Sorting Convolution Operation for Achieving Rotational Invariance

Hanlin Mo and Guoying Zhao, *Fellow, IEEE*

***Abstract*—The topic of achieving rotational invariance in convolutional neural networks (CNNs) has gained considerable attention recently, as this invariance is crucial for many computer vision tasks. In this letter, we propose a sorting convolution operation (*SConv*), which achieves invariance to arbitrary rotations without additional learnable parameters or data augmentation. It can directly replace conventional convolution operations in a classic CNN model to achieve the model's rotational invariance. Based on MNIST-rot dataset, we first analyze the impact of convolution kernel size, sampling grid and sorting method on *SConv*'s rotational invariance, and compare our method with previous rotation-invariant CNN models. Then, we combine *SConv* with VGG, ResNet and DenseNet, and conduct classification experiments on texture and remote sensing image datasets. The results show that *SConv* significantly improves the performance of these models, especially when training data is limited.**

***Index Terms*—Image rotation, rotational invariance, convolutional neural network, sorting operation, interpolation.**

I. Introduction

FEATURE extraction is one of the core tasks in computer vision. Ideal image features should be invariant to spatial deformations caused by imaging geometry, which ensures that they can capture intrinsic information of images. In many practical applications, such as object detection and image classification, we need to consider images with arbitrary orientations. Actually, numerous rotation-invariant handcrafted features have been developed in past decades [1], [2], [3], [4], [5], [6]. Since 2012, deep neural networks, especially convolutional neural networks (CNNs), have been proven to be more effective than most handcrafted features in various computer vision tasks. Nonetheles, convolution operations do not have rotational invariance. Even if an input is slightly rotated, CNNs may not be able to recognize it correctly. To address this, a direct approach is to train a CNN with data augmentation. However, it has some drawbacks, such as learning more redundant weights, and reducing the interpretability of CNNs [7], [8]. Hence, recent research has aimed to design new network architectures to achieve rotation invariance of CNNs. Based on the concepts of group theory and steerability, some researchers design rotation-equivariant CNNs represented by Group Equivariant Convolutional Network (G-CNN) [9] and E(2)-Equivariant Steerable CNN (E(2)-CNN) [10]. These networks achieve the final output's rotation invariance by ensuring the rotation equivariance of intermediate group representations [11], [12], [13]. Other research directly addresses rotation invariance, proposing various types of rotation-invariant CNN models using methods such as orientation assignment, polar/log-polar transform, and multi-orientation feature extraction. These models include Spatial Transformer Network (STN) [14], Polar Transformer Network [15], Oriented Response Network (ORN) [16], Rotation-Invariant Coordinate CNN (RIC-CNN) [17], and so on [18], [19], [20], [21]. Although they have been used in different practical tasks [22], [23], [24], [25], [26], [27], these existing rotation-invariant/equivariant CNNs have three major limitations: **1)** Most methods are invariant to specific rotation angles rather than arbitrary angles [9], [16], [20]. Some of them, like RIC-CNN [17], are only invariant to rotations around image center. **2)** Some methods require extra trainable parameters and rely on data augmentation when training [14], [19], [22], [27]. **3)** Many rotation-invariant convolution operations are really complex [8], [9], [16], [20]. They cannot be directly substituted for traditional convolutions, making it challenging to incorporate them into popular CNN backbones.

The goal of this letter is to address these limitations. Our contributions can be summarized as follows: **1)** Inspired by some handcrafted features of texture images [28], [29], [30], we propose a Sorting Convolution (*SConv*) which achieves invariance to arbitrary rotation angles without data augmentation. By substituting all standard convolutions in a CNN model with the corresponding *SConv*, we can obtain a Sorting Convolutional Neural Network (SCNN). **2)** We train SCNN on original MNIST training set without data augmentation, evaluate its performance on MNIST-rot test set, and analyze the impact of convolution kernel size, sampling grid, and sorting method on its rotational invariance. In comparison to previous rotation-invariant CNN models, our SCNN achieves state-of-the-art result. **3)** We integrate *SConv* into classical CNN backbones and perform classification experiments on texture and remote sensing image datasets. Our results show that *SConv* significantly increases the classification accuracy of these models, particularly when training data is limited.

