

Achieving Rotation Invariance in Convolution Operations: Shifting from Data-Driven to Mechanism-Assured

Hanlin Mo

Guoying Zhao*

{hanlin.mo, guoying.zhao}@oulu.fi

Center for Machine Vision and Signal Analysis, University of Oulu
Oulu, Finland

ABSTRACT

Achieving rotation invariance in deep neural networks without relying on data has always been a hot research topic. Intrinsic rotation invariance can enhance the model's feature representation capability, enabling better performance in tasks such as multi-orientation object recognition and detection. Based on various types of non-learnable operators, including gradient, sort, local binary pattern, maximum, etc., this paper designs a set of new convolution operations that are naturally invariant to arbitrary rotations. Unlike most previous studies, these rotation-invariant convolutions (RCONVs) have the same number of learnable parameters and a similar computational process as conventional convolution operations, allowing them to be interchangeable. Using the MNIST-Rot dataset, we first verify the invariance of these RCONVs under various rotation angles and compare their performance with previous rotation-invariant convolutional neural networks (RI-CNNs). Two types of RCONVs based on gradient operators achieve state-of-the-art results. Subsequently, we combine RCONVs with different types and depths of classic CNN backbones. Using the OuTex_00012, MTARSI, and NWPU-RESISC-45 datasets, we test their performance on texture recognition, aircraft type recognition, and remote sensing image classification tasks. The results show that RCONVs significantly improve the accuracy of these CNN backbones, especially when the training data is limited. Furthermore, we find that even with data augmentation, RCONVs can further enhance model performance.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**.

KEYWORDS

Image Rotation, Rotation Invariance, Convolutional Neural Network, Non-learning Operators

*The corresponding author.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ACM MM, 2024, Melbourne, Australia

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM

<https://doi.org/10.1145/nnnnnnnn.nnnnnnnn>

1 INTRODUCTION

In simple terms, the core challenge in the field of computer vision is how to enable machines to process visual information they acquire efficiently, much like the human visual system. Since 2012, Deep Neural Networks (DNNs), represented by Convolutional Neural Networks (CNNs), have made remarkable advancements in core tasks of computer vision and pattern recognition, such as object classification [19, 36], retrieval [30, 34], detection [20, 38], and segmentation [23, 31]. This has led people to question whether the core challenge mentioned above has already been addressed. In reality, there is still a huge gap between current DNN models and the human brain, and we choose an important property, rotation invariance, to illustrate this issue.

Rotation invariance is one of the most common type of perceptual constancy, which enables the human visual system to efficiently and accurately recognize objects with arbitrary orientations, and plays a vital role in numerous practical applications, such as remote sensing image classification [4, 27], and medical image analysis [16]. However, CNN models lack the invariance to image rotations, and simply increasing the number of model layers or learnable parameters does not effectively enhance the rotation invariance of CNNs. A common solution is to use data augmentation techniques to rotate input samples during DNN training, resulting in data-driven rotation-invariant CNNs (RI-CNNs). However, previous research has indicated that data augmentation further weakens the interpretability of CNNs that are already "black boxes", disrupts semantic information of data, and reduces the effective utilization of learnable parameters, leading to increased redundancy [33, 37]. Clearly, it is not an ideal solution for achieving rotation invariance in CNNs.

Thus, recent research has focused on ensuring the rotation invariance of CNNs through the mechanisms within the models themselves rather than relying on data. Representative works include the Spatial Transformer Network (STN) [14], Polar Transformer Network [8], Group Equivariant Convolutional Network (G-CNN) [6], Oriented Response Network (ORN) [39], Harmonic Network (H-Net) [37], and General E(2)-Equivariant Steerable CNN (E(2)-CNN) [32], among others [2, 7, 15, 21]. However, they suffer from the following three issues: (1) Most of them exhibit invariance only to specific rotation angles and cannot handle continuous/arbitrary rotation angles [6, 21, 39]. (2) Some rotation-invariant network modules have architectures that are completely different from conventional convolution operations, making them non-interchangeable [7, 12, 33]. This prevents them from being integrated with classic CNN backbones such as VGG and ResNet. (3) Due to the inclusion of additional learnable parameters, training of some methods still relies on data augmentation [14, 17, 27].

The most ideal scenario would be to design new convolution operations that can be interchanged with conventional convolution operation without increasing the number of learnable parameters and naturally exhibits invariance to arbitrary rotations. In fact, researchers have begun exploring this direction. Recently, some researchers designed several rotation-invariant convolution operations (RConvs) based on Sobel operator[9], Local Binary Pattern (LBP)[10], rotation-invariant coordinate systems[24], and sort operation[25], meeting the aforementioned requirements. However, these tasks are still preliminary and not systematic. Firstly, many other methods can also be used to achieve rotation invariance in convolution operations, such as maximum operation and gradient estimation based on first-order Gaussian derivatives. Secondly, the performance of RConvs implemented in different ways has not been compared in practical tasks. The purpose of this paper is to address these issues. Our main contributions are as follows:

- We develop a set of RConvs using Sobel operator, Gaussian derivatives, sort operation, maximum operation, LBP and other methods. Each of RConvs has the same number of learnable parameters as its corresponding conventional convolution operation, and the output feature maps of both also have the same size, thus enabling them to be interchangeable.
- Without using data augmentation, we validate the invariance of different RConvs to various rotation angles using the MNIST-rot dataset [18], and compare their performance with previous rotation-invariant CNN models. Inherent flaws in some RConvs are analyzed in detail.
- We combine these RConvs with classical CNN backbones of different types and depths (such as VGG[28], Inception[29], ResNet[11], and DensNet[13]), and test their performance in texture image classification, aircraft type recognition, and remote sensing image classification based on Outex_TC_00012[26], MTARSI-20[35], and NWPU-RESISC45 datasets[3]. Also, we analyze the influence of data augmentation on RConvs.

2 RELATED WORK

This section briefly introduces the existing methods for achieving rotation invariance in CNN models, most of which can be categorized into the following three types.

2.1 Data-driven RI-CNNs

Using data augmentation is the most direct method to achieve rotation invariance in CNN models, namely applying random rotations to input images during model training. However, previous research has found that using data augmentation can lead to a significant amount of learnable parameter redundancy in the model [37]. As the model learns from the training data that it should extract multi-orientation information, there inevitably are different rotation "copies" of the same convolutional kernel in its convolutional layers. Clearly, these redundant parameter "copies" are the cost the model incurs to achieve rotation invariance. Additionally, some researchers have found that data augmentation further diminishes the feature interpretability of CNNs[33]. Beyond direct data augmentation, some methods use "implicit" data augmentation to achieve rotation-invariant networks [2, 14, 17]. For instance, TI Pooling [17] first generates multiple rotated versions of the input

image, then uses the same CNN to extract their features separately, and performs a pooling operation on these features. Methods like STN [5, 14] essentially belong to this category too. These methods first use learnable modules to estimate rotation angle of the input, and then calibrate the input to a standard orientation using the angle before extracting features. However, to enable the angle estimation module to predict correct rotation angles, we need to train it using samples which are randomly rotated. In addition, such methods generally can only handle global rotations of images, not local rotations (i.e., only some objects rotate while other objects and background remain unchanged). Moreover, some utilize Siamese structure and contrastive learning loss to achieve the model's rotation invariance [15], but also employ data augmentation when constructing positive sample pairs.

