Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/MBE

Mathematical Biosciences and Engineering, 16(3): 1300–1312.
DOI: 10.3934/mbe.2019063
Received: 18 November 2018
Accepted: 20 January 2019
Published: 20 February 2019

Research article

A sparse deep learning model for privacy attack on remote sensing images

Eric Ke Wang[1], Nie Zhe[2], Yueping Li [2,*], Zuodong Liang[1], Xun Zhang[1], Juntao Yu[1] and Yunming Ye[1]

[1] Harbin Institute of Technology, Shenzhen, 518055, China

[2] School of Computer Engineering, Shenzhen Polytechnic, Shenzhen, 518055, China

* **Correspondence:** Email: yuepinglee@gmail.com; Tel: +8613632783348.

Abstract: Deep learning tools have been a new way for privacy attacks on remote sensing images. However, since labeled data of privacy objects in remote sensing images are less, the samples for training are commonly small. Besides, traditional deep neural networks have a huge amount of parameters which leads to over complexity of models and have a great heavy of computation. They are not suitable for small sample image classification task. A sparse method for deep neural network is proposed to reduce the complexity of deep learning model with small samples. A singular value decomposition algorithm is employed to reduce the dimensions of the output feature map of the upper convolution layers, which can alleviate the input burden of the current convolution layer, and decrease a large number of parameters of the deep neural networks, and then restrain the number of redundant or similar feature maps generated by the over-complete schemes in deep learning. Experiments with two remote sensing image data sets UCMLU and WHURS show that the image classification accuracy with our sparse model is better than the plain model,which is improving the accuracy by 3%,besides, its convergence speed is faster.

Keywords: singular value decomposition; sparse model; image classification; convolutional neural network

1. Introduction

Since many remote sensing images have a plenty of sensitive information, privacy attackers have been considering them as new targets, as more and more high-resolution remote sensing images are produced. For example, it is easy to locate the military base, aircraft carriers in the navy harbor from Google earth, and even some cases show that city residents' roofs can be detected in the remote sensing images. How to find more privacy information in a remote sensing image has become a new interesting topic. Traditional security methods are not applicable for the area[1, 2]. Since the remote

sensing images are commonly large scale in size, some machine learning tools are employed to assist to identify the sensitive objects. In recent years, deep neural network have been a new hot way to execute the privacy attacks for remote sensing images.

However, as all know, producing a huge number of parameters is a typical feature of deep learning models [3],sometimes parameters increasing is to be exponential level, for example, the parameters of a typical Convolutional Neural Network(CNN) are millions or ten millions, even to be a hundred millions. The model with a mass of parameters demands great capacity of computation and storage. Besides, since labeled data of privacy objects in remote sensing images are less, the samples for training are commonly small. Therefore, it is too complex to achieve a good deep leaning performance with small sample remote sensing images.

Actually, small sample data are more common to appear in real world. With deep learning methods becoming popular, how to apply deep learning models on small sample data has been concerned in recent years. Besides of fine-tuning technique [4, 5], sparse model to cut redundant parameters is one of the important ways. For example, LeCun, Y. proposed an Optimal Brain Damage article [6] to remove the unimportant parameters from the network, so as creating a sparse model; Han, S.[7] apply the reduction, weight sharing, quantization, coding and other methods to model compression, and achieved a good result. In order to fit various practical situations, researchers are required to design more detailed and efficient network models based on modular designed CNN, such as SqueezeNet [8], MobileNet [9] and so on, these models greatly reduce many parameters, and achieve good performance.

Besides, many learning frameworks [10, 11] simplify convolutions into matrices by matrix multiplication in order to reduce computation and storage. They use tensor compression technique to compress the full connection layers, and use tensor train format [12] to represent the parameters of the network layers. Denton, E. and Jaderberg, M. [13, 14] use low rank decomposition to sparse the neural network and achieves approximately twice the acceleration ratio, while Liu, B. et al. [15] greatly improves the efficiency by compressing the convolution layers.