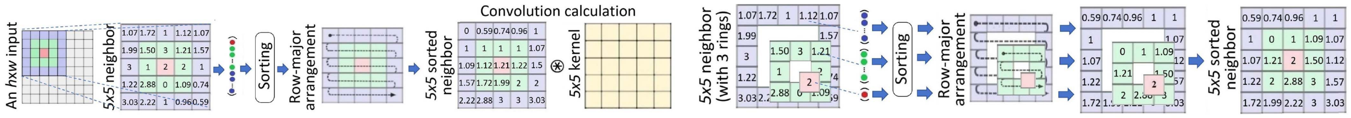
Manuscript received 24 December 2023; revised 18 March 2024; accepted 20 March 2024. Date of publication 26 March 2024; date of current version 2 May 2024. This work was supported in part by the Research Council of Finland Academy Professor Project EmotionAI under Grant 336116 and Grant 345122 and in part by the University of Oulu & Research Council of Finland Profi 7 under Grant 352788. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. John Ball. *(Corresponding author: Guoying Zhao.)*

Hanlin Mo is with the Center for Machine Vision and Signal Analysis, University of Oulu, 90014 Oulu, Finland (e-mail: hanlin.mo@oulu.fi).

Guoying Zhao is with the School of Informatics, Hunan University of Chinese Medicine, Changsha 410208, China, and also with the Center for Machine Vision and Signal Analysis, University of Oulu, 90014 Oulu, Finland (e-mail: guoying.zhao@oulu.fi).

Our code can be downloaded from https://github.com/HanlinMo/Sorting-Convolution-Operation-for-Achieving-Rotational-Invariance.

Digital Object Identifier 10.1109/LSP.2024.3381909



(a) Taking 5×5 *SConv* as an example, we describes how to sort values and then arrange them in row-major order within a 5×5 neighborhood.

(b) The explanation of using the ring sorting method within a 5×5 neighborhood (with three rings).

Fig. 1. The computational process of the proposed *SConv* and ring sorting method. (The numbers in the figure represent the values of the input at sampling positions.).

II. Methodology

A. Sorting Convolution Operation

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 two-dimensional function $F(X) : \Omega \rightarrow \mathbb{R}$, where the domain $\Omega = \{1, 2, \ldots, h\} \times \{1, 2, \ldots, w\}$. Then, a conventional convolution operation *Conv* acting on a given position $X_0 \in \Omega$ can be expressed as below

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

Here, W is a $(2n+1) \times (2n+1)$ learnable kernel, n is a non-negative integer, and P enumerates all sample positions on the square grid $\mathcal{S} = \{-n, -n+1, \ldots, n\} \times \{-n, -n+1, \ldots, n\}$. For example, when W is a 3×3 kernel, we have $\mathcal{S} = \{(-1,-1), (-1,0), \ldots, (0,1), (1,1)\}$, which contains 9 sample positions. We only consider odd-sized W because the shift issue occurs in even-sized ones [31].

Let $G(Y)$ be a rotated version of $F(X)$, that is, $G(Y) = F(R_{-\theta} Y)$, where $R_{-\theta}$ is a 2×2 rotation matrix and $\theta \in [0, 2\pi)$ represents the rotation angle. Supposing that Y_0 is the corresponding position of X_0 after rotation, the convolution operation at Y_0 is

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

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

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

By substituting (3) into (2), we can find $Conv(Y_0, G(Y)) \neq Conv(X_0, F(X))$, which means that *Conv* is not invariant to two-dimensional rotation.