2.2 Rotation-equivariant CNNs

Based on the theories of Lie groups and Lie algebras, some researchers have designed rotation-equivariant networks represented by group convolution networks[6] and E(2)-CNN[32], and have successfully applied them to practical tasks such as medical image analysis and protein structure prediction[1, 16]. Taking group convolution as an example, it extracts features of the input in multiple orientations simultaneously. This set of multi-orientation features forms the group features of the input, which exhibits equivariance rather than invariance to rotations, and is input as a whole into the next group convolution layer. However, group convolution generally only has invariance to specific rotation angles, such as multiples of 90° . It should be noted that the structure of equivariant convolutions often differs significantly from standard convolutions, making their integration into classic CNN backbones quite challenging. Moreover, the computational cost of some such methods is relatively high[24].

2.3 Mechanism-assured RI-CNNs

Due to various issues with data-driven RI-CNNs, some researchers have started exploring certain mechanisms to ensure CNNs' rotation invariance, proposing different solutions. For example, some researchers use the representation of inputs in polar or log-polar coordinates as the input for CNNs[8, 22]. In fact, 2D rotations in Cartesian coordinates become translations in log-polar/polar coordinates, and traditional convolution operations are invariant to 2D translations. However, the transformation of coordinates disrupts the spatial relationships between image contents, leading to a decline in model performance. Moreover, similar to STNs[14], these methods can only handle global rotations of images. ORN[39] and Rotation Equivariant Vector Field Network (RotEqNet)[21] extract features of the input in multiple orientations through rotating convolutional kernels. For example, if convolutional kernels are rotated by $k \cdot 45^\circ$, $k = 1, 2, \dots, 8$, these models are invariant to those specific rotation angles. Clearly, to achieve rotation invariance at any angle, we need to generate as many rotated copies of the convolutional kernel as possible, which significantly increases the model's computational efficiency.

Recent work has begun to explore how to achieve invariance of CNN models for any rotation angle, and requires that the designed RConvs and conventional convolution can be replaced with each

other. To this end, Hao et al. designed Gradient-Align CNN using Sobel operator[9], Hao and Zhang et al. designed the Regional Rotate Layer using LBP[10], and Mo et al. designed rotation-invariant coordinate CNN[24] and sorting convolution [25] using rotation-invariant coordinates system and ring sorting operations, respectively. In fact, these methods share a similar computational process, which involves calibrating the local region involved in the convolution operation with a non-learnable operator first, followed by the standard convolution operation. Since non-learnable operators are used, these RConvs have the same number of learnable parameters as their corresponding traditional convolutions. Furthermore, their computational processes are similar, allowing them to be interchangeable. These methods form the research foundation and starting point of our paper. In this paper, we design a set of new RConvs using various types of non-learnable operators. Based on the experimental results obtained on synthetic and real image datasets, we systematically compare the performance of different RConvs on various tasks and identified some inherent shortcomings in certain RConvs.

3 METHODOLOGY

This section first illustrates how to achieve the invariance of conventional convolution operation under any rotation angles using gradient operators, sorting operation, LBP feature, and maximum operation, respectively. Then, we explain how to combine the designed RConvs with classical CNN backbones.

3.1 Implementing RConv Using Non-learnable Operations

The input of a convolution operation is typically an image or a feature map, which can be represented as a $h \times w \times c$ tensor, where h , w , and c represent the height, width, and number of channels of the tensor, respectively. To simplify subsequent analysis, we assume $c = 1$. In this way, the input tensor can be represented as a 2D function $F(X) : \Omega \rightarrow \mathbb{R}$, where the domain $\Omega = 1, 2, \dots, h \times 1, 2, \dots, w$. At a position $X_0 \in \Omega$, the conventional convolution operation $Conv$ can be expressed as

$$Conv(X_0, F(X)) = \sum_{P \in S} W(P) \cdot F(X_0 + P) \quad (1)$$

where W is a $K \times K$ learnable kernel, K is a non-negative integer; P enumerates all sampling positions on a $K \times K$ square grid S where X_0 is the origin center of the grid. For example, when W is a 3×3 kernel, we have $S = (-1, -1), (-1, 0), \dots, (0, 1), (1, 1)$, which contains 9 sample positions. Let $G(Y)$ be a rotated version of $F(X)$, i.e., $G(Y) = F(R_{-\theta}Y)$, where $R_{-\theta}$ is a 2×2 rotation matrix and $\theta \in [0, 2\pi)$ represents the rotation angle. Assuming Y_0 is the corresponding position of X_0 after rotation, the convolution operation at Y_0 is

$$Conv(Y_0, G(Y)) = \sum_{P \in S} W(P) \cdot G(Y_0 + P) \quad (2)$$

Since $G(Y) = F(R_{-\theta}Y)$, we have

$$G(Y_0 + P) = F(R_{-\theta}(Y_0 + P)) = F(X_0 + R_{-\theta}P) \quad (3)$$

By substituting (3) into (2), we can find that

$$\begin{aligned} Conv(Y_0, G(Y)) &= \sum_{P \in S} W(P) \cdot F(X_0 + R_{-\theta}P) \\ &\neq \sum_{P \in S} W(P) \cdot F(X_0 + P) \neq Conv(X_0, F(X)) \end{aligned} \quad (4)$$

This means that $Conv$ is not invariant to 2D rotations.

In fact, a direct and simple way to achieve the rotation invariance of $Conv$ is to eliminate the influence of rotations on the $K \times K$ region involved in the convolution operation. In this paper, we achieve this through various non-learnable operations (NLOPs). For example, we can utilize certain gradient operators to compute the gradient direction of the region, and eliminate the influence of the rotation angle θ by aligning the region to a standard orientation. Therefore, the proposed RConv can be defined as:

$$RConv(X_0, F(X)) = \sum_{P \in S} W(P) \cdot NLOP(F(X_0 + P)) \quad (5)$$

For any $P \in S$, $NLOP$ ensures the relationship $NLOP(G(Y_0 + P)) = NLOP(F(X_0 + P))$ always holds, thus ensuring $RConv(F(X), X_0) = RConv(G(Y), Y_0)$.