In order to simplify the deep model to fit small sample data, a sparse method is proposed to achieve model reduction. By compressing output feature map of each layer, it achieves the dimension reduction and compression. The method is proposed to solve the classification of remote sensing images which are typical small-sample images data. The experiments with remote sensing images data have been enforced by the sparse model and show that our method is able to improve the model accuracy. Commonly, high resolution optical remote sensing image classification is difficult due to the large scale images but limited training samples. Traditional classification techniques can not effectively adapt to the complexity of high resolution optical remote sensing images and the diversity of targets. With the remote sensing image resolution becomes higher, the image quality is close to the conventional HD images. In recent years, deep learning technology for conventional images has been developed rapidly, but in the field of large-scale but small-sample remote sensing image classification, it is relatively slow. One of most important reasons is that labeling large-scale images are costly. Therefore, it is practical and of significance to explore a more effective image classification algorithm for small samples of remote sensing images.

2. Methods

CNN is often employed to solve computer vision tasks [16]. In order to extract enough features, CNNs adopt an over-complete scheme [17], so there may be one or more similar feature maps in each layer of convolution layers. That would greatly increase the burden of the next layer, resulting in excessive network parameters. Therefore, reducing the input dimension of each layer, so as to reduce the number of model parameters, is expected.

Based on the idea of singular value decomposition, LeCun, Y. et al. improve a CNN by singular value decomposition for weighting matrix of the model after training the network, and enhance the running speed of the model. The scheme shows that it enforces singular value decomposition after training so that it can obtain a similar network structure, while increase extra workload after training.

Unlike the scheme proposed by LeCun, Y. and others, the network structure of our singular value decomposition compresses the feature map of current layer before the layer output feature map into the next layer of convolution layer. Because CNN adopts an over-complete method, there are one or more similar feature maps in each convolution layer. Our method compresses the output feature maps of each layer to simulate the effect of SVD dimension reduction and network compression. Although this is not strictly singular value decomposition, it also realizes dimension reduction and compression. Moreover, it decreases the additional workload required by the strict singular value decomposition since it can directly learn the weight matrix of the network when it is trained, and as so to reduce the number of model parameters and sparse the CNN.

2.1. Singular value decomposition

Singular value decomposition(SVD) [18] is a convenient matrix decomposition method, which can mine potential patterns of data. Therefore, it is widely used in various fields, such as principal component analysis and recommendation system. The full rank decomposition of arbitrary matrices can be obtained by SVD.

$$A_{m\times n} = X_{m\times k} Y_{k\times n} \tag{2.1}$$

Where $k = Rank(A)$, that is, k is the rank of matrix A. Therefore, in the case of particularly high data correlation, the storage matrix X and the matrix Y occupy less space than the storage of the original matrix A so that it can be applied to data dimension reduction.

SVD is applicable to any matrix, and the steps to solve singular values and singular vectors are as follows:

(1) solving the feature vectors and feature values of the product of the transpose matrix of matrix A multiplication matrix A.

$$(A^T A)v = \lambda v \tag{2.2}$$

The feature vector obtained by the above formula is the right singular vector of matrix A.

(2) solving singular values of matrix A.

$$\sigma_i = \sqrt{\lambda_i} \tag{2.3}$$

(3)solving left singular vector of matrix A

$$\mu_i = \frac{1}{\sigma_i} A v_i \tag{2.4}$$

Arrange all singular values from large to small, then form a diagonal matrix Σ, and the corresponding singular vectors form a left singular matrix U and a right singular matrix V, then the original matrix A can be written as

$$A_{m\times n} = U_{m\times m}\Sigma_{m\times n}V_{n\times n} \tag{2.5}$$

In most cases, the sum of the singular values of the first 10% or even 1% accounts for more than 99% of the sum of all singular values. That is to say, the singular value of the first k rows can be used to approximate the original matrix A.

$$A_{m\times n} = U_{m\times k}\Sigma_{k\times k}V_{k\times n} \tag{2.6}$$

if $X_{m\times k} = U_{m\times k}\Sigma_{k\times k}$ and $Y_{k\times n} = V_{k\times n}$,then equation (2.6) can be transfer to equation (2.1).

2.2. Our method

Our method called RSparse SVD is shown in Figure 1.

