entropy

MDPI

Article

Spectral Convolution Feature-Based SPD Matrix Representation for Signal Detection Using a Deep Neural Network

Jiangyi Wang [1,†], Min Liu [2,†], Xinwu Zeng [1,*] and Xiaoqiang Hua [1,*]

1 College of Meteorology and Oceanography, National University of Defence Technology, Changsha 410073, China; wjy05460731@163.com
2 College of Computer, National University of Defence Technology, Changsha 410073, China; liumin_nudt@163.com
* Correspondence: xinwuzeng@nudt.edu.cn (X.Z.); huaxiaoqiang12@nudt.edu.cn (X.H.)
† Co-first author, these authors contributed equally to this work.

Received: 30 July 2020; Accepted: 26 August 2020; Published: 28 August 2020

Abstract: Convolutional neural networks have powerful performances in many visual tasks because of their hierarchical structures and powerful feature extraction capabilities. SPD (symmetric positive definition) matrix is paid attention to in visual classification, because it has excellent ability to learn proper statistical representation and distinguish samples with different information. In this paper, a deep neural network signal detection method based on spectral convolution features is proposed. In this method, local features extracted from convolutional neural network are used to construct the SPD matrix, and a deep learning algorithm for the SPD matrix is used to detect target signals. Feature maps extracted by two kinds of convolutional neural network models are applied in this study. Based on this method, signal detection has become a binary classification problem of signals in samples. In order to prove the availability and superiority of this method, simulated and semi-physical simulated data sets are used. The results show that, under low SCR (signal-to-clutter ratio), compared with the spectral signal detection method based on the deep neural network, this method can obtain a gain of 0.5–2 dB on simulated data sets and semi-physical simulated data sets.

Keywords: SPD matrix learning; local features; deep neural networks; signal detection

1. Introduction

In recent years, convolutional neural networks have shown excellent performance in visual tasks. Starting from AlexNet [1], many successful convolutional neural network models have been developed, such as VGG [2], GoogLeNet [3], ResNet [4], and DenseNet [5]. With the help of hierarchical convolution kernel and nonlinear computation, deep neural networks can extract more discriminative local features for visual representations. Literature [6] proves that the convolution feature is a set of local features related to objects, all of which together describe the visual characteristics of objects. Rich feature descriptions draw researchers' great interest in deep convolution features. To enhance feature learning, a lot of aggregation operations based on deep convolution features are proposed, such as max pooling [7], cross-dimensional pooling [8], sum pooling [6], and bilinear pooling [9]. These operations are in high-dimensional Euclidean spaces.

For the past few years, the SPD (symmetric positive definite) matrix has drawn considerable attention because of its powerful representation ability. Based on non-Euclidean Riemannian geometric properties, the SPD matrix is more suitable for capturing the desired data distribution properties. The SPD matrix manifold is widely used in medical imaging [10], sonar signal processing [11], radar signal processing [12], face recognition [13,14], action recognition [15,16], object or image classification [17,18],

and transfer learning [19]. The nonlinear distribution of the SPD matrix in the Riemannian manifold can be measured on a non-Euclidean scale, based on the geodetic distance.

Transforming convolution features into SPD matrices is a problem of converting first-order data into second-order statistic information. Gaussian distribution and covariance matrix are widely used SPD matrix representations that transform first-order features into second-order statistical information [20,21]. However, the dimension of the convolution feature is often higher than the number of features. In this case, both the Gaussian distribution and the covariance matrix are singular matrices, and the data distribution does not conform to the manifold property of the SPD matrix. It is inappropriate to exploit Riemannian metrics to distinguish the information difference. To solve this problem, a small perturbation can be added to the covariance [22]. Besides, kernel functions can characterize nonlinear relationships of data. In [23], positive definite kernel based on Gaussian radial basis function is defined on manifold, and Euclidean space algorithms, such as support vector machine (SVM) and principal component analysis (PCA) are extended to Riemannian manifold with the help of the proposed positive definite kernel, based on Gaussian radial basis function. With the advent of deep matrix learning [24–26], literature [27] proposes a deep SPD matrix learning model, which exploits RBF kernel function to aggregate convolution features into SPD matrices. Their ultimate goal is to convert the SPD matrix from a Riemannian manifold to another more distinctive manifold.

Based on the above empirical observations, we introduce a method to transform spectral convolution features into SPD matrices through an RBF kernel function, and then detect the target signal using a deep SPD matrix learning network. Two modules are included in this method. The first module uses a nonlinear function to transform convolution features of the spectrum into SPD matrices. All generated SPD matrices become processing objects for the second module. The second module utilizes SPDnet [25]. It is a deep matrix learning network consisting of a SPD matrix transformation layer, SPD matrix nonlinear processing layer, log-Euclidean projection layer, and fully connected (FC) layers. SPD matrices become more compact and discriminative after being passed through the SPD matrix transformation layer and SPD matrix nonlinear processing layer. After using the log-Euclidean metric, the difference in geodetic distance between SPD matrices generated by different samples will be projected into the Euclidean space, and the dichotomization problem based on the Riemannian manifold will be solved by the FC layer based on the Euclidean space. The convolution features of the spectrogram correspond to the information annotation of the spectrogram in advance. Theoretically, SPD matrices based on pre-processed samples can have more significant differences. The experimental results show that compared with the signal detection method based on spectrum, our method can obtain a gain of 0.5–2 dB on the simulation data set, and can obtain a gain of 0.5–1 dB on the semi-physical simulation data set.

The rest is in the following order: In Section 2, we elucidate the processing approach of input samples by the convolution neural network and some relevant contents of convolution features. In Section 3, we introduce the nonlinear kernel function and the RBF kernel function used in our paper. In Section 4, we provide the spectrum SPD matrix learning method on the basis of a deep network for signal detection. In Section 5, we will first evaluate the performance of the method, by using the simulation data set based on K-distribution clutter as disturbance, and explore the influence of its hyperparameters, and then compare the performance by using the semi-physical simulation data set on the basis of the measured sea clutter as disturbance. In the end, we provide a conclusion in Section 6.

2. Acquisition of Convolution Features

The convolution features are derived from the convolutional neural network. It originates from the neocognitron proposed in 1980 [28], and the first model LeNet appeared in 1998 [29]. Since 2012, the research of convolutional neural networks has gained significant achievement. The main advantage of convolutional neural networks compared to their predecessors is that it can automatically detect the important features of input data without any human supervision. For example, in image classification, the convolutional neural network can automatically learn the differences between different categories

of images. Additionally, in semantic segmentation, through continuous iteration, the convolutional neural network can discover the features of different types of regions in the image, and identify different targets in the image according to the features learned.

Although models of the convolutional neural network vary widely, they all follow some similar principles. Figure 1 shows the general architecture of a convolutional neural network for a classification task.