We take a 3×3 convolution as an example to introduce seven types of NLOPs that can eliminate the influence of rotations. It is worth noting that compared to the Cartesian coordinate system, previous studies have shown that *RConvs* implemented based on the polar coordinate system have better invariance[24, 25]. Therefore, we adopt this approach as well. For a 3×3 region under the polar coordinate system, apart from X_0 , we need to sample 8 positions, $P_1, P_2, \dots, P_8$, at equal angular intervals on the circumference with X_0 as the center and a radius of 1. The angle corresponding to the position P_i is $(i-1) \cdot \frac{360^\circ}{8}$ (clockwise direction), where $i = 1, 2, \dots, 8$. Using bilinear interpolation, we can obtain the pixel values of the input $F(X)$ at these positions, which will participate in the convolution operation.

SB-Conv: SB-Conv uses 3×3 Sobel operators to compute the gradient of a given region, and then rotates the region to a standard orientation using the obtained gradient orientation. Specifically, 3×3 Sobel operators on x and y directions are defined as follows:

$$Sobel_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, Sobel_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix} \quad (6)$$

We use a simple nearest-neighbor interpolation instead of bilinear interpolation to transform $Sobel_x$ and $Sobel_y$ into the polar coordinate system. Then, we convolve them separately with the 3×3 region to obtain the gradient $Gradient = [G_x, G_y]$ of that region. Subsequently, the gradient orientation $\phi \in (0, \pi]$ is calculated using the arctan function $\phi = \arctan(G_y/G_x)$, and the region is rotated to the standard orientation (i.e., the gradient orientation coincides with the horizontal axis direction) using this angle. Fig.1(a) illustrates this process more clearly. When the region is rotated by any angle θ , the gradient orientation calculated using Sobel operators also becomes $(\phi + \theta)$. Therefore, the region is always aligned to the same standard orientation. Since the convolution operation is performed on the aligned region, SB-Conv evidently exhibits rotation invariance.

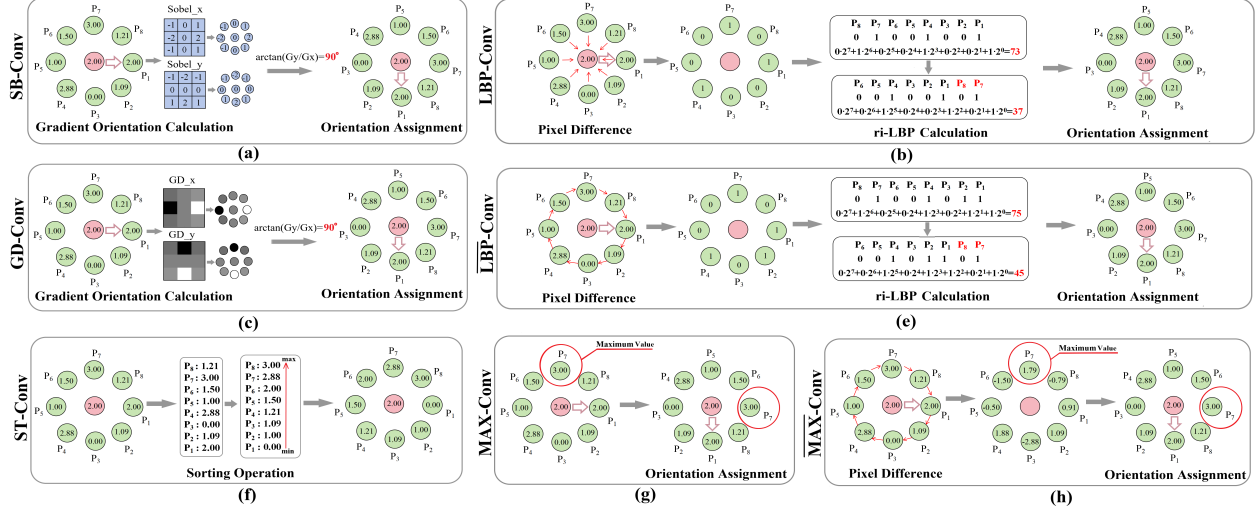


Figure 1: Taking a 3×3 convolutional kernel as an example, we explain the computation process of seven different RConvs.

GD-Conv: Similar to SB-Conv, GD-Conv also calibrates the 3×3 region using gradient orientation to achieve rotation invariance of convolution. The only difference is that we use the first derivatives of 2D Gaussian function GD_x and GD_y instead of the Sobel operator to compute the gradient (see Fig.1(b)). The definitions of GD_x and GD_y are as follows:

$$GD_x = \frac{-1}{2\pi\sigma^4} \cdot x \cdot e^{-\frac{x^2+y^2}{2\sigma^2}}, GD_y = \frac{-1}{2\pi\sigma^4} \cdot y \cdot e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (7)$$

where the parameter σ is the scale factor used to control the decay rate of the Gaussian function. To ensure that all information of GD_x and GD_y falls within the $K \times K$ region as much as possible, according to the three-sigma rule, we set $\sigma = K/6$. Therefore, when the region size is 3×3 , $\sigma = 0.5$. Additionally, due to the smoothing effect of the Gaussian function, the first-order Gaussian derivative is more robust to disturbances such as noise compared to Sobel operators.

ST-Conv: ST-Conv uses a sorting operation to eliminate the influence of rotation. As shown in Fig.1(c), we first sort pixel values of the input at the other 8 sampling positions $[F(X_0 + P_1), \dots, F(X_0 + P_8)]$ in ascending order, except for the central position X_0 . Then, the sorted values are placed back at positions $P_1, P_2, \dots, P_8$ in order, with the minimum value placed at P_1 and the maximum value at P_8 . Obviously, even if the local region undergoes rotation, the sorted sequence of values remains unchanged, ensuring the rotation invariance of ST-Conv. It should be noted that the sorting operation actually disrupts the local structure of the input to some extent, leading to information loss, especially when dealing with regions of large size K . Therefore, when implementing convolution kernels of size 5×5 and 7×7 , we use the ring sorting operation proposed by Mo et al.[25], which sorts the sampling positions with different radial values separately, to preserve some spatial information as much as possible.