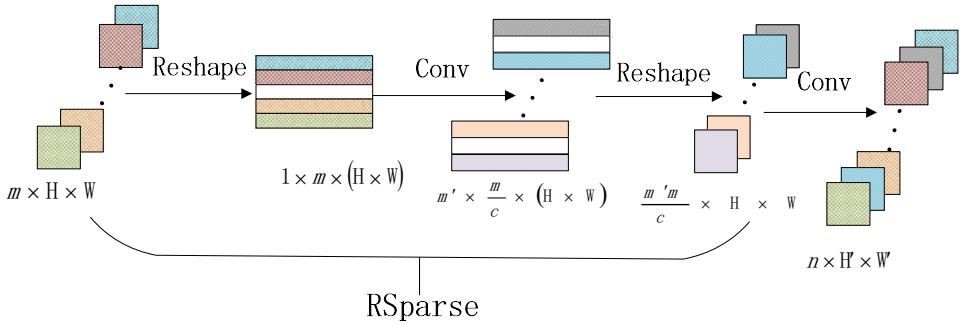

Figure 1. RSparse SVD.

Each convolution layer can be represented by $c \times H \times W$, where c is the number of channels, that is, the number of feature maps, H and W respectively represent the height and width of the feature map. RSparse in Figure 1 is a reduced-dimension compression operation. Firstly, m pieces of $H \times W$ feature maps are reconstructed into a $m \times (H \times W)$ feature map, and then it adopts the convolution kernel of $c \times 1$ and enforce the convolution operation with step size of c to obtain m' feature maps. It's an identical activation function; then it reorganizes the m' pieces of $\frac{m}{c}(H\times W)$ feature maps into $\frac{m'm}{c}$ pieces of $H \times W$ feature maps, and input them into the next convolution layer. The number of parameters of the RSparse SVD network structure is:

$$m' \times c + \frac{m'm}{c} \times n \times k^2 \tag{2.7}$$

In order to reduce the compression dimension, c and m' should be small enough, and $m' < c$, so the c and m' in this experiment are far less than m and n, then the first term of equation (2.7) can be ignored, so the number of parameters of the new CNN model is m'/c of the original CNN.

2.3. Dimension reduction and compression design

There is one operation of convolution and two operations of reorganization in RSparse dimension reduction operation, and both operations use identical activation function. The reason why the convolution layer uses identical activation function is that RSparse itself has the function of dimension reduction compression. If it uses ReLU activation function, namely $Sigma(x) = Max(0, x)$, then the left side is truncated. In addition, RSparse's dimension reduction function may cause most of the information to be lost, thus decrease the performance of the model.

The feed-forward of the convolution layer in RSparse dimension reduction compression and ordinary convolution layer is basically the same as feedback, except that the convolution layer in RSparse uses the identical activation function, in this section we mainly introduce the forward and backward propagation of the reassembly layer.

Assuming the output of the previous layer is a^{l-1}, when it passes through the reassembly layer, the reassembly layer reorganizes the input feature maps firstly, that is,

$$h^l = reshape(a^{l-1}) \tag{2.8}$$

Then, after the mapping of the identical activation function, the feature map a^l is output.

$$a^l = h^l \tag{2.9}$$

The forward propagation of reassembly layer can be represented by Figure 2.

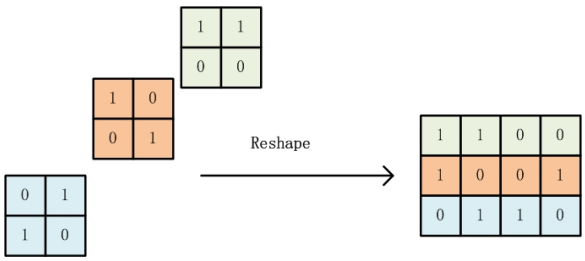

Figure 2. Forward propagation of reassembly layer.

Assuming that an error term in the next layer is δ^L, an upward reorganization recovery operation is performed on the error term, and then the chain derivative is multiplied by the derivative of the identity function, $\frac{da^l}{dh^l} = 1$, so

$$\delta^{l-1} = upreshape(\delta^l) \tag{2.10}$$

The error back propagation of the reassembly layer is shown in Figure 3.

