(256), ReLU	MaxPool(2, 2)	AvgPool(224)	CCL(6), Softmax
6	MaxPool(2, 2)	MaxPool(2, 2)	CCL(256), BN, ReLU	UpSample(2, 2)	
7	CCL(10), ReLU	AvgPool(8)	AvgPool(300)	CCL(50), ReLU	
8	AvgPool(7), Softmax	FC(256, 120), ReLU	FC(256, 100), ReLU	UpSample(2, 2)	
9		FC(120, 84), ReLU	FC(100, 55), Softmax	CCL(50), ReLU	
10		FC(84, 10), Softmax		CCL(1), Sigmoid	

CCL(c_{OUT}): c_{OUT} implies number of output channels. FC(l_{IN}, l_{OUT}): l_{IN} and l_{IN} imply input and output features dimensions, respectively. MaxPool(k, s): k and s imply kernel and stride sizes, respectively. AvgPool(k): k implies kernel sizes. BN denotes batch normalization. The global average pooling makes the network invariant to the input roll. For regular CNN and circular CNN, the CCL layers are replaced with Conv2d and circularly-padded Conv2d layers, respectively

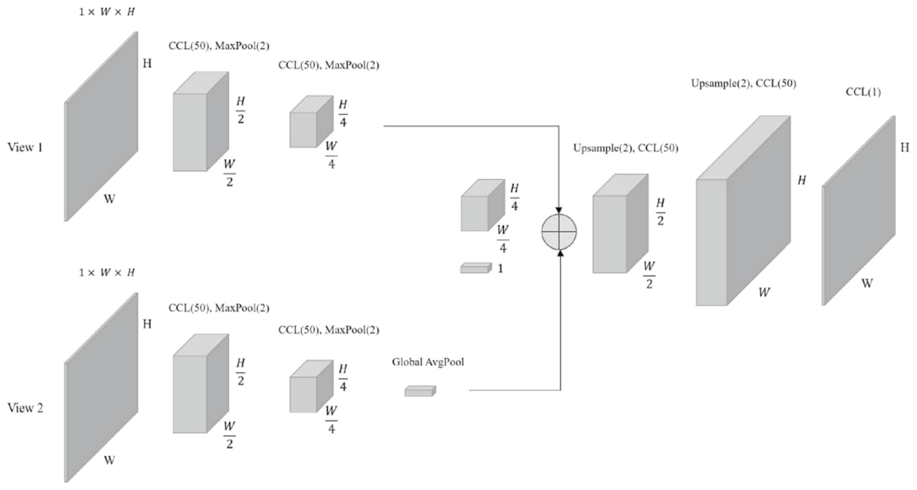
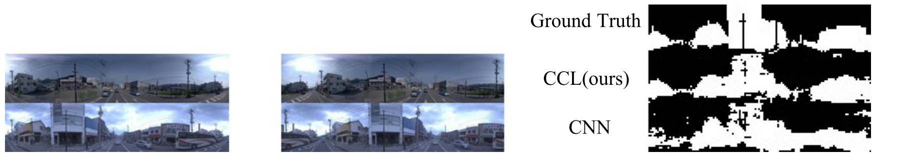


Fig. 7 Network architecture design for change detection



(a) Original aligned panoramics (b) Relatively rolled panoramics (c) Change detection performance comparison

Fig. 8 Change detection performance comparison for Tsunami dataset

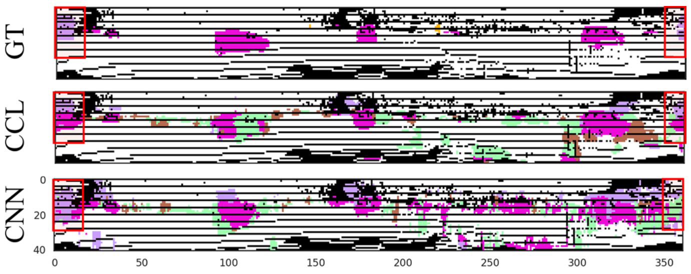


Fig. 9 Segmentation results compared to ground truth (GT). CCL performs better particularly on the vertical borders