LBP-Conv: This method, proposed by Hao et al. based on LBP[10], first computes the pixel differences between the input at the sampling position $P_1 \sim P_8$ and the center position X_0 , i.e., $[F(X_0 + P_1) - F(X_0), F(X_0 + P_2) - F(X_0), \dots, F(X_0 + P_8) - F(X_0)]$. Then,

the pixel differences are binarized, where values less than 0 are set to 0, and values greater than or equal to 0 are set to 1, resulting in a binary encoding called LBP feature, for example, 01001001 (from P_8 to P_1). To achieve LBP feature's rotation invariance, the leading bit of the binary descriptor needs to be continuously moved to the end, and the corresponding decimal value of the new binary encoding is computed. By repeating this process, the smallest decimal value and its corresponding binary encoding can be found. For example, the decimal value corresponding to 01001001, which corresponds to the smallest decimal value 37. At this point, the sampling positions corresponding to this new binary number are $P_6, P_5, P_4, P_3, P_2, P_1, P_8, P_7$. That is, P_7 is rotated to the original position of P_1 . Clearly, through this method, we can also rotate the region to the standard orientation, eliminating the influence of rotations.

LBP-Conv: We design a variant of LBP-Conv, whose calculation process is basically the same as LBP-Conv, except that we do not calculate the pixel difference between other positions and the center position, but instead calculate the pixel difference between adjacent positions, i.e., $[F(X_0 + P_1) - F(X_0 + P_2), F(X_0 + P_2) - F(X_0 + P_3), \dots, F(X_0 + P_8) - F(X_0 + P_1)]$. In fact, previous researchers have designed various variants of standard LBP using different encoding schemes, and we plan to compare the invariance of RConvs implemented using different LBP encodings.

MAX-Conv: For the pixel values at the 8 sampling positions around the center point, $[F(X_0 + P_1), \dots, F(X_0 + P_8)]$, MAX-Conv first identifies the position of the maximum value, for example, P_7 . Suppose a 3×3 region involved in the convolution computation is rotated by 90° , the maximum value within the region remains unchanged, but its position changes from P_7 to P_9 (rotating P_7 clockwise by 90° will aligns it with P_9). Clearly, by rotating the position where the maximum value occurs to the standard direction (aligning it with the horizontal direction), we can achieve the convolution's rotation invariance. The entire process is illustrated in Fig.1(g).

$\overline{\text{MAX}}$ -Conv: Similar to $\overline{\text{LBP}}$ -Conv, $\overline{\text{MAX}}$ -Conv first computes the pixel differences at adjacent sampling positions of the input, i.e., $[F(X_0 + P_1) - F(X_0 + P_2), F(X_0 + P_2) - F(X_0 + P_3), \dots, F(X_0 + P_8) - F(X_0 + P_1)]$; then identifies the position where the maximum difference occurs and rotates that position to the standard orientation (see Fig.1(h)). In fact, MAX-Conv can only be used if there is only one maximum value within the $K \times K$ region. If there are multiple identical maximum values within the region, the positions where two $K \times K$ regions satisfying the rotation relationship are aligned to the standard orientation may differ. This weakens the rotation invariance of MAX-Conv. The introduction of $\overline{\text{MAX}}$ -Conv is to mitigate this problem to some extent, as the probability of multiple identical pixel differences occurring is relatively low.

3.2 Integrating RConv with CNNs

Obviously, all RConv proposed in the previous section have the same number of learnable parameters as their corresponding Conv. For an input, if the outputs of these RConvs are of the same size as the output of the Conv, it indicates that they can be mutually substituted. Then, we can further achieve a CNN backbone's rotation invariance by replacing its all Convs with a certain type of RConvs. Assuming the height and width of an input are h, w , and the kernel size of both RConvs and the Conv is $K \times K$. When using zero padding during computation, the output size of the Conv is $h \times w$. For RConvs, before performing the convolution operation, we first need to calibrate $K \times K$ local region at each spatial position $X \in \Omega$ using non-learnable operators. Then, we concatenate all calibrated local regions according to their spatial positions to form a new input, where its height and width are $(K \cdot h)$ and $(K \cdot w)$, respectively. Next, we perform $K \times K$ convolution operation on this input but set the convolution stride to K and do not use padding. Obviously, the output size obtained with this setting is also $h \times w$, which is exactly the same as that obtained with the Conv. Therefore, RConvs and Conv can replace each other.

By replacing all Conv in a traditional CNN models with some type of RConvs, we can obtain a RI-CNN. When simultaneously inputting an image and its rotated version into the RI-CNN, the two features obtained in each RConv layer satisfy the same rotation relationship. If we use max or average pooling operations to down-sample the spatial size of this output to 1×1 (downsampling can also be achieved in previous RConv layers by setting convolution stride > 1 or using pooling operations), the resulting feature is invariant to any rotations and can be further used as the input of fully connected layers.

4 EXPERIMENTS

This section validates the performance of RConvs in various practical tasks. Based on the MNIST-Rot dataset, we first train various types of RConvs without using data augmentation, and then test their performance on test sets that contains rotated images to evaluate their rotation invariance, and compare their performance with some existing RI-CNNs. Subsequently, we deploy RConvs into classical CNN backbones and test their performance on texture image recognition, aircraft classification, and remote sensing image tasks based on the Outex_00012, MTARSI, and RESISC-45 datasets. Finally, we demonstrate through experiments that RConvs can still

further improve the performance of CNN models even when using data augmentation.

4.1 Experiment Setup

Datasets: (1) MNIST dataset [18] contains 70K 28×28 images of handwritten digits (0-9), with 60K designated for training and 10K for testing. Each test image is rotated 36 times, with rotation angles at $0^\circ, 10^\circ, 20^\circ, \dots$, up to 360° , thus generating a total of 360K rotated test images. These form a new test set called MNIST-rot. Additionally, 10K training images are randomly selected for validation, leaving 50K images in the training set. Part of the images from the MNIST training set and the MNIST-Rot test set are shown in Fig.2a. (2) The Outex_TC_00012 dataset[26] contains 9120 texture images belonging to 24 different categories (all are 128×128 grayscale images). For each category, the collectors select 20 texture surfaces of that category and initially photographed them under three lighting conditions ("inca", "t184", and "horizon"), which serve as training images. Then, under "t184" and "horizon" lighting conditions, the collectors photograph each texture surface again from 8 different angles ($5^\circ \sim 90^\circ$) to serve as test images. Therefore, the training set includes 1440 images, and the test set comprises 7680 images. Part of the images from the training and test sets are shown in Fig.2b. (3) The MTARSI dataset is an RGB image dataset for aircraft type recognition tasks, containing 9385 images of aircraft belonging to 20 different types. As shown in Fig.2c, aircrafts in these images often have different orientations. Also, there are significant variances in the background. The size of each image was adjusted to 128×128 , and then 200 images are randomly selected from each category as training samples. Thus, the training set totals 4000 images, with the remaining 5385 images forming the test set. (4) NWPU-RESISC45 is a dataset for remote sensing image scene classification. It contains 31500 RGB images belonging to 45 scene categories, with each category comprising 700 samples. All images were resized to 128×128 , and 400 images from each category are randomly selected as training images, with the remaining images used for testing. As shown in Fig.2d, due to arbitrary shooting angles, many categories such as "bridges" and "ground track fields" exhibit rotation variations.