As shown in the above forward propagation and back propagation formulas, the operation of the reassembly layer is very simple. Many open source frameworks, such as Caffe, TensorFlow, have implemented the reassembly layer, so researchers can quickly implement the RSparse layer in real application.

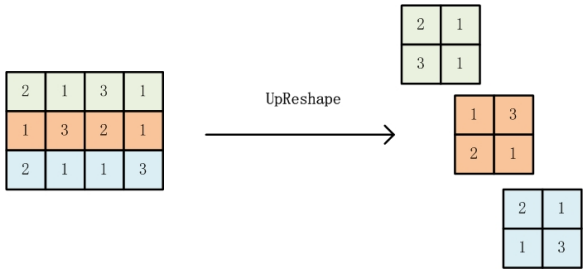

Figure 3. Error back propagation of the reassembly layer.

3. Experiment and analysis

3.1. RSparse SVD network structure

The bottom convolution layer of the GoogleNet model outputs fewer features than the top convolution layer, and the bottom convolution layer of a general convolution neural network model extracts the texture, edge and other information of the image, which is the basis of the convolution neural network model abstraction, so the bottom convolution of GoogleNet does not use RSparse layer to compress for dimension reduction. While, the top convolution layer of GoogleNet output more feature maps, some of them are similar or even identical. In order to avoid the burden of the next convolution layer, RSparse layer is used to compress and reduce the dimension for the top convolution layer in our model, so as to achieve the effect of sparse model. The GoogleNet-RSparse model with RSparse layer compression dimension reduction is shown in Figure 4.

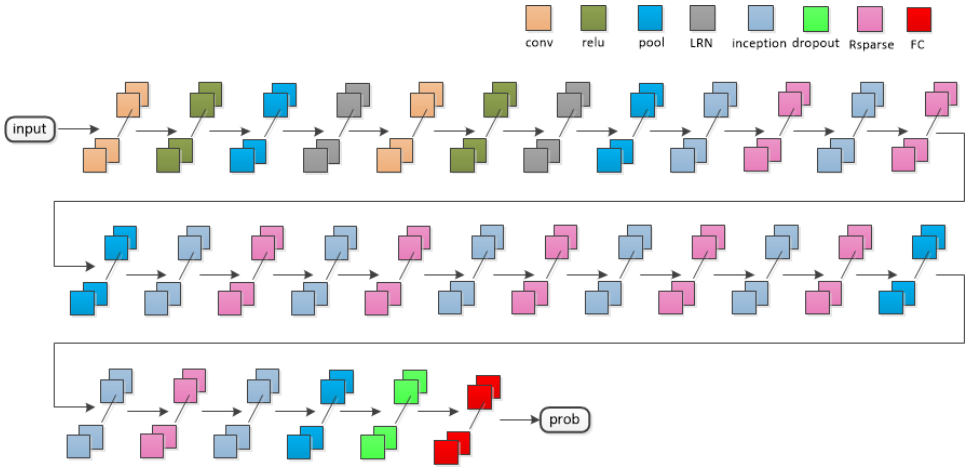

Figure 4. GoogleNet-RSparse model.

Comparing with the plain GoogleNet model, GoogleNet-RSparse adds RSparse layer after each Inception layer, so that the input of the lowest layer of RSparse layer is m'/c input from the plain GoogleNet model. While, m' , $c' < c$, the parameters of Google Net-RSparse are less than GoogleNet model, and the parameters of GoogleNet-RSparse are less than GoogleNet model. Because of the compression and dimension reduction of RSparse, a large amount of noise are removed, so the accuracy of learning is able to be improved.

Because the model with sparse is quite different from the original model, the whole deep model

Table 1. Configuration.

Operating System	CPU	GPU	Memory	Hardisk
Ubuntu16.04	Intel Core i7-6800K	NVIDIA GeForce 1080	32GB	2TB

needs to be retrained. Commonly, firstly, the GoogleNet-RSparse model is trained with the ImageNet data set, and then the parameters of the GoogleNet-RSparse model trained by the ImageNet are transferred to the target task, i.e. the UCMLU data set or the WHU-RS data set. Like sorting tasks. The Google Net-RSparse model trained by ImageNet can get a better solution space for parameters in the whole image classification task, so it can get a better recognition effect when migrating to small-sample data.