In fact, assuming that $\{R_{-\theta} P\}_{P \in \mathcal{S}} = \mathcal{S}$, we have $\{G(Y_0 + P)\}_{P \in \mathcal{S}} = \{F(X_0 + R_{-\theta} P)\}_{P \in \mathcal{S}} = \{F(X_0 + P)\}_{P \in \mathcal{S}}$. This indicates that the input values used for the convolution operation at positions X_0 and Y_0 are the same, but with different arrangements. Obviously, if all values in $\{G(Y_0 + P)\}_{P \in \mathcal{S}}$ and $\{F(X_0 + P)\}_{P \in \mathcal{S}}$ are sorted in ascending order separately, the two resulting sorted sequences are exactly the same.

Thus, we can first sort all values in $\{F(X_0 + P)\}_{P \in \mathcal{S}}$ and arrange the sorted sequence in row-major order on the square grid $\mathcal{S}$, and then define *SConv* as

$$SConv(X_0, F(X)) = \sum_{P \in \mathcal{S}} W(P) \cdot F^s(X_0 + P) \tag{4}$$

where $F^s(X_0 + P)$ represents the new value at $(X_0 + P)$ after the sorting and row-major arrangement. In Fig. 1(a), we show the computation process of 5×5 *SConv*. Since $F^s(X_0 + P) = G^s(Y_0 + P)$ for any $P \in \mathcal{S}$, we have $SConv(Y_0, G(Y)) = SConv(X_0, F(X))$, meaning that *SConv* is invariant to arbitrary rotations. Moreover, the sorting operation and row-major arrangement do not require any parameters, so the number of learnable parameters in *SConv* is the same as in *Conv*.

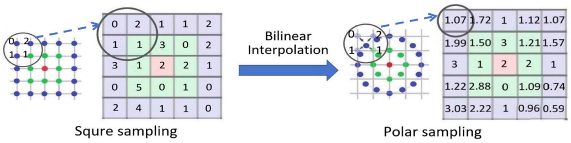


Fig. 2. Converting sampling positions on the square grid to sampling positions on the polar grid within a 5×5 neighborhood. (The numbers in the figure represent the values of the input at sampling positions.).

B. Ring Sorting and Polar Grid

It should be noted that the sorting operation disrupts the local structure of $F(X)$ in the $(2n+1) \times (2n+1)$ neighborhood of X_0, and reduce the discriminability of features. To address this issue to some extent, we employ a ring sorting method. As shown in Fig. 1(b), this method sorts and arranges values in different rings centered at X_0, with no interference between different rings. Clearly, it still ensures rotational invariance while retaining some structural information between different rings within the neighborhood. In fact, some early researches used the ring sorting to construct rotation-invariant handcrafted features for texture images [28], [29], [30]. However, to the best of our knowledge, this is the first time this method has been applied to CNNs.

Additionally, as mentioned in Section II-A, the rotational invariance of *SConv* is based on an assumption: $\{R_{-\theta} P\}_{P \in \mathcal{S}} = \mathcal{S}$. In fact, for discrete convolutions based on a square grid, this assumption only holds true when rotation angle $\theta = k \cdot 90^\circ$ (k is an integer). To address this issue, a polar coordinate system centered at X_0 is established. As shown in Fig. 2, on a $(2n+1) \times (2n+1)$ neighborhood, we evenly sample $8r$ positions on a circumference with radius r centered at X_0, where $r = 1, 2, \ldots, n$. Bilinear interpolation is used to obtain the values of $F(X)$ at these positions. When the rotation angle $\theta = k \cdot 360^\circ / (8r)$, the $8r$ positions on the circle with radius r coincide before and after rotation, where k is an integer. Compared to the square grid, the polar grid better ensures the validity of the assumption, thereby better guaranteeing the rotational invariance of *SConv*.

C. The Implementation of Sorting CNN

Given an input $F(X)$ of size $h \times w$, *Conv* defined by (1) produces an output of size $h \times w$ when the stride is set to 1 and padding is performed. For the corresponding *SConv* defined by (4), we first need to sort the input values within a $(2n+1)\times(2n+1)$ neighborhood for each position $X \in \{1,2,\ldots,h\}\times\{1,2,\ldots,w\}$, and then concatenate all the sorted neighborhoods to form a new input with size $((2n+1)\cdot h)\times((2n+1)\cdot w)$. Next, we perform a $(2n+1)\times(2n+1)$ convolution on this input with a stride of $(2n+1)$ and no padding is applied. Obviously, the output size of *SConv* is still $h \times w$, the same as the output of *Conv*. Hence, *SConv* and *Conv* can be swapped.