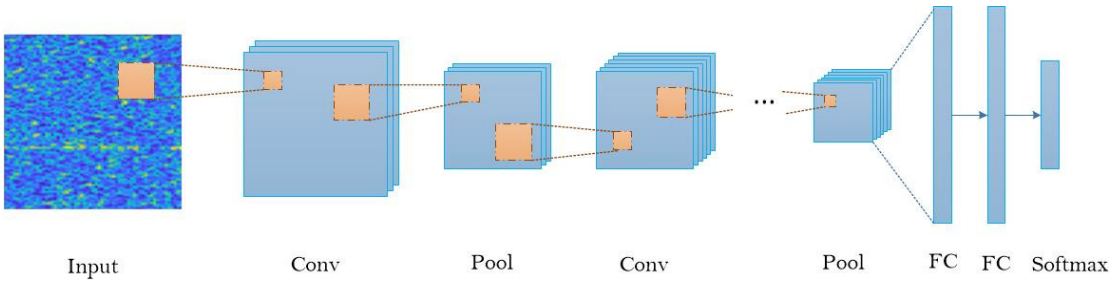


Figure 1. General flow diagram of the convolutional neural network. Although different network models differ, they all basically follow this paradigm.

When an image is fed to the convolutional neural network, it will be processed by a series of convolutional layers and pooling layers. The image is continuously reduced in dimension and finally vectorized into the full connection layers to get the decision result.

Figure 1 omits layers such as nonlinear layers, and normalized layers that perform particular calculations. They are usually immediately behind a convolutional layer and are not distributed throughout the network. Convolution and pooling are the main layers of a convolutional neural network. We mainly introduce the theoretical basis and process of image convolution features extracted by convolutional layers, and the role of pooling in convolution feature extraction.

An image is represented by a matrix of integers, and each pixel corresponds to each element value in the matrix. A monochromatic image is a two-dimensional matrix, while an RGB image is a three-dimensional matrix. Figure 2a is an input example, and can be viewed as a single-channel image. The "0" and "1" represent pixel values. Figure 2b is a 3×3 convolution filter. Each of its element values represents the weight of the position. The feature extraction ability of the convolution filter is closely related to its size and weights.

0	0	1	1	0
1	1	1	0	0
0	1	0	0	1
0	1	0	0	1
0	1	0	0	1

(**a**)

1	0	1
0	2	0
1	0	4

(**b**)

Figure 2. Input image and filter. (**a**) Input image converted to matrix array form; (**b**) The convolution filter/kernel, and in this paper, it is called the convolution filter.

In mathematics, convolution is an operation on two functions that produces a third function expressing how the shape of one is modified by the other. The convolution operation in the convolutional neural network adds each element of the image (or feature map) to its local neighbors, weighted by the convolution filter, to produce a new feature map. The computational relationship between the convolution filter and the image in Figure 2 is shown in Figure 3a. We slide the convolution filter over the input data. As Figure 3b shows, a local multiplication is performed at every location, and we sum the result onto the feature map.

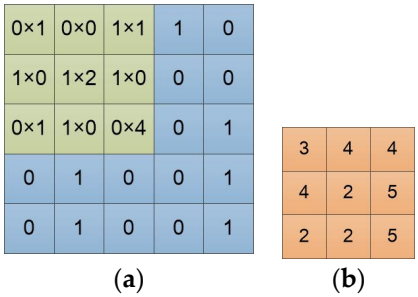


Figure 3. The processing and result of the input image by a convolution filter. (**a**) The computational details of the convolution filter on the input image; (**b**) The computed result of the convolution filter on the input image, and this result is directly fed to the next convolutional layer, or is vectorized firstly and then fed to the following fully connected layer.

The weighted sum between the convolution filter and the image enhances the effects of some local features of the image and suppresses the rest. The convolution features are enhancements of some features of the original image. Different convolution filters can extract different features, and these filters are learnable. Typically, the output feature maps are enhanced with specific features. The feature map will be used as the input of the next convolutional filter, to generate a more abstract feature map with a lower dimension.

An image is represented as a three-dimensional matrix, with dimensions of height, width, and depth (color channels). So, the convolution filter has a depth dimension, and it covers the entire depth of its input. It is also important to note that multiple convolutions on input are performed, each using a different filter and resulting in a distinct feature map. The final output of the convolution layer is the superposition of all feature maps. Figure 4 shows the convolution computation of an input image by a convolution filter. There are twenty convolution filters to deal with the input, respectively, and the generated feature map has a depth of 20. Figure 5 is some feature maps output by different convolution layers of VGG19. From the Conv1_1 layer to the Conv5_1 layer, the depth of the network is increasing, the extracted convolution feature is more and more abstract, the number of feature maps generated by the same layer is increasing, and the dimension is getting lower and lower.

The pooling layer also plays a role in the generation of convolution features. This section provides a brief introduction to the most commonly used pooling: max pooling.

As shown in Figure 6, the calculation process of a max pooling is similar to convolution. However, the max pooling calculation reserves the maximum value in the calculation region. The output of max pooling is the image that retains the most obvious convolution feature. Max pooling reduces the dimension of feature maps, but does not increase or decrease the number of feature maps.

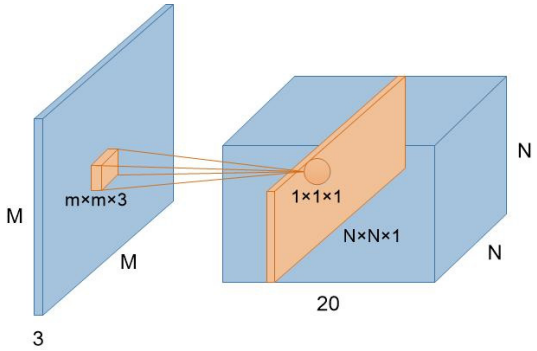


Figure 4. The detailed flowchart of the convolution layer to the input image.

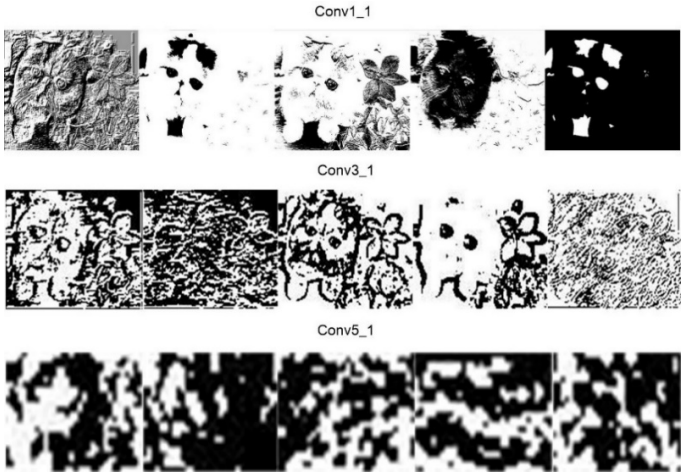


Figure 5. The convolution of VGG19 with different depth extracts convolution features of the input image. As layers get deeper, the convolution features of feature maps become more and more abstract. Different convolution filters generate different feature maps, which can be seen in the figure.