Model: (1) First, we need to verify the rotation invariance of all RConvs based on the MNIST dataset. For this purpose, we initially design a CNN model with six convolutional layers as the Baseline, with the number of kernels in each layer being 32, 32, 64, 64, 128, and 128, respectively. The kernel size of the last two convolutional layers is 3×3 , while that of the first four layers is 7×7 . After the second and fourth layers, there is a 2×2 max pooling operation, and after the last convolutional layer, there is a 7×7 average pooling layer. Since the size of the input images is 28×28 , these pooling operations reduce the spatial size of the feature images to 1×1 , reducing them to feature vectors. This vector is then fed into a fully connected layer with ten units for the final classification. By replacing all conventional convolution operations in this Baseline with certain type of RConvs, we can obtain an RI-CNN model, such as SB-CNN, GD-CNN, and Max-CNN. (2) To illustrate that RConvs can be integrated with common CNN backbones, we chose VGG16, Inception V1, DenseNet40, and ResNet18/34/50/101 as baselines. Similarly, by replacing all Convs in these baselines with RConvs, we can obtain new RI-CNN models like GD-VGG16, Max-ResNet18, ST-DenseNet40, etc.

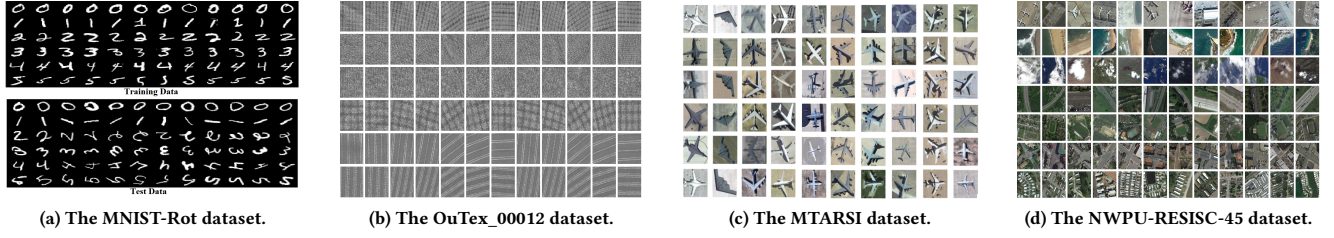


Figure 2: Various datasets used for evaluating the performance of RConvs.

Training protocol: Our experiments are conducted on a Tesla V100 GPU (16G) with the Rocky Linux 8.7 system and PyTorch 2.0.0 framework. All models are trained from scratch without the use of pretrained parameters or data augmentation. This allows us to directly observe the performance improvement brought by RConvs. (1) For experiments on the MNIST-Rot dataset, we train all baselines and RI-CNNs using the Adam optimizer, with an initial learning rate of 10^{-4} , multiplied by 0.8 every 10 epochs. The number of epochs and batch size are both set to 100. (2) For experiments with the Outex_00012, MTARSI, and NWPU-RESISC45 datasets, we also use the Adam optimizer, setting the number of epochs to 100 and batch size to 10. For VGG16, Inception V1, ResNet18/34/50/101, and the RI-CNNs derived from these backbones, the initial learning rate is set to 10^{-4} . For DenseNet40 and its rotation-invariant models, the learning rate is set to 10^{-2} . The value of the learning rate is reduced by a factor of 0.6 every 10 epochs.

4.2 Evaluating Rotation Invariance of RConvs on the MNIST-Rot Dataset

Based on the MNIST dataset, we first validate rotation invariance of seven RConvs. Without using data augmentation, we train seven types of RI-CNNs and their corresponding CNN baseline on the original MNIST training set. Then, we test the classification accuracy of these models on the MNIST-Rot test set. In fact, the MNIST-Rot test set contains 36 subsets, each with 10000 samples having the same rotation angle, such as 10° , 20° , etc. By observing the performance of the models on these subsets, we can analyze their invariance under different rotation angles. The final experimental results are shown in Fig.3. Firstly, we can find that SB-CNN, GD-CNN, ST-CNN, and $\overline{MAX}$ -CNN achieve more than 90% accuracy at almost any rotation angle, of course, with $\overline{MAX}$ -CNN being slightly inferior to the first three. It is evident that their classification curves have a period of 90° . Taking the first period of $0^\circ \sim 90^\circ$ as an example, the model's classification accuracy gradually decreases within the $0^\circ \sim 45^\circ$ interval and gradually increases from $45^\circ \sim 90^\circ$, reaching the performance minimum at a rotation angle of 45° . In fact, when rotating images, interpolation operations are needed; as the rotation angle increases from 0° to 45° , the interpolation error gradually increases, and as the angle continues to increase from 45° to 90° the error gradually decreases. At a rotation angle of 90° , the interpolation operation does not introduce any computational error.

Although the accuracy curve of $\overline{MAX}$ -CNN also shows a certain degree of periodicity, its performance is significantly worse than that of $\overline{MAX}$ -CNN. This indicates that, in most cases, the max operation cannot directly use pixel values to accurately estimate the

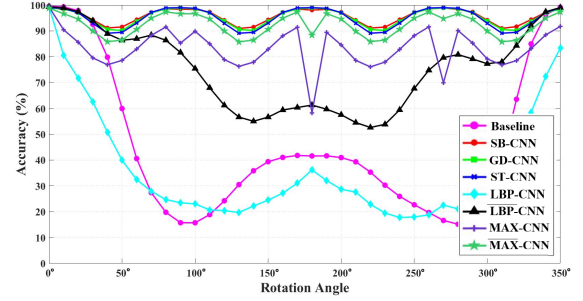


Figure 3: The classification accuracies from seven RConvs on 36 rotated test subsets of MNIST-Rot with specific rotation angles ($0^\circ, 10^\circ, 20^\circ, \dots, 350^\circ$).

orientation of a small region involved in convolution computations, whereas using the difference between adjacent pixels can achieve this. As we mentioned previous, the prerequisite for accurately estimating the orientation of a local region with the maximum operation is the existence of a unique maximum value within the region. However, this condition is not always met. When multiple positions in the region have the same pixel value, which is also the maximum value, we cannot use the maximum operation to achieve rotation invariance of the convolution. Calculating the difference between adjacent pixels can alleviate this problem to some extent. Suppose the pixel values at eight positions (excluding the center point) sampled in a 3×3 region based on the polar coordinate system are 00001111, obviously, there are four identical maximum values in the region. But if we calculate the pixel difference between adjacent pixels, the result becomes 000 – 10001. Clearly, there is only one maximum value 1 at this time. It is for this reason that the performance of $\overline{MAX}$ -CNN is significantly better than that of MAX-CNN.