By replacing all *Conv* in a standard CNN with the corresponding *SConv*, we can create a SCNN. When simultaneously inputting an image and its rotated version into SCNN, the two features obtained in each *SConv* layer satisfy the same rotation relationship. If we use max or average pooling operations to reduce the spatial resolution of the output of the last *SConv* layer to 1×1, the resulting feature is invariant to arbitrary rotations, and can be used as input for other network structures, such as fully connected layers.

III. Experiments

A. Experiment Setup

Datasets: **MNIST** [32] has 70000 28×28 handwritten digit images (0-9), with 60000 for training and 10000 for testing. 10000 training images are randomly selected for validation. Each test image is rotated from 0° to 350° every 10°, resulting in 360000 rotated test images. The new test set is called *MNIST-rot* and used to verify a CNN model's rotational invariance. **OutexTC00012** [33] contains 24 texture classes and 9120 grayscale images of size 128×128. For each class, 20 texture surfaces are captured under three lighting conditions ("inca", "t184" and "horizon") as training images. Then, they are captured as test images from 8 different rotation angles ($5^\circ\sim 90^\circ$) under "t184" and "horizon" lighting conditions. Thus, the training set contains $24\times 20\times 3 = 1440$ images, and the test set contains $24\times 20\times 2\times 8 = 7680$ images. **NWPU-RESISC45** [34] is a dataset for remote sensing image scene classification. It contains 31500 RGB images of size 256×256 divided into 45 scene classes, each class containing 700 images. We resize all images to 128×128 and randomly select 400 images from each class as training images, with the remaining images used as test images. Due to the arbitrary shooting angles, there are rotation variations present in many classes, such as "bridge" and "ground track field".

Models and Training Protocol: We initially design a CNN baseline with six *Conv* layers, having 32/32/64/64/128/128 channels, respectively. We apply 2×2 max pooling after the second and fourth layers, and use 7×7 average pooling after the final convolution layer. Then, the feature vector is fed into a fully connected layer with ten units. The kernel size for the last two convolution layers is 3×3, while for the first four layers, the kernel size is the same $K\times K$, where $K\in\{3,5,7\}$. By replacing each *Conv* in the baseline with the corresponding *SConv*, we can obtain a SCNN model. When implementing *SConv*, we have options for square grid (S) or polar grid (P), as well as global sorting (G) or ring sorting (R). This results in 12 different SCNNs ($\{S,P\}\times\{G,R\}\times\{3,5,7\}$). For example, P-R-5 indicates that the first four layers use 5×5 *SConv* with polar grid and ring sorting. We train all these SCNNs on MNIST traning dataset with Adam optimizer, while the initial learning rate is 10^{-4}, multiplied by 0.8 every 10 epochs. The number of epochs and the batch size are 100.

TABLE I
The Classification on MNIST and MNIST-Rot

Methods	Input Size	MNIST	MNIST-rot
ORN[16]	32×32	99.42%	80.01%
RotEqNet[20]	28×28	99.26%	73.20%
G-CNN[9]	28×28	99.27%	44.81%
H-Net[8]	32×32	99.19%	92.44%
B-CNN[21]	32×32	97.40%	88.29%
E(2)-CNN[10]	29×29	98.14%	94.37%
Baseline	28×28	**99.43%**	44.53%
SCNN	28×28	99.04%	**95.05%**

Bold stands for best results.