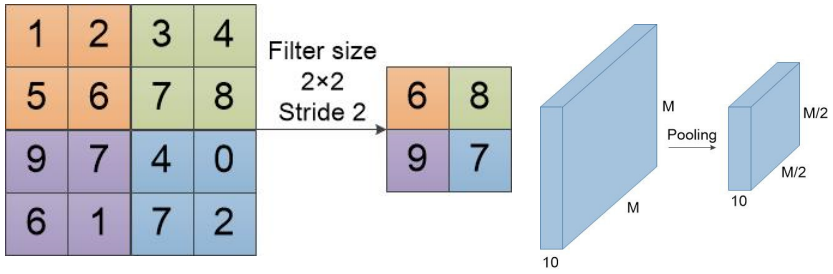


Figure 6. Schematic diagram of the calculation process of a max pooling layer. (**a**) The calculation of a feature map by max pooling; (**b**) The detailed dimension reduction of pooling on the input.

3. Construction of SPD Matrix Based on Convolution Feature

Although the covariance matrix is used to model the features and has acquired gratifying results, two issues still remain ineluctable. One is that the covariance matrix may be singular. This situation usually occurs when the dimension of local features is larger than the number of local features extracted from the image area. The other one is that the covariance matrix cannot evaluate the nonlinear correlation of features. It is not conducive to the modeling of convolution features. To settle the above two issues, we exploit the nonlinear kernel function to construct the SPD matrix.

We set $X \in \mathbb{R}^{C \times H \times W}$ as the set of all convolution features of one layer in the convolutional neural network. C is the number of feature maps, H and W are the height and width of the feature map, respectively. $x_i \in \mathbb{R}^C$ is the i-th local feature and $f_i \in \mathbb{R}^{H \times W}$ is the i-th feature map. The modeling of nonlinear kernel function revolves around f_i.

Elements in the SPD matrix $K \in \mathbb{R}^{C \times C}$ generated by nonlinear kernel function can be defined as:

$$K_{ij} = K\left(f_i, f_j\right) = \left\langle \varphi(f_i), \varphi\left(f_j\right) \right\rangle, \tag{1}$$

where $\varphi(\cdot)$ is the implicit mapping, $\langle \cdot, \cdot \rangle$ is the pairwise inner product.

In this paper, we exploit the RBF kernel function to construct SPD matrices. It is expressed as:

$$K\left(f_i, f_j\right) = exp\left(-\|f_i - f_j\|^2 / 2\sigma^2\right), \tag{2}$$

where σ is the mean Euclidean distances of all feature maps, $\| \cdot \|$ is the Euclidean norm.

The RBF kernel function can guarantee the positive definiteness of the matrix. The proof is as follows:

To simplify the proof, we rewrite Formula (2) as:

$$K\left(f_i, f_j\right) = exp\left(-\alpha\|f_i - f_j\|^2\right), \tag{3}$$

where α is used to denote $1/2\sigma^2$ for the purpose of simplification.

The Fourier transform convention of the Formula (3) is:

$$\kappa(\omega) = (2\pi/\alpha)^{n/2} \int_{R^n} e^{i\omega f_i} e^{-i\omega f_j} e^{-\omega^2/(2\alpha)} d\omega, \tag{4}$$

Then, we prove that the quadratic form of the kernel matrix K is always greater than 0. Set $\mathbf{c} = (c_1, \ldots, c_M) \in \mathbb{R}^{C\times 1}$, and it is an arbitrary non-zero vector. The quadratic expression is obtained as:

$$\begin{aligned} Q = \boldsymbol{c}^T K \boldsymbol{c} &= \sum_{j=1}^{C}\sum_{k=1}^{C} c_j c_k exp\left(-\alpha\|f_i - f_j\|^2\right) \\ &= \sum_{j=1}^{C}\sum_{k=1}^{C} c_j c_k (2\pi/\alpha)^{n/2} \int_{R^n} e^{i\omega f_i} e^{-i\omega f_j} e^{-\|\omega\|^2/(2\alpha)} d\omega \\ &= (2\pi/\alpha)^{n/2} \int_{R^n} e^{-\|\omega\|^2/(2\alpha)} \| \sum_{j=1}^{C} c_j e^{-i\omega f_j} \|^2 d\omega, \end{aligned} \tag{5}$$

In Formula (5), $e^{-\|\omega\|^2/(2\alpha)}$ is positive and continuous, and the quadratic form is 0 on the condition that $\sum_{j=1}^{C} c_j e^{-i\omega f_j} = 0$. However, the complex exponentials $e^{-i\omega f_1}, \ldots, e^{-i\omega f_M}$ are linear independence. Thus, the quadratic form is always greater than 0, and matrix K is positive definite.

The above is the whole process of the positive definiteness; proof of the RBF kernel function. The use of the RBF kernel function to construct the SPD matrix is essentially a correlation calculation between feature maps. Figure 7 shows the calculation process of the correlation operation.

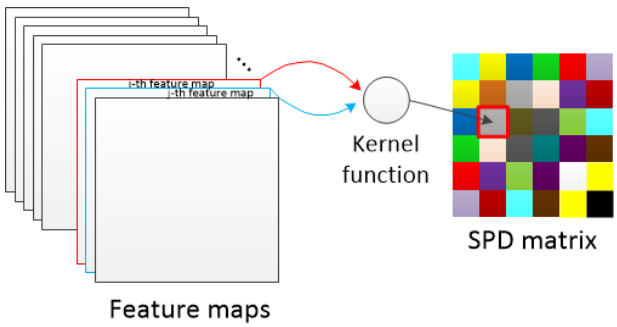


Figure 7. The process of converting a set of feature maps into an SPD matrix. An SPD matrix is constructed from all feature maps in the same layer of a spectrogram sample. Each element on the SPD matrix reflects the correlation between the two feature maps.

Formula (2) reveals the nonlinear relationship between convolution features. It can be calculated faster by means of matrix operations through proper modification.

Convolution features $X \in \mathbb{R}^{C\times H\times W}$ can be reshaped to a matrix $M \in \mathbb{R}^{C\times N}$ and $N = H \times W$. Each feature map $f_i \in \mathbb{R}^{H\times W}$ can be expanded into vector form in a certain order $\mathbf{f}_i \in \mathbb{R}^{1\times N}$. Moreover, the $\|f_i - f_j\|^2$ is equivalent to:

$$\|f_i - f_j\|^2 = \mathbf{f}_i \mathbf{f}_i^T - 2\mathbf{f}_i \mathbf{f}_j^T + \mathbf{f}_j \mathbf{f}_j^T, \tag{6}$$

Three terms to the right of Formula y can be computed by matrix multiplication:

$$\begin{aligned} \mathbf{f}_i\mathbf{f}_i^T &= \mathbf{1}(MM)^T \\ \mathbf{f}_j\mathbf{f}_j^T &= (MM)\mathbf{1}^T \\ \mathbf{f}_i\mathbf{f}_j^T &= MM^T, \end{aligned} \tag{7}$$

where stands for the Hadamard product and $\mathbf{1} \in \mathbb{R}^{C\times N}$ is the matrix whose elements are all "1".