The poor performance of $\overline{LBP}$ -CNN and LBP-CNN is caused by similar issues. For instance, when the binary pattern calculated within a 3×3 region exhibits periodicity, such as 01010101, 00110011, etc., the LBP operator cannot accurately determine the region's rotation angle. Replacing pixel difference between other positions and the central positions with the difference between adjacent positions can mitigate this issue, but only to a limited extent. Therefore, although $\overline{LBP}$ -CNN performs better than LBP-CNN, it is still significantly inferior to other methods.

Table 1 shows the classification accuracies of seven RConvs, their CNN baseline, and six previously proposed RI-CNN models on the original MNIST test set and MNIST-rot test set. Without

Table 1: The classification on original MNIST and MNIST-rot test sets. Bold stands for best results.

Methods	Input Size	MNIST	MNIST-rot
ORN[39]	32×32	99.42%	80.01%
RotEqNet[21]	28×28	99.26%	73.20%
G-CNN[6]	28×28	99.27%	44.81%
H-Net[33]	32×32	99.19%	92.44%
B-CNN[7]	32×32	97.40%	88.29%
E(2)-CNN[32]	29×29	98.14%	94.37%
Baseline	28×28	99.57%	45.35%
SB-CNN	28×28	99.08%	95.68%
GD-CNN	28×28	98.88%	95.35%
ST-CNN	28×28	99.02%	95.05%
LBP-CNN	28×28	99.34%	36.31%
$\overline{LBP}$ -CNN	28×28	99.36%	75.28%
MAX-CNN	28×28	99.31%	83.32%
$\overline{MAX}$ -CNN	28×28	98.85%	92.33%

relying on data augmentation, Harmonic Network (H-Net)[33], Bessel CNN (B-CNN)[7], and E(2)-CNN [32] also exhibit invariance to arbitrary rotation angles, while ORN[39], RotEqNet[21], and G-CNN[6] only exhibit invariance to specific rotation angles such as 45° or multiples of 90° . We directly train them using the code and protocols provided by their authors. Additionally, we do not compare with methods like STN [14], TI-Pooling [17], as their invariance relies on data augmentation. Our results indicate that: (1) on MNIST-rot, SB-CNN, GD-CNN, and ST-CNN surpass the previous state-of-the-art method E(2)-CNN, increasing the accuracy from 94.37% to 95.68%, 95.35%, and 95.05%, respectively. MAX-CNN achieves an accuracy of 92.33%, comparable to H-Net. The performances of MAX-CNN and $\overline{LBP}$ -CNN are similar to those of ORB and RotEqNet, which exhibit invariance only to specific rotation angles, but significantly better than our baseline (45.35%). The accuracy of LBP-CNN is only 36.31%, even worse than the baseline model. We have previously analyze the reasons. When the computed LBP based on regional information has periodicity, we cannot accurately estimate the orientation of that region. This situation occurs more frequently for binary images. (2) On the original MNIST test set, the baseline model achieves the best result (99.57%). Previous research [9, 24, 25] has shown that RI-CNNs have difficulty distinguishing certain digits, such as "9" and "6", which leads to slightly lower performance on this test set. This problem also results in stronger rotation-invariant models (such as SB-CNN, GD-CNN, and ST-CNN) performing worse than models with weaker rotation invariance (such as ORN, RotEqNet, and MAX-CNN) on the original test set.

4.3 The performance of RIConvs in Real-World Image Classification

By replacing traditional convolutions with different types of RI-Convs in seven common CNN backbones, we obtain a new set of rotation-invariant backbones. This section validates the performance of these rotation-invariant models on three real datasets: Outex_TC_00012, MTARSI, and NWPU-RESISC45. Theoretically, rotation invariance will enhance the feature extraction capability

of the model, enabling it to learn key features of the object based on fewer training samples. To illustrate this point, we gradually reduce the number of samples in the training set while keeping the test set unchanged. Taking MTARSI as an example, we reduced the number of samples in the training set from 4000 (200 per class) to 3000 (150 per class) and 2000 (100 per class). Fig.4 shows the classification accuracies achieved by original backbones and corresponding RI-CNN models on the same test set when they are trained on different sizes of training sets. Analyzing the results obtained on the NWPU-RESISC dataset (see Fig.4c), we can observe the followings: (1) In most cases, RIConvs significantly improve the performance of various CNN backbones. For example, when trained with a the dataset containing 18000 samples (400 per class), the original ResNet18 achieves a classification accuracy of 82.85% on the test set, while all RI-ResNet18 models perform better. Specifically, SB-ResNet18, GD-ResNet, and $\overline{MAX}$ -ResNet18 achieve classification accuracies of 90.63%, 90.92%, and 90.55%, respectively. Even the previously poorly performing LBP-Convs on the MNIST-Rot dataset also achieve an accuracy of 87.02%. (2) The fewer training samples, the more significant the performance improvement brought by RI-Convs. For example, when the number of training samples is 18000, compared to the original ResNet34 with a classification accuracy of 82.82%, using seven RI-ResNet34 (SB, GD, ST, LBP, $\overline{LBP}$, MAX, and $\overline{LBP}$) respectively achieve accuracy improvements of 7.39%, 7.35%, 7.19%, 3.28%, 5.81%, 7.06%, and 7.13%. When the training samples are reduced to 13000, although the performance of all models decreases, the accuracy improvement brought by using seven RIConvs increases to 10.34%, 11.03%, 10.14%, 6.46%, 8.48%, 10.10%, and 10.54%. (3) RI-CNN models implemented based on SB-Conv, GD-Conv, and ST-Conv achieve better accuracy in most cases. This indicates that RIConvs implemented using gradient operators and sorting operations exhibit better invariance. Additionally, compared to the experimental results on the MNIST-Rot dataset, all RIConvs perform relatively well on the NWPU-RESISC45 dataset, including MAX-Conv and LBP-Conv. This is because local patterns in binary images tend to be simpler, making it easier to encounter issues such as multiple identical maximum values and periodicity in computed LBP features, while local patterns in color image data are more complex, reducing the likelihood of encountering such issues. The above observations can also be made based on the experimental results obtained on the other two datasets (see Fig.4a and Fig.4b).