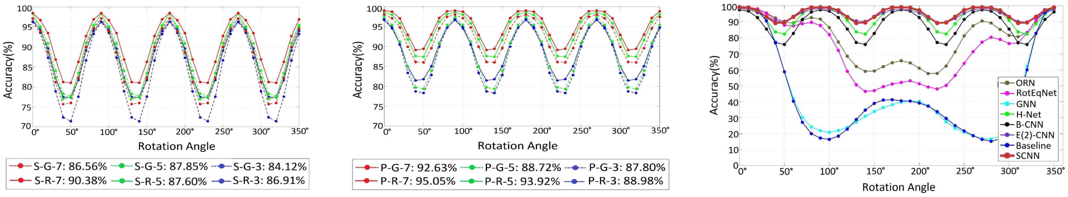
To demonstrate the ease of integrating *SConv* with commonly used CNN backbones, we select ResNet18/34/50/101 [35], VGG16 [36] and DenseNet40 [37] as baselines. By replacing all *Conv* in these models with *SConv*, we obtain RI-ResNet18/34/50/101, RI-VGG16 and RI-DenseNet40. All of them are trained on OutexTC00012 and NWPU-RESISC45, respectively. Again, the Adam optimizer is used, and the training process involves 100 epochs with a batch size of 10. The initial learning rate is set to 10^{-2} for DenseNet40 and RI-DenseNet40, while it is 10^{-3} for other models. We reduce it by a factor of 0.6 every 10 epochs.

Our experiments are performed on a Tesla V100 GPU (16 G) upon Rocky Linux 8.7 system and PyTorch 2.0.0 framework. All models are trained from scratch without using pretrained parameters or data augmentation. This allows us to directly observe the performance improvement brought by *SConv*.

B. Results on MNIST-Rot

First, we test rotational invariance of 12 SCNNs with different convolution kernel sizes, sampling grids and sorting methods on the MNIST-rot. This test set contains 36 subsets, each containing 10000 samples with the same rotation angle $(0^\circ, 10^\circ, \ldots, 350^\circ)$. Fig. 3(a) illustrates the classification accuracy of six SCNNs using square grid on each subset, while Fig. 3(b) displays the accuracy of six SCNNs using polar grid. Our findings are as follows: **1) Polar grid better than square grid.** The accuracy curves in Fig. 3(b) show significant overall improvement compared to Fig. 3(a). For example, S-R-5 just achieves 87.60% accuracy, whereas P-R-5 achieves 93.92% on the entire MNIST-rot. This aligns with our analysis in Section II-B. **2) Ring sorting better than global sorting.** Notably, P-R-7 achieves the highest accuracy of 95.05%, surpassing P-G-7's accuracy of 92.63% by 2.42%. This is because the ring sorting partially preserves spatial information within a convolution region. **3) Large kernel better than small kernel.** Those SCNNs with larger kernel sizes yield better results, especially when combined with the ring sorting. For example, the accuracies obtained by P-R-3, P-R-5, and P-R-7 are 88.98%, 93.92%, and 95.05%, respectively.

Fig. 3(c) and Table I show the classification accuracies of P-S-7, the corresponding baseline model, and six previous rotation-invariant CNN models on the original MNIST test set and MNIST-rot. Similar to SCNN, Harmonic Network (H-Net) [8], Bessel CNN (B-CNN) [21], and E(2)-CNN [10] also have the invariance to arbitrary rotation angles even without data augmentation. In contrast, ORN [16], Rotation Equivariant Vector Field Network (RotEqNet) [20], and G-CNN [9] are only invariant to specific rotation angles like multiples of 45° or 90°. These models are trained using the protocols from their authors. We do not select STN [14], TI-Pooling [19], and



(a) Six SCNN models using square grid. (b) Six SCNN models using polar grid. (c) PG-RS-7 and some models for comparison.

Fig. 3. The classification accuracies from SCNNs and other rotation-invariant CNN models on the MNIST-rot test set.