We name $\mathbf{f}_i\mathbf{f}_i^T$ to X1, name $\mathbf{f}_j\mathbf{f}_j^T$ to X2 and name $\mathbf{f}_i\mathbf{f}_j^T$ to X3. formula (2) is modified to the following:

$$K = exp\Big(-(X1 + X2 - 2X3)/2\sigma^2\Big), \tag{8}$$

where $exp(A)$ is the exponential operation to each element in the matrix A.

In this paper, we use VGG19 and GoogLeNet-V3 to extract convolution features. For VGG19, we use feature maps for its pool1, pool2, and pool3 output. These three layers are located in layer 5, layer 10, and layer 19 of VGG19, and their depths are 64, 128, and 256, respectively. This also means that SPD matrices they build are 64×64, 128×128, 256×256, respectively. For GoogLeNet-V3, we use feature maps output by its Maxpool_5a_3×3 layer, which is located in layer 6 of GoogLeNet-V3 and has a depth of 192.

4. SPDnet: A Deep Matrix Learning Method

SPDnet is characterized by embedding bilinear mapping (BiMap) and eigenvalue rectification (ReEig) into the algorithm, in the form of layers of the deep learning network. Their effect on SPD matrices is equivalent to the convolution layer and nonlinear computing layer of the convolution neural network, both of which aim to make objects to be processed more compact and discriminative.

The computation expression for a BiMap layer to perform bilinear mapping is

$$X_k = W_k X_{k-1} W_k^T, \tag{9}$$

where $X_{k-1} \in Sym^+_{d_{k-1}}$ is the input matrix of the k-th layer, $W_k \in \mathbb{R}^+_*, (d_k < d_{k-1})$ is a transformation matrix, which is row full-rank, and $X_k \in Sym^+_{d_k}$ is the output matrix.

The computation expression for a ReEig layer to perform eigenvalue rectification is

$$X_k = U_{k-1}\max(\varepsilon I, \Sigma_{k-1})U_{k-1}^T, \tag{10}$$

where U_{k-1} and Σ_{k-1} are achieved by singular value decomposition, $X_{k-1} = U_{k-1}\Sigma_{k-1}V_{k-1}^T$, ε indicates a threshold, I indicates an identity matrix, and $\max(\varepsilon I, \Sigma_{k-1})$ means retaining the elements in Σ_{k-1} that are greater than ε and replacing the rest with ε.

Exploiting singular value decomposition instead of eigenvalue decomposition to analyze the SPD matrix is an efficient approximation method. The computational speed of singular value decomposition is faster than that of eigenvalue decomposition. Similar to the eigenvalues, singular values are arranged from large to small in the diagonal matrix generated by the decomposition. In most cases, the sum of the first 10% singular values occupies around 99% of the sum of all singular values. Based on the square matrix property of the SPD matrix, the approximate reconstruction by the ReEig layer is reasonable.

The LogEig layer is used to perform Riemannian calculation on SPD matrices generated after multiple bilinear mappings and eigenvalue correction. The log-Euclidean Riemannian metric endows the SPD matrices Riemannian manifold with a Lie group structure, and the SPD matrix manifold is reduced to a flat space. The log-Euclidean metric is described in detail below.

The space of $C \times C$ SPD matrices can form a Riemannian manifold $\boldsymbol{S}_+^C$. It has a globally defined differential structure and has the possibility to define the derivatives of the curves on the SPD matrix manifold. The tangent space $\mathbf{T}_{\mathbf{S}_1}\boldsymbol{S}_+^C$ of point $\mathbf{S}_1$ on the manifold can be constructed by

exploiting logarithm map $log_{\mathbf{S}_1} : S_+^C \to \mathbf{T}_{\mathbf{S}_1} S_+^C \left(\mathbf{S}_1 \in S_+^C\right)$, and an inner product $\langle , \rangle_{S_1}$. The family of inner products on Riemannian manifold's all tangent spaces is called the Riemannian metric of the manifold. The geodesic distance between two points $\mathbf{S}_1$, $\mathbf{S}_2$ on the SPD matrix manifold can be expressed as $\left\langle log_{\mathbf{S}_1}(\mathbf{S}_2), log_{\mathbf{S}_1}(\mathbf{S}_2) \right\rangle_{S_1}$ in the framework of Riemannian metric.

The log-Euclidean metric of the SPD matrix manifold corresponds to the Euclidean metric in the SPD matrix logarithmic domain. Thus, the scalar product between $\mathbf{T}_1$ and $\mathbf{T}_2$ in the tangent space at the point $\mathbf{S}$ is

$$\langle \mathbf{T}_1, \mathbf{T}_2 \rangle_S = \langle D_S log.\mathbf{T}_1, D_S log.\mathbf{T}_2 \rangle, \tag{11}$$

where $D_S log.\mathbf{T}$ is the directional derivative of the matrix logarithm at $\mathbf{S}$ along with $\mathbf{T}$.

The logarithmic and exponential maps related to the metric can be expressed as

$$\begin{array}{c} log_{\mathbf{S}_1}(\mathbf{S}_2) = D_{\log(\mathbf{S}_1)} exp.(\log(\mathbf{S}_2) - \log(\mathbf{S}_1)) \\ exp_{\mathbf{S}_1}(\mathbf{T}_2) = exp\left(\log(\mathbf{S}_1) + D_{\mathbf{S}_1} log.\mathbf{T}_2\right), \end{array} \tag{12}$$

where $D_{\log(\mathbf{S}_1)} exp. = (D_{\mathbf{S}} log.)^{-1}$ roots in the differentiation of the equality $logexp = \mathbf{I}$, and $\mathbf{I}$ is the identity matrix.

From Formulas (11) and (12), the geodesic distance obtained through log-Euclidean metric is

$$D_{le}(\mathbf{S}_1, \mathbf{S}_2) = \left\langle log_{\mathbf{S}_1}(\mathbf{S}_2), log_{\mathbf{S}_1}(\mathbf{S}_2) \right\rangle_{S_1} = \| \log(\mathbf{S}_1) - \log(\mathbf{S}_2) \|_F^2, \tag{13}$$

Formula (13) is to the Euclidean distance in the logarithmic domain. Under the log-Euclidean metric, the distance between any two points on SPD matrix manifold is acquired through propagating by translation the scalar product in the tangent space at the identity matrix. The space of SPD matrices is reduced to a flat Riemannian space with the help of the log-Euclidean.

The disadvantage of the log-Euclidean metric is that it is not an affine invariant metric, which leads to the fact that it cannot fully reflect the geodesic distance between two points on the Riemannian manifold. However, the form of the log-Euclidean metric has low computational complexity, and it is widely used in Riemannian manifold-based algorithms. For more details about the log-Euclidean metric, please refer to the literature [30].

The computation expression for the LogEig layer is

$$X_k = log(X_{k-1}) = U_{k-1} log(\Sigma_{k-1}) U_{k-1}^T, \tag{14}$$

where $X_{k-1} = U_{k-1} \Sigma_{k-1} U_{k-1}^T$ and $log(\sum_{k-1})$ is the diagonal matrix of eigenvalue logarithms. After the treatment of LogEig layer, the linear classification method on the basis of Euclidean space can be used to perform tasks.