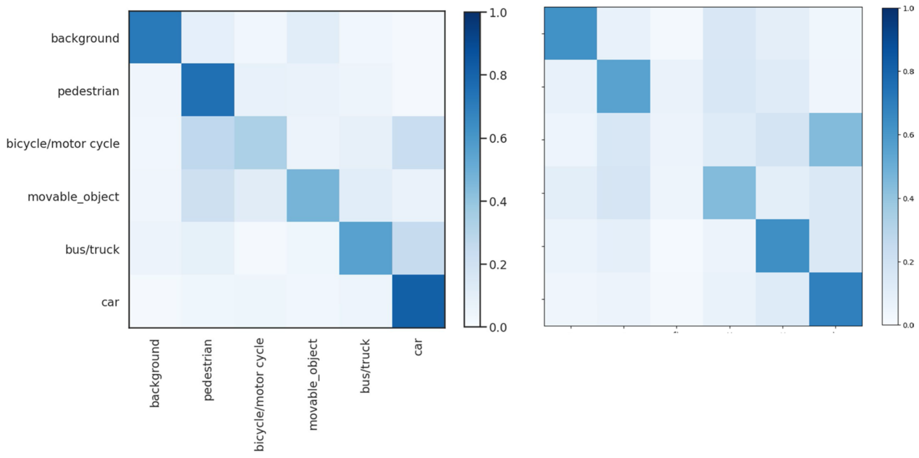


Fig. 10 Confusion matrix, CCL (Left), Conv2d CNN (Right)

5 Discussion and conclusion

We have proposed a Circular-symmetric Correlation Layer (CCL) based on the formalism of roto-translation equivariant correlation on the continuous group $S^1 \times \mathbb{R}$, and implement it efficiently using the well-known FFT and DCT algorithm. Our numerical results demonstrate the effectiveness and accuracy obtained from adopting the CCL layer. A neural network equipped with CCL generalizes across rotations around the principal axis and outperforms its CNN counterpart and state-of-the-art on competitive 3D model recognition. Note that the achieved gain is not at the expense of increasing the number of parameters (by zero- or input-padding of the input data) or data augmentation and hence longer training time and sample complexity. It is merely due to the intrinsic property of the CCL layer in mimicking the circular symmetry and reflection symmetry in the data.

Appendix A Background on group theory

The formalism used in the paper is based on various concepts in group theory and abstract algebra. In this part, we aim to provide these key concepts, and their corresponding notations and definitions.

Symmetry: a symmetry is a set of transformations applied to a structure. The transformations should preserve the properties of the structure. Generally it is also presumed that the transformations must be invertible, i.e. for each transformation there is another transformation, called its inverse, which reverses its effect.

Symmetry is thus can be stated mathematically as an operator acting on an object, where the defining feature is that the object remains unchanged. In other words, the object is invariant under the symmetry transformation. Symmetries are modeled by *Groups*.

Group: let $\mathcal{G}$ be a non-empty set with a binary operation defined as $\circ : \mathcal{G} \times \mathcal{G} \mapsto \mathcal{G}$. We call the pair $(\mathcal{G};\circ)$ a group if it has the following properties:

(*Closure*): $\mathcal{G}$ is closed under its binary operation,
(*Associativity axiom*): the group operation is associative -i.e., $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$ for $g_1, g_2, g_3 \in \mathcal{G}$,
(*Identity axiom*): there exists an identity $e \in \mathcal{G}$ such that $g \circ e = e \circ g = g$ for all $g \in \mathcal{G}$,
(*Inverse axiom*): every element $g \in \mathcal{G}$ has an inverse $g^{-1} \in \mathcal{G}$, such that $g \circ g^{-1} = g^{-1} \circ g = e$.

Subgroup: A non-empty subset $\mathcal{H}$ of $\mathcal{G}$ is called a subgroup, if $\mathcal{H}$ is a group equipped with the same binary operation of as in $\mathcal{G}$. We show this as $\mathcal{H} \leq \mathcal{G}$. $\mathcal{H}$ is called a proper subgroup of if $\mathcal{H} \neq \mathcal{G}$ and we show it as $\mathcal{H} < \mathcal{G}$.

Group order: the number of elements in a group $\mathcal{G}$ is called the *order* of $\mathcal{G}$ and is denoted $\mid \mathcal{G} \mid$. $\mathcal{G}$ is called a finite group if $\mid \mathcal{G} \mid < \infty$ and infinite otherwise.