TABLE II
THE CLASSIFICATION ON OUTEXTC00012

Training Data	$24 \cdot 40 = 1440$	$24 \cdot 40 = 960$	$24 \cdot 40 = 480$
ResNet18	63.18%	64.44%	63.57%
RI-ResNet18	**95.63%**	**96.47%**	**93.31%**
ResNet34	63.50%	63.31%	64.43%
RI-ResNet34	**95.35%**	**96.21%**	**93.31%**
ResNet50	69.80%	70.31%	66.02%
RI-ResNet50	**96.12%**	**97.30%**	**91.59%**
ResNet101	70.96%	67.42%	63.09%
RI-ResNet101	**95.33%**	**94.23%**	**90.25%**
VGG16	62.40%	61.84%	60.09%
RI-VGG16	**95.43%**	**95.95%**	**93.96%**
DenseNet40	65.91%	70.13%	59.84%
RI-DenseNet40	**94.23%**	**96.09%**	**93.03%**

several methods utilizing rotation-invariant loss functions [22], [38] for comparison, because their invariance relies on data augmentation. Our experimental results indicate the following: **1)** On MNIST-rot, P-S-7 surpasses the previous state-of-the-art method, E(2)-CNN, by improving the accuracy from 94.37% to 95.05%. Additionally, the performance of P-S-7, H-Net, B-CNN, and E(2)-CNN significantly outperforms ORN, RotEqNet, and G-CNN. This highlights the importance of achieving invariance of CNNs under arbitrary rotations. Furthermore, due to the inability to learn rotational invariance from the training data, even though Baseline and SCNN have an equal number of learnable parameters, Baseline just achieves 44.53% accuracy. **2)** On the original MNIST test set, Baseline achieves the best result (99.43%). Previous research [17] has indicated that rotation-invariant CNNs struggle to distinguish between some digits, like “9” and “6”, which contributes to their slightly lower performance on this test set.

C. Results on OutexTC00012 and NWPU-RESISC45

We evaluate six commonly used CNN backbones and their corresponding rotation-invariant models on the OutexTC00012 dataset. The rotation-invariant models are obtained by replacing *Conv* with *SConv* (using polar grid and ring sorting). The classification accuracy is displayed in the first column of Table II. Our rotation-invariant CNNs exhibit significantly higher accuracy compared to their baseline counterparts. For example, RI-ResNet18 outperforms ResNet18 by a substantial margin of 32.45%. We subsequently reduce the number of training images from 1440 to 960 (only using training images captured under “inca” and “t184” lighting conditions) and 480 (“inca” only). We train twelve models on these smaller training sets and evaluate their performance on the original test set. The results are shown in the second and third columns of Table II. Remarkably, even when the training set excludes certain lighting conditions, all rotation-invariant models achieve accuracy exceeding 90%. This is because lighting variations do not disrupt the local structure of textures, and the rotational invariance of *SConv* enables it to better extract essential information about local texture structures. Fig. 4 shows the visualization results of the feature maps obtained from the first *SConv* layer in RI-ResNet18. It can be observed that the feature maps rotate with the input image, which is ensured by *SConv*’s rotational invariance. Further, we conduct classification experiments on the NWPU-RESSC45 dataset, and also reduce the training set size to 13500 and 9000 (randomly selecting 300 and 200 images from each category, respectively). Table III presents the classification accuracies of these models on the test set. Clearly, our rotation-invariant models continue to outperform the corresponding baselines significantly, with a wider gap as the training data decreases. For instance, when the number of training images is reduced from 18000 to 13500 and 9000, the accuracy difference between RI-ResNet18 and ResNet18 increases from 7.14% to 8.77% and 13.42%, respectively.

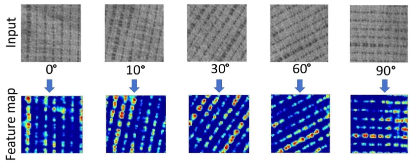


Fig. 4. Some visualization results of the feature maps obtained from the first *SConv* layer in RI-ResNet18.