3.2. Experiment setting

Our experiments of deep learning models are all implemented in Caffe framework which uses C++/CUDA architecture, supports command line, Python and MATLAB interface, and adopts the Google Protocol Buffer data standard [15]. Since it can improve the efficiency of model training and testing, many scholars adopt Caffe as the framework of model building. The computer hardware and software configuration used in this experiment is shown in Table 1.

Accuracy is the most commonly used performance benchmark in classification tasks. Accuracy is the ratio of the number of samples correctly classified by the classifier to the total number of samples for a given testing data set. For the sample data set D, the accuracy rate is

$$acc(f; D) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{I}(f(x_i) = y_i) \tag{3.1}$$

m is the sample size of the sample data set D, $\mathbb{I}(*)$ is the indicator function and f denotes a classification process. Accuracy is the evaluation of the overall accuracy of the classifier. Commonly, higher accuracy means the more effective of the classifier.

3.3. Analysis

In this section, firstly we compare the classification accuracy of the GoogleNet model with the GoogleNet-RSparse model under various hyper-parameter settings without the use of fine-tuning technique, and then compare the GoogleNet model with the GoogleNet-RSparse model before and after dimension reduction using RSparse layer compression with employing fine-tuning technique.

Without fine-tuning technique, the classification accuracy of the GoogleNet-RSparse model on UCMLU data sets with different hyper-parameter settings of m' and fixed c=8 is shown in Table 2, where c=8 is the result of parameter adjustment experiments. m is a parameter which is used to compress the feature map of the layer before the output feature map of the upper layer is input to the next convolution layer. Because the CNN adopts a complete method, there may be one or more similar feature maps in the output of each convolution layer. In this paper, we compress the output feature map of each layer to simulate the function of SVD. By decomposing the effect of dimension reduction

Table 2. Classification accuracy without fine-tuning technique.

	GoogleNet	(1,8)	(2,8)	(3,8)	(4,8)	(5,8)	(6,8)	(7,8)
UCMLU	0.9167	0.9214	0.9404	0.9309	0.9357	0.9333	0.9476	0.9404
WHU-RS	0.8958	0.8958	0.9114	0.8958	0.9114	0.9010	0.9219	0.9010

and compression, our network can directly learn the weight matrix of the network during training, thus reducing the additional workload required for the strict SVD, and achieving the goal of reducing the number of parameters of the CNN structure and sparsifying the convolution neural network.

Table 2 shows that the plain GoogleNet model has a classification accuracy of 0.9167 on UCMLU data set without the use of fine-tuning technique, while the recognition accuracy of GoogleNet-RSparse model with various m' on UCMLU under the fixed experimental conditions of $c = 8$ is higher than that of the non-sparse GoogleNet. When $(m', c) = (6, 8)$, GoogleNet-RSparse has the highest recognition accuracy, 3% higher than GoogleNet model. While, the GoogleNet model has a classification accuracy of 0.8958 on the WHU-RS data set without the use of fine-tuning technique. Good results can be obtained by selecting the appropriate hyper-parameter m'. When $(m', c) = (6, 8)$, the recognition accuracy of the GoogleNet-RSparse model on the WHU-RS data set is the highest, 2.6% higher than that of GoogleNet model.

Visualizing the feature maps of the Inception 3A layer of GoogleNet and the corresponding Inception 3A layer corresponding to Google Net-RSparse-(2,8) and the next RSparse layer output, as shown in Figure 5.

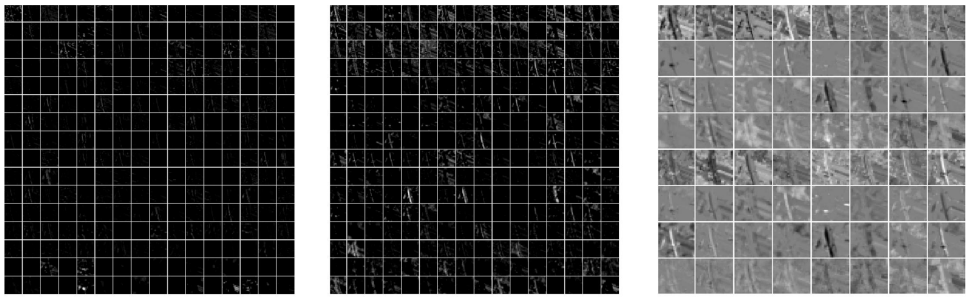

Figure 5. Feature maps output from inception 3A layer before compression and after.