Group action: we are interested on the way a group "acts" on the input and output of a deep network. Function $\gamma : \mathcal{G} \times \mathbb{X} \rightarrow \mathbb{X}$ is the left action of group $\mathcal{G}$ on $\mathbf{x}$ iff I) $\gamma(e, \mathbf{x}) = \mathbf{x}$ and; II)$\gamma(g_1, \gamma(g_2, \mathbf{x})) = \gamma(g_1 g_2, \mathbf{x})$.

Faithful $\mathcal{G}$-action: $\mathcal{G}$-action is faithful iff two groups are isomorphic $\mathcal{G} \cong \mathcal{G}_{\mathbb{N}}$.

Normal subgroup: for $\mathcal{H}$, a subgroup of a group $\mathcal{G}$, the similarity transformation of $\mathcal{H}$ by a fixed element g in $\mathcal{G}$ not in $\mathcal{H}$ always gives a subgroup. If

$$gHg^{-1} = H$$

for every element g in $\mathcal{G}$, then $\mathcal{H}$ is said to be a normal subgroup of $\mathcal{G}$, written $\mathcal{H} \lhd \mathcal{G}$. Normal subgroups are also known as invariant subgroups or self-conjugate subgroup.

Homogeneous space and transitivity: transitivity is the property that taking any $x_0 \in \mathcal{X}$, any other $x \in \mathcal{X}$ can be reached by the action of some $g \in \mathcal{G}$, i.e., $x = g(x_0)$. If the action of $\mathcal{G}$ on $\mathcal{X}$ is transitive, we say that X is a homogeneous space of $\mathcal{G}$.

Homomorphism: let $\mathcal{G}$ with binary operation $\circ$ and $\mathcal{H}$ with binary operation $\star$ be groups. The map $\Phi : \mathcal{G} \to \mathcal{H}$ is called a homomorphism from $(\mathcal{G}, \circ)$ to $(\mathcal{H}, \star)$, if for all $g_1, g_2 \in \mathcal{G}$ we have:

$$\Phi(g_1 \circ g_2) = \Phi(g_1) \star \Phi(g_2).$$

A homomorphism $\Phi : \mathcal{G} \to \mathcal{H}$ is called

monomorphism if the map Φ is injective,
epimorphism if the map Φ is surjective,
isomorphism if the map Φ is bijective,
endomorphism if $\mathcal{G} = \mathcal{H}$,
automorphism if $\mathcal{G} = \mathcal{H}$ and the map Φ is bijective.

Isomorphic: two groups are isomorphic $\mathcal{G} \cong \mathcal{H}$ if there exists a bijection $\Phi : \mathcal{G} \to \mathcal{H}$ between them.

Appendix B Code, Datasets, and Experimental Settings

In this section, we explain the details of dataset preparation, code, and experimental settings.

B.1 Datasets

Rolled MNIST and Rolled CIFAR10 We first evaluate the generalization performance a neural network equipped with CCL with respect to rotations of the input along z-axis. We propose a version of MNIST and CIFAR10 datasets called $\mathcal{R}$MNIST and $\mathcal{R}$CIFAR10, respectively, wrapped around a cylindrical surface as shown in Fig. 4.

In this dataset, we augment the actual MNIST and CIFAR10 datasets with horizontally rolled version of the original images using random samples of $\varphi \in [0, 2\pi]$ (see Fig. 4). Therefore, for a standard image size of 28×28, the rotation by $\pi/2$ is equivalent to shifting the image horizontally by $\pi/2 \times 28/2\pi = 7$. As the result, the images could be cut in the middle and destruct the consistency of the object in the figure, namely the digits in MNIST dataset, or the animal in CIFAR10 dataset.

For the dataset to be representative of all the define transformations mentioned in the paper, namely, rotation around the z-axis and translation along the z-axis, we randomly generated the discretised rolls ($\varphi_i \in [0, 2\pi]$ with step size of 1/28). As illustrated in Fig. 11(*bottom-left*: original cylindrical image. *bottom-right*: The image is translated up by v. *up-left*: panoramic is rolled by φ_1 (the image is cut in the middle). *up-right*: panoramic is rolled by φ_2) this transformations will disturb the perception neural network.