TABLE III
THE CLASSIFICATION ON NWPU-RESISC45

Training Data	$45 \cdot 400 = 18000$	$45 \cdot 300 = 13500$	$45 \cdot 200 = 9000$
ResNet18	82.85%	80.16%	72.02%
RI-ResNet18	**89.99%**	**88.93%**	**85.44%**
ResNet34	82.82%	77.73%	71.47%
RI-ResNet34	**90.17%**	**87.87%**	**84.27%**
ResNet50	81.82%	78.96%	70.65%
RI-ResNet50	**88.31%**	**84.39%**	**80.26%**
ResNet101	82.93%	78.73%	71.34%
RI-ResNet101	**81.21%**	**82.96%**	**78.93%**
VGG16	86.16%	83.89%	79.67%
RI-VGG16	**89.55%**	**87.41%**	**83.57%**
DenseNet40	84.14%	83.10%	77.18%
RI-DenseNet40	**86.24%**	**84.45%**	**81.00%**

IV. CONCLUSION

We develop a *SConv* to achieve rotational invariance in CNNs without additional learnable parameters or data augmentation. Using the MNIST-rot dataset, we analyze the impact of different factors on *SConv*’s rotational invariance and compare its performance with other rotation-invariant CNNs. *SConv* can directly replace conventional convolution operation. We combine it with classical CNN backbones, and conduct classification experiments on popular image datasets. Our results show *SConv* excels in these tasks, especially when training data is limited.

REFERENCES

[1] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, 2004.

[2] T. Ojala, M. Pietikainen, and T. Maenpaa, “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[3] X. Shi, A.-L. R. Castro, R. Manduchi, and R. Montgomery, “Rotational invariant operators based on steerable filter banks,” *IEEE Signal Process. Lett.*, vol. 13, no. 11, pp. 684–687, Nov. 2006.

[4] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, “Speeded-up robust features (SURF),” *Comput. Vis. Image Understanding*, vol. 10, no. 3, pp. 346–359, 2008.

[5] T. Chakraborti, B. McCane, S. Mills, and U. Pal, “LOOP descriptor: Local optimal-oriented pattern,” *IEEE Signal Process. Lett.*, vol. 25, no. 5, pp. 635–639, May 2018.

[6] H.-L. Mo, Q. Li, Y. Hao, H. Zhang, and H. Li, “A rotation invariant descriptor using multi-directional and high-order gradients,” in *Proc. Chin. Conf. Pattern Recognit. Comput. Vis.*, 2018, pp. 372–383.

[7] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks,” in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 818–833.

[8] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow, “Harmonic networks: Deep translation and rotation equivariance,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5028–5037.

[9] T. Cohen and M. Welling, “Group equivariant convolutional networks,” in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 2990–2999.

[10] M. Weiler and G. Cesa, “General E(2)-equivariant steerable CNNs,” in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 14357–14368.

[11] M. Weiler, F. A. Hamprecht, and M. Storath, “Learning steerable filters for rotation equivariant CNNs,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 849–858.

[12] S. C. Mouli and B. Ribeiro, “Neural networks for learning counterfactual G-invariances from single environments,” in *Proc. Int. Conf. Learn. Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=HktRlUlAZ

[13] J. Lee, B. Kim, S. Kim, and M. Cho, “Learning rotation-equivariant features for visual correspondence,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 21887–21897.

[14] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, “Spatial transformer networks,” in *Proc. Neural Inf. Process. Syst.*, 2015, pp. 2017–2025.

[15] C. Esteves, C. Allen-Blanchette, X.-W. Zhou, and K. Daniilidis, “Polar transformer networks,” in *Proc. Int. Conf. Learn. Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=7t1FcJUWhi3

[16] Y.-Z. Zhou, Q.-X. Ye, Q. Qiu, and J.-B. Jiao, “Oriented response networks,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 519–528.

[17] H.-L. Mo and G.-Y. Zhao, “RIC-CNN: Rotation-invariant coordinate convolutional neural network,” *Pattern Recognit.*, vol. 146, 2024, Art. no. 109994.

[18] L. Sifre and S. Mallat, “Rotation, scaling and deformation invariant scattering for texture discrimination,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 1233–1240.

[19] D. Laptev, N. Savinov, J. M. Buhmann, and M. Pollefeys, “TI-Pooling: Transformation-invariant pooling for feature learning in convolutional neural networks,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 289–297.