4.4 The Influence of Data Augmentation

Finally, we analyze the impact of data augmentation on the performance of RIConvs. Using different sizes of the MTARSI training set, we obtain the classification accuracies of ResNet18 and seven RI-ResNet18 models on the test set. Unlike the experiment setting in previous sections, this time we apply data augmentation during model training, where each input image was randomly rotated by an angle within $(0, 2\pi]$. The results are shown in Table2. Comparing Table2 with Fig.4b, we can observe a significant improvement in performance for both ResNet18 and all RI-ResNet18 when data augmentation is employed. For example, when the training set contained 3000 images (150 per class), without data augmentation, the classification accuracies of ResNet18 and GD-ResNet18 are 83.14% and 95.30%, respectively. With data augmentation, their accuracies

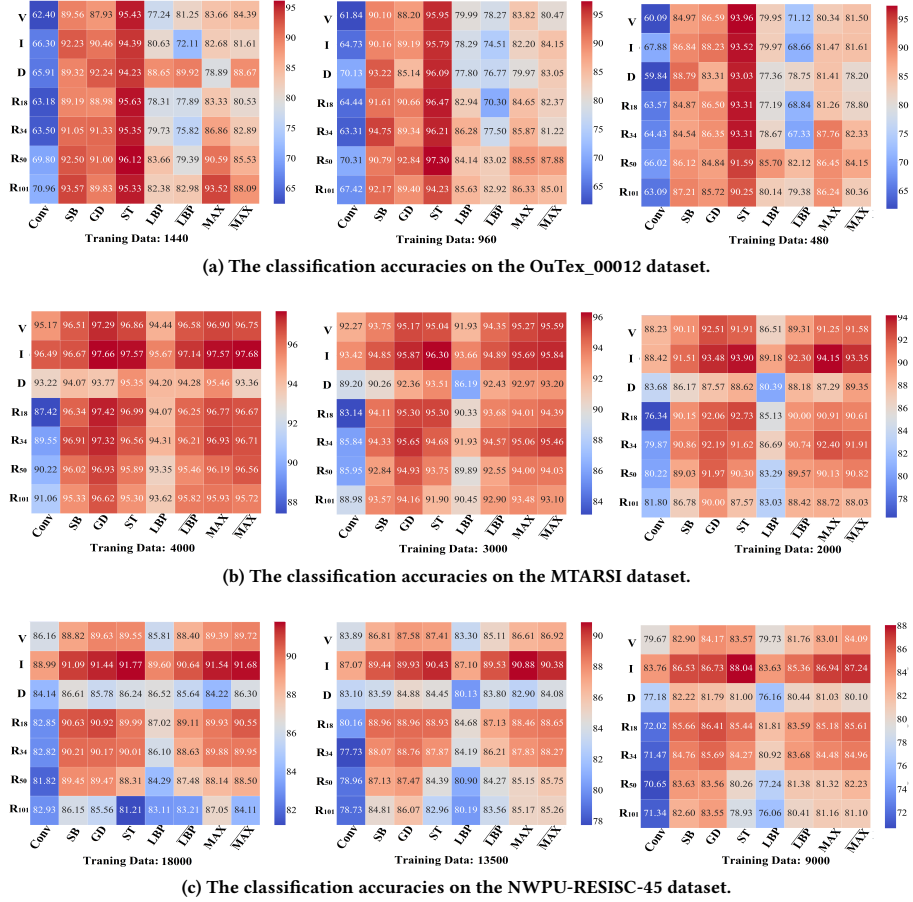


Figure 4: The performance of classical CNN backbones and the corresponding RI-CNN models on different tasks, where V, I, D, R₁₈, R₃₄, and R₅₀ represent VGG16, Inception V1, DenseNet40, ResNet18, ResNet34, ResNet50, and ResNet101, respectively.

improve to 93.83% and 96.97%, demonstrating a notable enhancement. More importantly, we find that even when the models are trained with data augmentation, most RI-ResNet18 models (except LBP-ResNet18) still outperform the original ResNet18, especially when the training data is limited. This once again underscores the significance of constructing rotation-invariant convolutional operations. In fact, mechanism-assured rotation invariance endows CNN models with stronger feature learning capabilities and higher utilization efficiency of learnable parameters. Therefore, even when data augmentation is applied during training, this property further enhances the model's performance.

5 CONCLUSIONS

Based on non-learnable operations, this paper designs a set of convolutional operations that are invariant to arbitrary rotations. They have the same number of learnable parameters as their corresponding traditional convolutions and can be interchangeable. Using the MNIST-Rot dataset, we first verify the invariance of these RConvs under different rotation angles, analyze the shortcomings of some non-learnable operations, and compare new RConvs with existing RI-CNN models. Two types of RConvs designed based on gradient

Table 2: The performance of ResNets and seven RI-ResNet18 trained with data augmentation on the MTARSI test set. Bold stands for best results.

Training Data	20×200=4K	20×150=3K	20×100=2K
ResNet18	93.83%	90.89%	85.99%
SB-ResNet18	95.11%	92.88%	88.36%
GD-ResNet18	96.97%	94.83%	90.06%
ST-ResNet18	96.02%	93.92%	90.32%
LBP-ResNet18	92.99%	88.40%	83.31%
<i>LBP</i> -ResNet18	95.17%	92.83%	88.96%
MAX-ResNet18	95.35%	93.25%	89.31%
<i>MAX</i> -ResNet18	95.30%	93.44%	89.28%

operators achieve state-of-the-art results on this dataset. Subsequently, we combine these RConvs with classic CNN backbones to obtain different RI-CNN models and validate their performance on texture classification, aircraft type recognition, and remote sensing image classification tasks based on the OuTex_00012, MTARSI, and NWPU-RESISC45 datasets. Our experimental results demonstrate that the proposed RConvs can effectively improve the performance

of classic CNN models, especially when the training data is limited, regardless of whether data augmentation is used during training. In the future, we plan to further apply RConvs to object detection and image segmentation tasks.