Panoramic change detection dataset: TSUNAMI dataset, Sakurada and Okatani (2015), consists of one hundred panoramic image pairs of scenes in tsunami-damaged areas of Japan. The size of these images is 224×1024 pixels. For each image, they hand-labeled the ground truth of scene changes. It is given in the form of binary image of the same size as the input pair of images. The binary value at each pixel is indicative of the change that occurred at the corresponding scene point on the paired images. The scene changes are defined to be detected as 2D changes of surfaces of objects (e.g., changes of the advertising board) and 3D, structural changes (e.g., emergence/vanishing of buildings and cars).

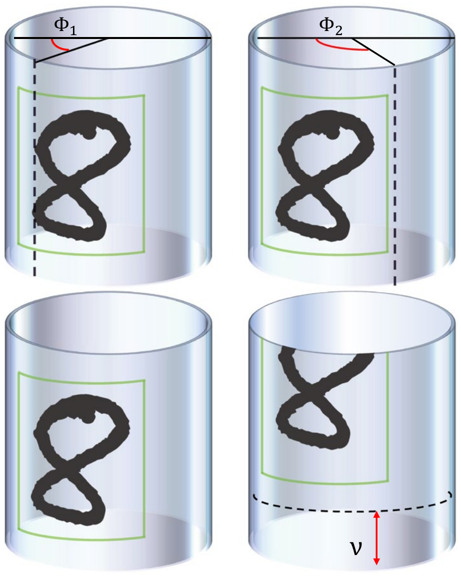
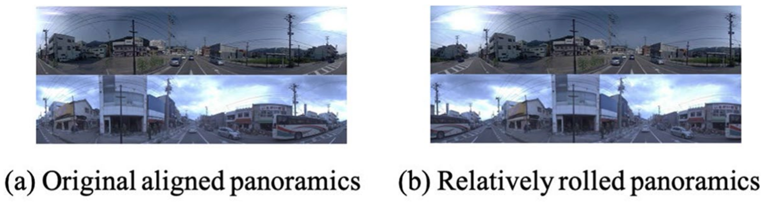


Fig. 11 $\mathcal{R}$MNIST. For the dataset to be representative of all the define transformations mentioned in the paper, namely, rotation around the z-axis and translation along the z-axis, we randomly generated the discretised rolls ($\varphi_i \in [0, 2\pi]$ with step size of 1/28).*bottom-left*: original cylindrical image. *bottom-right*: the image is translated up by ν. *up-left*: panoramic is rolled by φ_1 (the image is cut in the middle). *up-right*: panoramic is rolled by φ_2



(a) Original aligned panoramics
(b) Relatively rolled panoramics

Fig. 12 TSUNAMI dataset. **a** The original dataset with the two panoramic images aligned at the time of capture. **b** The dataset we built based on the original one to roll/shift one of the panoramic image relative to the other

The changes due to differences in illumination and photographing condition and those of the sky and the ground are excluded, such as changes due to specular reflection on building windows and changes of cloud and signs on the road surface. For the ground-truth, all image pairs have ground truths of temporal scene changes, which are manually obtained by the authors in Sakurada and Okatani (2015).

The original dataset is captured in a way that the two panoramic images to be aligned completely as it can be seen in Fig. 12. In order to make the dataset more challenging and realistic, and to demonstrate the power of using CCL on this dataset, we built a variation of the dataset based on the original one by rolling/shifting one of the panoramic image relative to the other. In this way the network needs to be invariant to these rolls in the images in order to perform the task.

LiDAR Dataset The LiDAR dataset consists of sequences of 3D point clouds collected from 1000 scenes and annotated at 2 Hz (40,000 point clouds/LiDAR sweeps in total). The LiDAR's vertical field of view (FOV) is between −10° to 30° and its horizontal FOV is between −180° to 180°. We projected each point cloud to a panoramic image of height $40 = 30 - (-10)$ and width $360 = 180 - (-180)$ considering a vertical and horizontal

Table 3 Training time per epoch (s) and inference time per sample (ms) for CCL, Conv2d, circular Conv2d, and spherical convolution layers

Dataset	MNIST	CIFAR10	ShapeNet	TSUNAMI	LiDAR
CCL	**11 (4)**	**17 (8)**	**83 (19)**	**31 (12)**	**470 (41)**
Conv2d	16 (7)	29 (13)	119 (39)	47 (18)	699 (75)
CCNN	18 (7)	33 (13)	126 (40)	52 (19)	734 (76)
$\mathcal{S}^2$-CNN	–	–	882 (94)	–	1707 (133)

The fastest training and inference times are highlighted in bold

The numbers are in the following format: training time (inference time)

resolution of 1°, and encoded 3D coordinates of each point as well as its LiDAR intensity in the four channels of this panoramic image, resulting in a 360°−panoramic image of size $4 \times 40 \times 360$ for each point cloud along with its ground-truth annotation.