Figure 5(a) is the feature map for the output of Inception 3A layer of GoogleNet model, figure 5(b) is the feature map of Inception 3A layer output of GoogleNet-RSparse model, figure 5(c) is the feature maps for the output of Inception 3A layer of the GoogleNet-RSparse model after compression and dimension reduction by RSparse layer. In CNN, the feature map output by the last layer of convolution layer passes through the global average pooling layer and connect to the classifier. Figure 6 is the histogram obtained by the global average of Figure 5.

As shown in Figure 5 and Figure 6, the inception 3A layer of the GoogleNet model and the GoogleNet-RSparse model have many similar or even identical feature maps, but after dimension reduction and compression of the GoogleNet-RSparse model through the RSparse layer, the result of feature maps are quite different. Therefore, it shows that the GoogleNet-RSparse model in the selec-

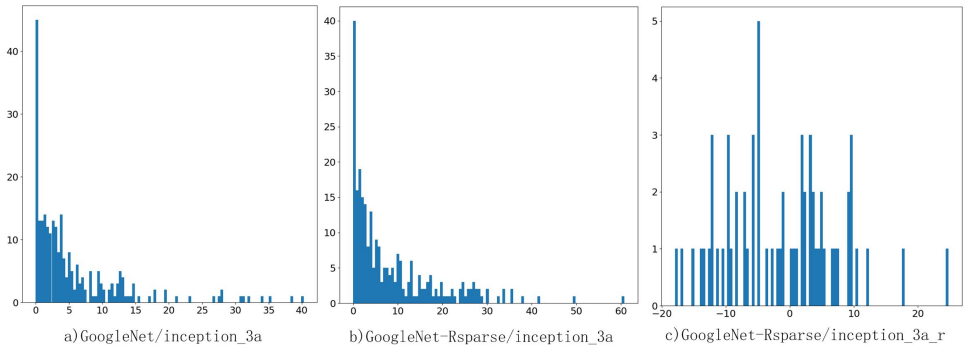

Figure 6. Global average histogram of the feature maps output from inception 3A layer with sparse and without it.

tion of appropriate hyperparameters can preserve sufficient information and reduce the noise into the next convolution layer, so as to compress and reduce the dimension, and make the deep model simpler, which can be better used in remote sensing image classification.

The iteration-error curves of GoogleNet on UCMLU data sets and WHU-RS data sets before and after dimension reduction using RSparse compression are shown in Figure7.

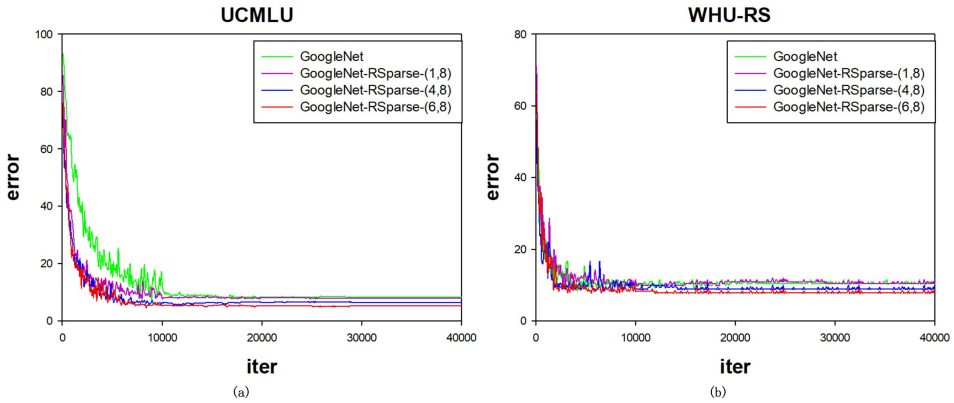

Figure 7. Iteration error curves of googleNet with RSparse compression.