We mainly exploit the following SPDnet model with 8 layers: $X_0 \to f_{BM}^1 \to f_{RE}^2 \to f_{BM}^3 \to f_{RE}^4 \to f_{BM}^5 \to f_{LE}^6 \to f_{FC}^7 \to f_{soft}^8$, where $X_0, f_{BM}, f_{RE}, f_{LE}, f_{FC}, f_{soft}$ is the input SPD matrix to the model, BiMap layer, ReEig layer, LogEig layer, fully-connected layer, and softmax log-loss layer, respectively.

5. Results

5.1. Experimental Analysis of Simulation Data

In this part, we made a comparision with two signal detection methods on the basis of the deep neural network: deep network based on time-frequency spectrum for signal detection and deep SPD matrix learning network for signal detection based on spectrum convolution feature. The data originated from a simulated signal data set, and the simulated clutter based on K-distribution is used as interference. K-distribution is a widely accepted sea clutter simulation model. Compared with Rayleigh distribution, lognormal distribution and Weibull distribution, it considers the correlation

between echo pulses, and better fits the amplitude distribution of sea clutter.The probability density of K-distribution follows the following formula:

$$f(x) = \frac{2}{a\Gamma(v)}\left(\frac{x}{2a}\right)^{v} K_{v-1}\left(\frac{x}{a}\right), \tag{15}$$

where $K_{v-1}(\cdot)$ indicates the second modified Bessel function of order $v-1$, $\Gamma(\cdot)$ indicates the Gamma function, a indicates the scale parameter to represent the mean value of clutter, which is set as 1 in this paper, v indicates the shape parameter to affect the shape of the distribution curve, which is set as 1 in this paper.

Target signal is generated by simulation. Its driving vector is

$$p = \frac{1}{\sqrt{N}}[1, \exp(j2\pi f_d), \ldots, \exp(j2\pi(N-1)f_d)]^T, \tag{16}$$

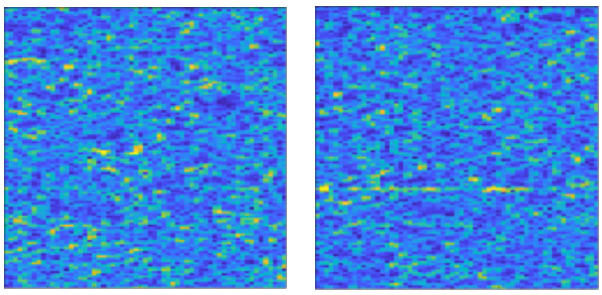
where N indicates the number of pulses of the signal and is set as 2048, f_d indicates the normalized Doppler frequency, set to 0.15.

The definition of normalized Doppler frequency is

$$f_d = \frac{F_d}{f_s} = \frac{vf_c}{cf_s}, \tag{17}$$

where $F_d = \frac{vf_c}{c}$ indicates the Doppler frequency of the simulated target, v indicates the relative velocity of the target, set to 5 m/s, c indicates the speed of the electromagnetic wave and is 3×10^8 m/s, f_c indicates the carrier frequency of the radar, set to 9.39 GHz, f_s indicates the pulse repetition rate of the radar, set to 1000 Hz.

There are 11,000 samples with merely clutter and no target signal. The SCR of clutter interfered samples with the target signal is distributed from −5 dB to −20 dB at 1 dB intervals, with 4000 samples for each SCR. The time-frequency spectrum in the form of color map involved in this section is displayed in Figure 8.



(**a**) (**b**)

Figure 8. Time-frequency spectrum color maps used in this study. K-distribution clutter is exploited. (**a**) The sample containing only clutter; (**b**) The sample containing signal interfered by clutter. Its SCR is −15 dB.

5.1.1. Comparison with Convolutional Neural Networks

For the purpose of signal detection, SPDnet is exploited to process SPD matrices generated by convolution features, and GoogLeNet and VGG19 are exploited to process time-frequency spectra. Featuremaps of time-frequency spectra are generated and ourput by Tensorflow 1.8, and then they are processed by RBF kernel function to generate SPD matrices. We use transfer learning to adapt GoogLeNet and VGG19 to our signal detection problem, and retrain the SPDnet. The samples used for training all models are samples that only contain clutter and clutter interfered signal samples with SCR of −5 dB, −10 dB and −15 dB. Every training session, we use 1000 pure clutter samples and 1000 clutter

disturbed signal samples, and the SCR of the latter is the same. The remaining samples are used to evaluate the detection performance of each model. Model training will have some preset parameters, which will not be updated with iteration, and we name them hyperparameters. The hyperparameters involved in this paper and their set values are as follows: the batch size, set as 20; the learning rate, set as 0.001; the weight decay, set as 0.0005; the epoch for SPDnet, set as 500; the epoch for GoogLeNet and VGG19, set as 800. All models have been fully trained. We trained SPDnet on an i7-8565U CPU, while GoogLeNet and VGG19 were trained on an Nvidia GeForce GTX 1080 Ti GPU. The detection probability of the above models is shown in Figure 9. The SCR of signal samples used for training is −5 dB.

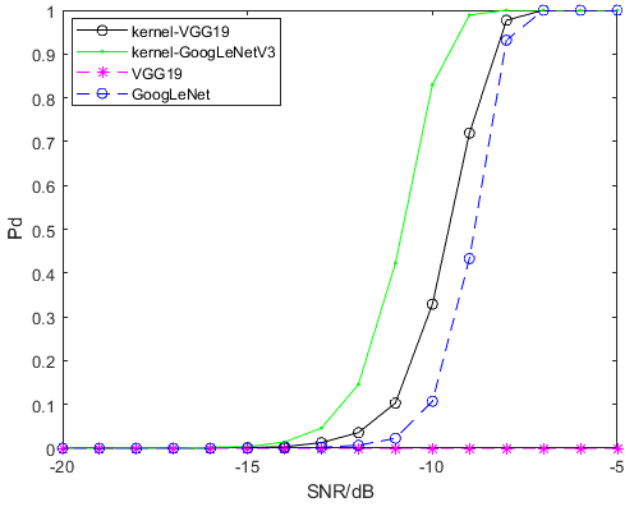


Figure 9. Detection probability of models involved in this section. To ensure that the false alarm probability of all models does not exceed 10^{-4}, the SCR of the signal samples interfered by clutter used in training is −5 dB. "kernel-VGG19" on behalf of the SPD matrix network trained by SPD matrices based on pool2 convolution features of VGG19, and "kernel-GoogLeNetV3" on behalf of the SPD matrix network trained by SPD matrices based on convolution features of GoogLeNetV3.

Figure 9 shows that two SPD matrix learning networks based on convolution features have advantages in detection problem. On the whole, the detection performance of two SPD matrix networks based on convolution features is better than that of GoogLeNet, and VGG19 cannot detect the target signal. The gain of the SPD matrix network trained by SPD matrices based on pool2 convolution features of VGG19 relative to GoogLeNet is about 0.6 dB when the detection probability of both models reaches 70%, and the gain of the SPD matrix network trained by SPD matrices based on the convolution features of GoogLeNetV3 relative to GoogLeNet is 2 dB when the detection probability of both models reaches 80%.