[20] D. Marcos, M. Volpi, N. Komodakis, and D. Tuia, “Rotation equivariant vector field networks,” in *Proc. Int. Conf. Comput. Vis.*, 2017, pp. 5048–5057.

[21] V. Delchevalerie, A. Bibal, B. Frénay, and A. Mayer, “Achieving rotational invariance with bessel-convolutional neural networks,” in *Proc. Neural Inf. Process. Syst.*, 2021. [Online]. Available: https://proceedings.neurips.cc/paper/2021/hash/f18224a1adfb7b3dbff668c9b655a35a-Abstract.html

[22] G. Cheng, P.-C. Zhou, and J.-W. Han, “Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images,” *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 12, pp. 7405–7415, Dec. 2016.

[23] K. Qi et al., “Polycentric circle pooling in deep convolutional networks for high-resolution remote sensing image recognition,” *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 13, pp. 632–641, 2020.

[24] K. Qi, C. Yang, C. Hu, Y. Shen, S. Shen, and H. Wu, “Rotation invariance regularization for remote sensing image scene classification with convolutional neural networks,” *Remote Sens.*, vol. 13, no. 4, 2021, Art. no. 569.

[25] R. Jiang, S. Mei, M. Ma, and S. Zhang, “Rotation-invariant feature learning in VHR optical remote sensing images via nested siamese structure with double center loss,” *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 4, pp. 3326–3337, Apr. 2021.

[26] S. Mei, R. Jiang, M. Ma, and C. Song, “Rotation-invariant feature learning via convolutional neural network with cyclic polar coordinates convolutional layer,” *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023, Art. no. 5600713.

[27] F. Zhang, H.-Y. Bian, Z. Lv, and Y.-F. Zhai, “Ring-masked attention network for rotation-invariant template-matching,” *IEEE Signal Process. Lett.*, vol. 30, pp. 289–293, 2023.

[28] L. Liu, P. Fieguth, G.-Y. Kuan, and H.-B. Zha, “Sorted random projections for robust texture classification,” in *Proc. Int. Conf. Comput. Vis.*, 2011, pp. 391–398.

[29] L. Liu, P. Fieguth, D. Clausi, and G.-Y. Kuan, “Sorted random projections for robust rotation-invariant texture classification,” *Pattern Recogn.*, vol. 45, no. 6, pp. 2405–2418, 2012.

[30] T.-C. Song, L.-L. Xin, C.-Q. Gao, G. Zhang, and T. -Q. Zhang, “Grayscale-inversion and rotation invariant texture description using sorted local gradient pattern,” *IEEE Signal Process. Lett.*, vol. 25, no. 5, pp. 625–629, May 2018.

[31] S. Wu, G.-R. Wang, P. Tang, F. Chen, and L. -P. Shi, “Convolution with even-sized kernels and symmetric padding,” in *Proc. Neural Inf. Process. Syst.*, 2019, pp. 1194–1205.

[32] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” in *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[33] T. Ojala, T. Maenpaa, M. Pietikainen, J. Viertola, J. Kyllonen, and S. Huovinen, “Outex-new framework for empirical evaluation of texture analysis algorithms,” in *Proc. Int. Conf. Pattern Recognit.*, 2002, pp. 701–706.

[34] G. Cheng, J.-W. Han, and X.-Q. Lu, “Remote sensing image scene classification: Benchmark and state of the art,” *Proc. IEEE*, vol. 105, no. 10, pp. 1865–1883, Oct. 2017.

[35] K.-M. He, X.-Y. Zhang, S.-Q. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[36] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *Proc. Int. Conf. Learn. Representations*, 2015. [Online]. Available: https://arxiv.org/abs/1409.1556

[37] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 4700–4708.

[38] G. Cheng, J.-W. Han, P.-C. Zhou, and D. Xu, “Learning rotation-invariant and fisher discriminative convolutional neural networks for object detection,” *IEEE Trans. Image Process.*, vol. 28, no. 1, pp. 265–278, Jan. 2019.