REFERENCES

- [1] Frimpong Boadu, Hongyuan Cao, and Jianlin Cheng. 2023. Combining protein sequences and structures with transformers and equivariant graph neural networks to predict protein function. *Bioinformatics* 39, 1 (2023), i318–i325.
- [2] Shuyi Chen, Mang Ye, and Bo Du. 2022. Rotation Invariant Transformer for Recognizing Object in UAVs. In *In ACM MM*. 2565–2574.
- [3] Gong Cheng, Junwei Han, and Xiaoqiang Lu. 2017. ‘Remote sensing image scene classification: benchmark and state of the art. *Proc. IEEE* 105, 10 (2017), 1865–1883.
- [4] Gong Cheng, Peicheng Zhou, and Junwei Han. 2016. Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing* 54, 12 (2016), 7405–7415.
- [5] Christopher B. Choy, JunYoung Gwak, Silvio Savarese, and Manmohan Chandraker. 2016. Universal Correspondence Network. In *In NeuIPS*.
- [6] Taco Cohen and Max Welling. 2016. Group equivariant convolutional networks. In *In ICML*. 2990–2999.
- [7] Valentin Delchevalerie, Adrien Bibal, Benoît Fréney, and Alexandre Mayer. 2021. Achieving rotational invariance with Bessel-convolutional neural networks. In *In NeuIPS*.
- [8] Carlos Esteves, Christine Allen-Blanchette, Xiaowei Zhou, and Kostas Daniilidis. 2015. Polar transformer networks. In *In ICLR*. 2017–2025.
- [9] You Hao, Ping Hu, Shirui Li, Jayaram K. Udupa, Yubing Tong, and Hua Li. 2022. Gradient-Aligned convolution neural network. *Pattern Recognition* 122 (2022).
- [10] Zongbo Hao, Tao Zhang, Mingwang Chen, and Kaixu Zou. 2022. RRL: Regional Rotate Layer in Convolutional Neural Networks. In *In AAAI*. 826–833.
- [11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *In CVPR*. 770–778.
- [12] João F. Henriques and Andrea Vedaldi. 2017. Warped convolutions: efficient invariance to spatial transformations. In *In ICML*. 1461–1469.
- [13] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2017. Densely connected convolutional networks. In *In CVPR*. 4700–4708.
- [14] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. 2015. Spatial transformer networks. In *In NeuIPS*. 2017–2025.
- [15] Ruqiao Jiang, Shaohui Mei, Mingyang Ma, and Shun Zhang. 2022. Rotation-invariant feature learning in VHR optical remote sensing images via nested Siamese structure with double center loss. *IEEE Transactions on Geoscience and Remote Sensing* 59, 4 (2022), 3326–3337.
- [16] Maxime W. Lafarge, Erik J. Bekkers, Josien P. W. Pluim, Remco Duits, and Mitko Veta. 2021. Roto-translation equivariant convolutional networks: Application to histopathology image analysis. *Medical Image Analysis* 68 (2021).
- [17] Dmitry Laptev, Nikolay Savinov, Joachim M. Buhmann, and Marc Pollefeys. 2016. TI-Pooling: transformation-invariant pooling for feature learning in convolutional neural network. In *In CVPR*. 289–297.
- [18] Yann Lecun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. 1998. Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 11 (1998), 2278–2324.
- [19] Shutao Li, Weiwei Song, Leyuan Fang, Yushi Chen, Pedram Ghamisi, and Jon Atli Benediktsson. 2019. Deep Learning for Hyperspectral Image Classification: An Overview. *IEEE Transactions on Geoscience and Remote Sensing* 57, 9 (2019), 6690–6709.
- [20] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikainen. 2020. Deep Learning for Generic Object Detection: A Survey. *International Journal of Computer Vision* 128 (2020), 261–318.
- [21] Diego Marcos, Michele Volpi, Nikos Komodakis, and Devis Tuia. 2017. Rotation equivariant vector field networks. In *In ICCV*. 5048–5057.
- [22] Shaohui Mei, Ruqiao Jiang, Mingyang Ma, and Chao Song. 2023. Rotation-invariant feature learning via convolutional neural network with cyclic polar coordinates convolutional layer. *IEEE Transactions on Geoscience and Remote Sensing* 61 (2023), 1–13.
- [23] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. 2022. Image Segmentation Using Deep Learning: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 7 (2022), 3523–3542.
- [24] Hanlin Mo and Guoying Zhao. 2024. RIC-CNN: rotation-invariant coordinate convolutional neural network. *Pattern Recognition* 146 (2024).
- [25] Hanlin Mo and Guoyin Zhao. 2024. Sorting Convolution Operation for Achieving Rotational Invariance. *IEEE Signal Processing Letters* (2024). <https://doi.org/10.1109/LSP.2024.3381909>
- [26] Timo Ojala, Topi Maenpää, Matti Pietikainen, Jaakko Viertola, Juha Kyllönen, and Sami Huovinen. 2002. Outex-new framework for empirical evaluation of texture analysis algorithms. In *In ICPR*. 701–706.
- [27] Kunlun Qi, Chao Yang, Chuli Hu, Yonglin Shen, Shengyu Shen, and Huayi Wu. 2021. Rotation invariance regularization for remote sensing image scene classification with convolutional neural networks. *Remote Sensing* 13, 4 (2021), 569–592.
- [28] Karen Simonyan and Andrew Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *In ICLR*.
- [29] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going Deeper With Convolutions. In *In CVPR*. 1–9.
- [30] Ji Wan, Dayong Wang, Steven Chu Hong Hoi, Pengcheng Wu, Jianke Zhu, Yongdong Zhang, and Jintao Li. 2014. Deep Learning for Content-Based Image Retrieval: A Comprehensive Study. In *In ACM MM*. 157–166.
- [31] Guotai Wang, Wenqi Li, Maria A. Zuluaga, Rosalind Pratt, Premal A. Patel, Michael Aertsen, Tom Doel, Anna L. David, Jan Deprest, Sébastien Ourselin, and Tom Vercauteren. 2018. Interactive Medical Image Segmentation Using Deep Learning With Image-Specific Fine Tuning. *IEEE Transactions on Medical Imaging* 37, 7 (2018), 1562–1573.
- [32] Maurice Weiler and Gabriele Cesa. 2019. General E(2)-equivariant steerable CNNs. In *In NeuIPS*.
- [33] Daniel E. Worrall, Stephan J. Garbin, Daniyar Turmukhambetov, and Gabriel J. Brostow. 2017. Harmonic networks: deep translation and rotation equivariance. In *In CVPR*. 5028–5037.
- [34] Pengcheng Wu, Steven C. H. Hoi, Xia Hao, Wang Dayong Zhao, Peilin, and Chunyan Miao. 2013. Online multimodal deep similarity learning with application to image retrieval.
- [35] Zhize Wu, Shouhong Wan, Xiaofeng Wang, Ming Tan, Le Zou, Xinlu Li, and Yan Chen. 2020. A benchmark data set for aircraft type recognition from remote sensing images. *Applied Soft Computing* 89 (2020).
- [36] Xiaofei Yang, Yunming Ye, Xutao Li, Raymond Y. K. Lau, Xiaofeng Zhang, and Xiaohui Huang. 2018. Hyperspectral Image Classification With Deep Learning Models. *IEEE Transactions on Geoscience and Remote Sensing* 56, 9 (2018), 5408–5423.
- [37] Matthew D. Zeiler and Rob Fergus. 2014. Visualizing and understanding convolutional networks. In *In ECCV*. 818–833.
- [38] Zhongqiu Zhao, Peng Zheng, Shoutao Xu, and Xindong Wu. 2019. Object Detection With Deep Learning: A Review. *IEEE Transactions on Neural Networks and Learning Systems* 30, 11 (2019), 3212–3232.
- [39] Yanzhao Zhou, Qixiang Ye, Qiang Qiu, and Jianbin Jiao. 2017. Oriented response networks. In *In CVPR*. 519–528.