Table 1 is the comparision of the false alarm probability about different models involved in this paper. The false alarm probability of them is not greater than 10^{-4} under the circumstance of the SCR is −5 dB. With the decrease of SCR, the false alarm probability of them increases, but that of two SPD matrix network models increases greatly.

Table 1. False alarm probability of different models (expressed as %).

Model / SCR (dB)	VGG19-Kernel SPD Matrix Network	GoogLeNetV3-Kernel SPD Matrix Network	GoogLeNet with Time-Frequency Spectra	VGG19 with Time-Frequency Spectra
−5	<0.01	<0.01	<0.01	<0.01
−10	0.97	0.1	<0.01	<0.01
−15	24.45	7.32	1.13	<0.01

5.1.2. The Effect of Hyperparameters

The learning effect of deep neural networks is influenced by hyperparameters. We explore the impact of the learning rate, the output layer of convolution features and the dimension of the SPD matrix before entering the LogEig layer, on the performance of the spectral convolution feature-based SPD matrix learning method, with deep learning for signal detection. In order to simplify the work, we merely use the SPD matrix network based on pool2 convolution features of VGG19. The signal samples interfered by clutter used by the network models involved in Figures 10–12 are all −5 dB.

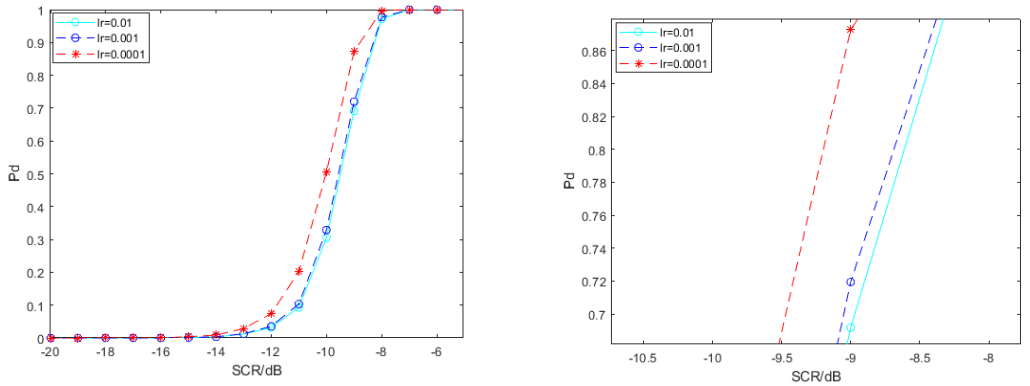


Figure 10. Detection probability under different learning rates. (**b**) is the local amplification of (**a**).

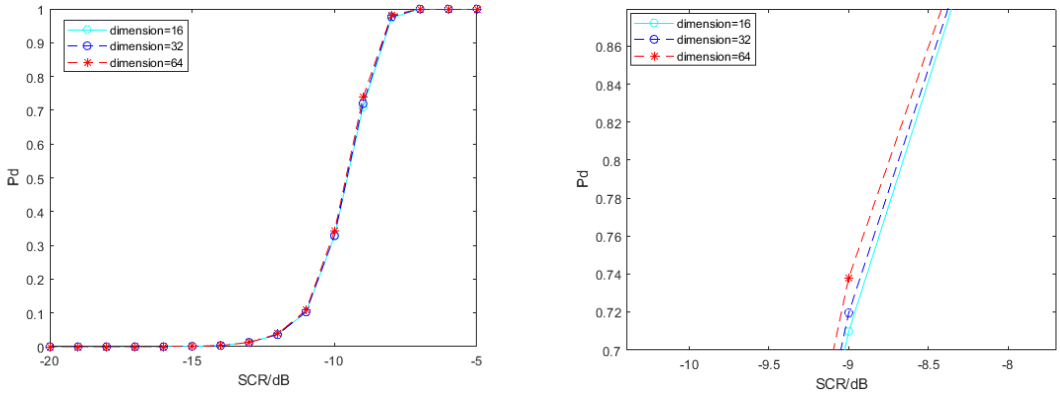


Figure 11. Detection probability under different dimensions. (**b**) is the local amplification of (**a**).

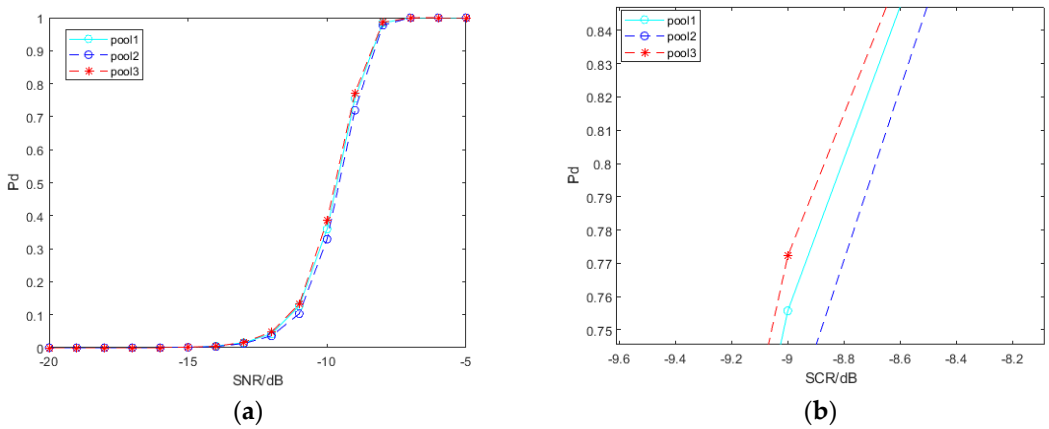


Figure 12. Detection probability curves under SPD matrices constructed by different convolution layers. (**a**) shows the general trends; (**b**) shows the differences when the detection probability is around 80%.

The detection probability of the SPD matrix learning network trained by SPD matrices based on pool2 convolution features of VGG19 under different learning rates is shown in Figure 10. The results show that the low learning rate is beneficial to the improvement of detection probability. The SPD matrix learning method for signal detection which the learning rate is set as 0.0001 gets a gain of about 0.5 dB compared with others under the circumstance of the detection probability; 80%. In the backpropagation algorithm of a neural network, the learning rate is the step length of each iteration to find the local optimum. A high learning rate may skip the optimum. If the training time is sufficient, a small learning rate helps find the local optimum.

In this paper, "dimension" refers to the size of the SPD matrix before entering the LogEig layer. The SPD matrix will be more compact and discriminative through bilinear mapping and eigenvalue rectification. However, the reduction of matrix size is bound to cause some features to be lost theoretically. The results shown in Figure 11 confirm this conjecture. The detection probability is slightly optimized as the dimension increases. It is important to note that the overall effect is small.