B.2 Code and computational resources

We implemented CCL and performed our experiemnts in PyTorch v1.8 (Paszke et al., 2017) and used the Adam optimizer (Kingma & Ba, 2014) with learning rate of 0.001. We initialized all parameters randomly. Each CCL layer has $C_{out} \times C_{in} \times K_1 \times K_2$ parameters similar to that of a CNN layer. The time complexity for training a CCL layer is $O(C_{in} \times C_{out} \times N \times \log(N))$ per data sample and epoch, where N is the dimension of input data, and the space complexity is $O(b \times C_{in} \times C_{out} \times N)$ where b is the batch size. We learned and tested all the models on an Intel Core i9 CPU@3.6GHz with 64 GB of RAM and Nvidia GeForce RTX 2080 with 11 GB of RAM . Per-epoch training time varied from 30ms in smaller datasets to 1s in larger experiments and 25 epochs sufficed for all experiments.

B.3 Training and inference time

We have reported the training and inference times in Table 3.

Author contributions Bahar Azari conceived the presented idea, developed the theory, implemented the algorithm, and conducted experiments. Deniz Erdogmus verified the analytical methods. All authors discussed the results. Bahar Azari wrote, and Deniz Erdogmus revised the final manuscript.

Funding The authors received no financial support for the research, authorship, and/or publication of this article.

Availability of data and materials All the data used in the paper are publicly available at the following links: MNIST: http://yann.lecun.com/exdb/mnist/. CIFAR-10: https://www.cs.toronto.edu/ kriz/cifar.html. TSUNAMI: https://kensakurada.github.io/pcd_dataset.html. LiDAR: https://www.nuscenes.org/lidar-segmentation.

Code availability The code will be available upon publication.

Declarations

Conflict of interest Not applicable.

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

References

Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., & Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving. In *2020 IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (pp. 11618–11628). IEEE Computer Society.

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., & Su, H., & Xiao, J. (2015). Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012.

Cohen, T., & Welling, M. (2016). Group equivariant convolutional networks. In *International conference on machine learning* (pp. 2990–2999).

Cohen, T. S., & Welling, M. (2017). Steerable CNNs. In *International conference on learning representations*.

Cohen, T. S., Geiger, M., & Weiler, M. (2019). A general theory of equivariant CNNs on homogeneous spaces. In *Advances in neural information processing systems (NeurIPS)* (Vol. 32).

Cohen, T. S., Geiger, M., Köhler, J., & Welling, M. (2018). Spherical CNNs. In *International conference on learning representations*.

Dieleman, S., De Fauw, J., & Kavukcuoglu, K. (2016). Exploiting cyclic symmetry in convolutional neural networks. In *International conference on machine learning* (pp. 1889–1898). PMLR.

Dieleman, S., Willett, K. W., & Dambre, J. (2015). Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly Notices of the Royal Astronomical Society, 450*(2), 1441–1459.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., & Gelly, S., et al (2020). An image is worth 16x16 words: Transformers for image recognition at scale. In *International conference on learning representations*.

Dym, N., & Maron, H. (2021). On the universality of rotation equivariant point cloud networks. In *International conference on learning representations*.

Elesedy, B., & Zaidi, S. (2021). Provably strict generalisation benefit for equivariant models. In Meila, M., Zhang, T. (Eds.), *Proceedings of the 38th international conference on machine learning, ICML 2021, 18–24 July 2021, virtual event. Proceedings of machine learning research* (Vol. 139, pp. 2959–2969). PMLR.

Gens, R., & Domingos, P. M. (2014). Deep symmetry networks. *Advances in Neural Information Processing Systems, 27,* 2537–2545.

Goodfellow, I., Lee, H., Le, Q., Saxe, A., & Ng, A. (2009). Measuring invariances in deep networks. *Advances in Neural Information Processing Systems, 22,* 646–654.