From Figure 5, the deeper the network layer is, the more abstract the convolution feature is extracted. In theory, there are differences in the convolution features extracted from different layers to construct the SPD matrix. We select the convolution features of VGG19 output at three layers to construct SPD matrices: pool1, pool2, and pool3. The influence of convolution features generated by different layers is shown in Figure 12. As illustrated, the detection performance of SPD matrices based on convolution features of pool1 and pool3 is better than that based on convolution features of pool2. This may be related to the convolution features and characteristics of SPD matrices.

On the one hand, the deeper the layer is, the more abstract the convolution feature is extracted, which may lead to a loss of information for signal detection. The convolution features extracted at the shallow level retain more information, but in this case, there are fewer feature maps, and the generated SPD matrix dimension is smaller. On the other hand, although the convolution features extracted at a deeper level are more abstract, there are more feature maps and more comprehensive descriptions of samples, which can build the SPD matrix with a larger dimension.

The false alarm probability of above models when changing learning rate, dimension and the output layer of convolution features is shown in Tables 2–4. In general, the false alarm probability increases rapidly with the decrease of SCR. Only when SCR is −5 dB, false alarm probability is within the allowed range.

Table 2. False alarm probability of different learning rates (expressed as %).

Model / SCR (dB)	Learning Rate 0.01	Learning Rate 0.001	Learning Rate 0.0001
−5	<0.01	<0.01	<0.01
−10	0.67	0.97	1.27
−15	5.35	24.45	25.57

Table 3. False alarm probability of different dimension (expressed as %).

Model / SCR (dB)	16 Dimension	32 Dimension	64 Dimension
−5	<0.01	<0.01	<0.01
−10	0.92	0.97	0.95
−15	22.58	24.45	20.95

Table 4. False alarm probability of SPD matrices generated by convolution features from different layers (expressed as %).

SCR (dB) \ Model	Pool1	Pool2	Pool3
−5	<0.01	<0.01	<0.01
−10	0.92	0.97	0.97
−15	23.30	24.45	17.03

5.2. Experimental Analysis of Semi-Physical Simulation Data

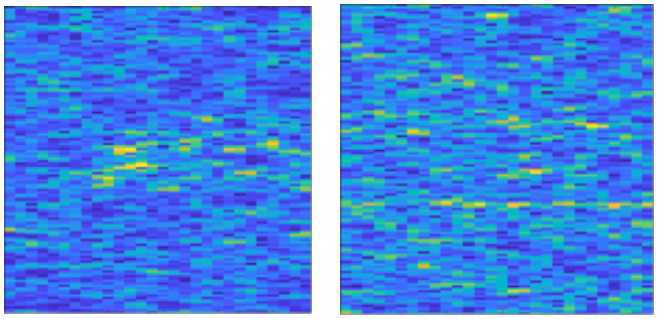
In this section, we select and use part of the IPIX (Ice Multiparameter Imaging X-Band) lake radar echo data as the clutter exerting interference [31]. Tables 5 and 6 are the data files used and their basic parameters. Due to the data having no clear target information, we added one target signal. The number of pulses per sample is 1000, and other parameters are consistent with Section 5.1. The time-frequency spectrum in the form of color map involved in this section is shown in Figure 13.

Table 5. The IPIX (Ice Multiparameter Imaging X-Band) radar data file used in this section.

Number	Name
1	19980223_171533_ANTSTEP
2	19980223_171811_ANTSTEP
3	19980223_172059_ANTSTEP
4	19980223_172410_ANTSTEP
5	19980223_172650_ANTSTEP
6	19980223_184853_ANTSTEP
7	19980223_185157_ANTSTEP

Table 6. Parameters of IPIX sea clutter signal.

Pulse Repetition Frequency	Carrier Frequency	The Length of the Pulse	Range Resolution	Polarization Mode
1000 Hz	9.39 GHz	60,000	3 m	HH



(**a**) (**b**)

Figure 13. Time-frequency spectrum color maps used in this study. IPIX sea clutter is exploited. (**a**) The sample containing only clutter; (**b**) The sample containing signal interfered by clutter. Its SCR is −15 dB.

The false alarm probability of different models is shown in Table 7. With the decrease of SCR, the false alarm probability of all models increases rapidly, which may be related to the uneven distribution of real sea clutter. The false alarm probability of the two models of convolutional neural increases more slowly than that of the two models based on SPD matrix learning.

Table 7. The false alarm probability of models involved in this section (expressed as %).

Model / SCR (dB)	VGG19-Kernel SPD Matrix Network	GoogLeNetV3-Kernel SPD Matrix Network	GoogLeNet with Time-Frequency Spectra	VGG19 with Time-Frequency Spectra
−5	0.64	<0.01	<0.01	<0.01
−10	7.56	1.39	<0.01	<0.01
−15	28.17	1.64	0.15	<0.01

The detection property of the SPD matrix learning networks for signal detection and the convolutional neural networks is shown in Figure 14. Compared with Section 5.1, the detection property have changed significantly. The detection property of two SPD matrix learning networks on the basis of convolution features are better than that of GoogLeNet and VGG19. The advantage of two SPD matrix networks based on convolution features over GoogLeNet is obvious when SCR is low.

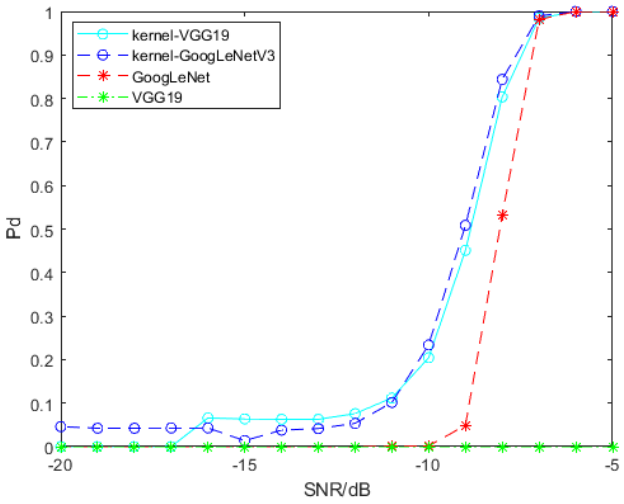


Figure 14. Detection probability of models in this section. The probability of false alarm is not greater than 10^{-4}. "kernel-VGG19" is the SPD matrix learning network based on pool2 convolution features of VGG19, and "kernel-GoogLeNetV3" is the SPD matrix learning network based on the convolution features of GoogLeNetV3.

5.3. Complexity Analysis

In this section, we analyze the complexity of the involved models from two aspects: training time and size of space occupied by models. The comparison of training time and size of space occupied is shown in Tables 8 and 9, respectively. Note that GPU is not used for training SPD matrix networks, as mentioned in Section 5.1.1. By comparison, it can be seen that our model is of low complexity.

Table 8. Training time comparision of different models per 100 epochs (in Section 5.1.1).

Model / SCR (dB)	VGG19-Kernel SPD Matrix Network	GoogLeNetV3-Kernel SPD Matrix Network	GoogLeNet with Time-Frequency Spectra	VGG19 with Time-Frequency Spectra
−5	1618 s	1551 s	1599 s	3470 s
−10	1634 s	1567 s	1598 s	3483 s
−15	1629 s	1564 s	1624 s	3521 s

Table 9. Size of space occupied comparision of different models (in Section 5.1.1).