Guttenberg, N., Virgo, N., Witkowski, O., Aoki, H., & Kanai, R. (2016). Permutation-equivariant neural networks applied to dynamics prediction. arXiv preprint arXiv:1612.04530.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 37*(9), 1904–1916.

Jaderberg, M., Simonyan, K., Zisserman, A., & Kavukcuoglu, K. (2015). Spatial transformer networks. In *Proceedings of the 28th international conference on neural information processing systems* (pp. 2017–2025).

Kim, J., Jung, W., Kim, H., & Lee, J. (2020). CyCNN: A rotation invariant CNN using polar mapping and cylindrical convolution layers. arXiv preprint arXiv:2007.10588.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kondor, R., & Trivedi, S. (2018). On the generalization of equivariance and convolution in neural networks to the action of compact groups. In *International conference on machine learning* (pp. 2747–2755).

Lenc, K., & Vedaldi, A. (2015). Understanding image representations by measuring their equivariance and equivalence. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 991–999).

Lin, M., Chen, Q., & Yan, S. (2013). Network in network. arXiv preprint arXiv:1312.4400.

Lo, S.-C.B., Li, H., Wang, Y., Kinnard, L., & Freedman, M. T. (2002). A multiple circular path convolution neural network system for detection of mammographic masses. *IEEE Transactions on Medical Imaging, 21*(2), 150–158.

Makhoul, J. (1980). A fast cosine transform in one and two dimensions. *IEEE Transactions on Acoustics, Speech, and Signal Processing, 28*(1), 27–34.

Maron, H., Ben-Hamu, H., Shamir, N., & Lipman, Y. (2019). Invariant and equivariant graph networks. In *International conference on learning representations*.

Maron, H., Litany, O., Chechik, G., & Fetaya, E. (2020). On learning sets of symmetric elements. In *International conference on machine learning* (pp. 6734–6744). PMLR.

Muchahary, D., Mondal, A. J., Parmar, R. S., Borah, A. D., & Majumder, A. (2015). A simplified design approach for efficient computation of dct. In *2015 5th international conference on communication systems and network technologies* (pp. 483–487). IEEE.

Olah, C. (2014). Groups and group convolutions. https://colah.github.io/posts/2014-12-Groups-Convolution/.

Papadakis, P., Pratikakis, I., Theoharis, T., & Perantonis, S. (2010). Panorama: A 3d shape descriptor based on panoramic views for unsupervised 3d object retrieval. *International Journal of Computer Vision, 89*(2), 177–192.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., & Lerer, A. (2017). Automatic differentiation in PyTorch. In *NIPS 2017 Workshop on Autodiff*.

Ravanbakhsh, S., Schneider, J., & Poczos, B. (2016). Deep learning with sets and point clouds. arXiv preprint arXiv:1611.04500.

Ravanbakhsh, S., Schneider, J., & Poczos, B. (2017). Equivariance through parameter-sharing. In *International conference on machine learning* (pp. 2892–2901). PMLR.

Sakurada, K., & Okatani, T. (2015). Change detection from a street image pair using CNN features and superpixel segmentation. *BMVC, 61,* 1–12.

Schmidt, U., & Roth, S. (2012). Learning rotation-aware features: From invariant priors to equivariant descriptors. In *2012 IEEE conference on computer vision and pattern recognition* (pp. 2050–2057). IEEE.

Schubert, S., Neubert, P., Pöschmann, J., & Protzel, P. (2019). Circular convolutional neural networks for panoramic images and laser data. In *2019 IEEE intelligent vehicles symposium (IV)* (pp. 653–660). IEEE.

Sfikas, K., Pratikakis, I., & Theoharis, T. (2018). Ensemble of panorama-based convolutional neural networks for 3d model classification and retrieval. *Computers & Graphics, 71,* 208–218.

Shi, B., Bai, S., Zhou, Z., & Bai, X. (2015). Deeppano: Deep panoramic representation for 3-d shape recognition. *IEEE Signal Processing Letters, 22*(12), 2339–2343.

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., & Brostow, G .J. (2017). Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 5028–5037).

Yavartanoo, M., Kim, E. Y., & Lee, K. M. (2018). Spnet: Deep 3d object classification and retrieval using stereographic projection. In *Asian conference on computer vision* (pp. 691–706). Springer.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., & Smola, A. J. (2017). Deep sets. In *Advances in neural information processing systems* (pp. 3391–3401).