SCR (dB) \ Model	VGG19-Kernel SPD Matrix Network	GoogLeNetV3-Kernel SPD Matrix Network	GoogLeNet with Time-Frequency Spectra	VGG19 with Time-Frequency Spectra
−5	163 KB	187 KB	22,217 KB	507,474 KB
−10	163 KB	187 KB	22,216 KB	507,473 KB
−15	162 KB	187 KB	22,216 KB	507,474 KB

6. Conclusions

We present a deep SPD matrix method for signal detection on the basis of spectral convolution features. Utilizing the SPD matrix obtained by transforming convolution features through the nonlinear kernel function and integrating with the deep SPD matrix learning network, a target signal under low SCR can be detected by the present method. The superiorities of this approach are that it combines the convolution features with the SPD matrix, converts original samples to the SPD matrix space, exploits its Riemannian manifold property to enhance the discriminability between different samples. Synchronously, the use of deep learning ameliorates the capacity of signal detection. A simulation date set with K-distribution clutter as interference and a semi-physical simulation data set with real sea clutter as interference are used by us to evaluate the performance of our method. The final results show that this method is effective, and it can obtain a gain of 0.5–2 dB on the simulation data set, and can obtain a gain of 0.5–1 dB on the semi-physical simulation data set. We explored the impacts of the hyperparameters associated with our method. It is inevitable that, due to the non-uniformity of interference in the real environment, our method may have poor detection ability. Our next goal is to enhance the anti-interference ability of our method and reduce the possibility of false alarms.

Author Contributions: J.W. and M.L. put forward the original ideas and performed the research. X.H. conceived and designed the simulations. X.Z. reviewed the paper and provided useful comments. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China under grant No. 61901479.

Acknowledgments: The authors are grateful for the valuable comments made by the reviewers, which have assisted us with a better understanding of the underlying issues and therefore a significant improvement in the quality of the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Harrahs and Harveys, Lake Tahoe, CA, USA, 3–8 December 2012; pp. 1097–1105.
2. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
3. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1–9.
4. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
5. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition(CVPR), Honolulu, HI, USA, 22–25 July 2017; pp. 4700–4708.
6. Yandex, A.B.; Lempitsky, V. Aggregating local deep features for image retrieval. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 11–18 December 2015; pp. 1269–1277.

7. Tolias, G.; Sicre, R.; Jégou, H. Particular object retrieval with integral max-pooling of cnn activations. *arXiv* **2015**, arXiv:1511.05879.
8. Kalantidis, Y.; Mellina, C.; Osindero, S. Cross-dimensional weighting for aggregated deep convolutional features. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 8–16 October 2016; pp. 685–701.
9. Gao, Y.; Beijbom, O.; Zhang, N.; Darrell, T. Compact bilinear pooling. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 317–326.
10. Pennec, X.; Fillard, P.; Ayache, N. A Riemannian Framework for Tensor Computing. *Int. J. Comput. Vis.* **2006**, *66*, 41–66. [CrossRef]
11. Wong, K.; Zhang, J.; Jiang, H. Multi-sensor signal processing on a PSD matrix manifold. In Proceedings of the 2016 IEEE Sensor Array and Multichannel Signal Processing Workshop, Rio de Janeiro, Brazil, 10–13 July 2016; pp. 1–5.
12. Hua, X.; Cheng, Y.; Wang, H.; Qin, Y. Robust Covariance Estimators Based on Information Divergences and Riemannian Manifold. *Entropy* **2018**, *20*, 219. [CrossRef]
13. Huang, Z.; Wang, R.; Li, X.; Liu, W.; Shan, S.; Van Gool, L.; Chen, X. Geometry-Aware Similarity Learning on SPD Manifolds for Visual Recognition. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *28*, 2513–2523. [CrossRef]
14. Harandi, M.; Salzmann, M.; Hartley, R. Dimensionality Reduction on SPD Manifolds: The Emergence of Geometry-Aware Methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 48–62. [CrossRef] [PubMed]
15. Zhang, J.; Wang, L.; Zhou, L.; Li, W. Exploiting structure sparsity for covariance-based visual representation. *arXiv* **2016**, arXiv:1610.08619.
16. Zhou, L.; Wang, L.; Zhang, J.; Shi, Y.; Gao, Y. Revisiting metric learning for SPD matrix based visual representation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 3241–3249.
17. Ionescu, C.; Vantzos, O.; Sminchisescu, C. Matrix backpropagation for deep networks with structured layers. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 13–16 December 2015; pp. 2965–2973.
18. Huang, Z.; Wang, R.; Shan, S.; Li, X.; Chen, X. Log-euclidean metric learning on symmetric positive definite manifold with application to image set classification. In Proceedings of the International Conference on Machine Learning (ICML), Lille, France, 7–9 July 2015; pp. 720–729.
19. Herath, S.; Harandi, M.T.; Porikli, F. Learning an invariant Hilbert space for domain adaptation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 3845–3854.
20. Li, P.; Xie, J.; Wang, Q.; Zuo, W. Is second-order information helpful for large-scale visual recognition. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; pp. 2070–2078.
21. Yu, K.; Salzmann, M. Second-order convolutional neural networks. *arXiv* **2017**, arXiv:1703.06817.
22. Wang, R.; Guo, H.; Davis, L.S.; Dai, Q. Covariance discriminative learning: A natural and efficient approach to image set classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 18–20 June 2012; pp. 2496–2503.
23. Jayasumana, S.; Hartley, R.; Salzmann, M.; Li, H.; Harandi, M. Kernel methods on Riemannian manifolds with Gaussian RBF kernels. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 2464–2477. [CrossRef] [PubMed]
24. Dong, Z.; Jia, S.; Zhang, C.; Pei, M.; Wu, Y. Deep manifold learning of symmetric positive definite matrices with application to face recognition. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017.
25. Huang, Z.; Van Gool, L. A riemannian network for spd matrix learning. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 4–9 February 2017.
26. Tong, Z.; Zheng, W.; Cui, Z.; Li, C. Deep manifold-to-manifold transforming network. In Proceedings of the 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 7–10 October 2018; pp. 4098–4102.
27. Gao, Z.; Wu, Y.; Bu, X.; Yu, T.; Yuan, J.; Jia, Y. Learning a robust representation via a deep network on symmetric positive definite manifolds. *Pattern Recognit.* **2019**, *92*, 1–12. [CrossRef]

28. The Entry on Neocognitron. Available online: http://www.scholarpedia.org/article/Neocognitron (accessed on 20 July 2020).
29. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
30. Arsigny, V.; Fillard, P.; Pennec, X.; Ayache, N. Geometric means in a novel vector space structure on symmetric positive-definite matrices. *SIAM J. Matrix Anal. Appl.* **2007**, *29*, 328–347. [CrossRef]
31. The McMaster IPIX Radar Sea Clutter Database. Available online: http://soma.ece.mcmaster.ca/ipix/ (accessed on 20 July 2020).

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).