As shown in Figure7, the convergence speed of the model reduced by RSparse compression is obviously faster than that of the GoogleNet, because the model reduced by RSparse compression is much smaller in parameters space than the GoogleNet, and reduces the noise of each convolution layer, so that the fitting of the model is easier. Moreover, because the plain GoogleNet model adopts an over-complete method in each layer of convolution, more feature maps are required to be fitted on the small sample data sets, and these feature maps can not guarantee to be able to distinguish the small sample data sets, so it is possible that the features learned in the training set can not be applied to the testing set. The convergence rate of the GoogleNet-RSparse model on the testing set is faster than the GoogleNet.

The following analysis compares the classification accuracy of the model before and after dimension reduction by RSparse compression under the condition of using fine-tuning technique. Because the model compressed by RSparse layer needs to be retrained from the beginning, two different configurations are selected by Table 2 in the following experiment. In order to obtain faster running speed

Table 3. Classification accuracy with fine-tuning technique.

	GoogleNet	SqueezeNet	GoogleNet-RSparse-(4,8)	GoogleNet-RSparse-(6,8)
UCMLU	0.9761	0.9643	0.9785	0.9904
WHU-RS	0.9947	0.9687	0.9843	0.9947

and good recognition accuracy, Iandola [6] and others designed a very sparse model - SqueezeNet model, which achieved good recognition results on the ImageNet.

The classification accuracy with two different configurations of three models on UCMLU data sets and WHURS data sets is shown in Table 3.

Figure8 is the iterative error curve of the four models using the fine-tuning technique of the deep learning model.

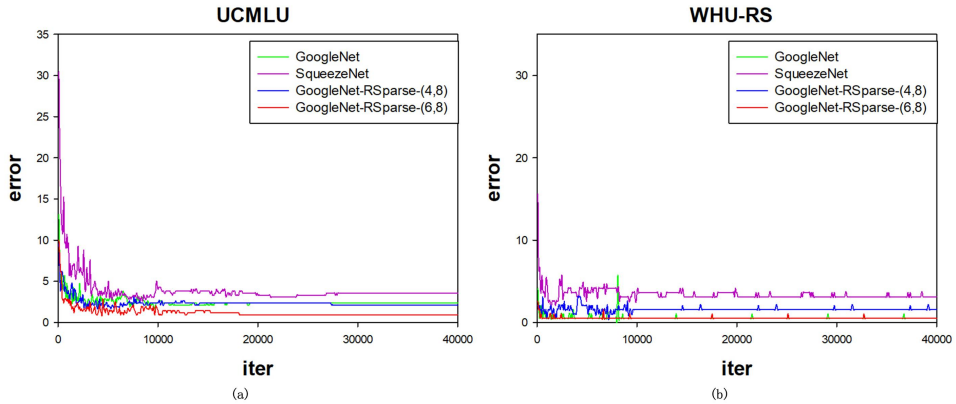

Figure 8. Iterative error curve of sparse model.

As shown in Table 3, the classification accuracy of GoogleNet-RSparse-(4,8) model and plain GoogleNet model are almost the same after using fine-tuning technique. Google Net-RSparse-(4,8) model can reduce the dimension of each layer's convolution output feature maps and theoretically reduce the input noise of each convolution layer. After using the fine-tuning technique, the features learned by the deep learning model have more semantic information and reduce the possibility of generating noise features in large-scale data sets. Therefore, the classification accuracy of GoogleNet-RSparse-(4,8) model is closely same to the plain GoogleNet model with fine-tuning technique. This may be because the sparse method proposed in this paper sacrifices the expressive ability of the model, and loses some information on sparse process. Because the number of remote sensing image samples is small and the learning is insufficient, there is a lot of noise in the model without the fine-tuning technique. In the condition using sparse method can reduce a large amount of noise.

After using fine-tuning technique, the classification accuracy of the GoogleNet-RSparse-(6,8) model is closely same to the plain GoogleNet model on the WHU-RS; but on the UCMLU, the recognition effect of the Google Net-RSparse-(6,8) model is better and the classification accuracy of the Google Net-RSparse-(6,8) model is 1.43% higher than GoogleNet. It shows that the model can be sparsified by SVD in the case of appropriate hyper-parameters, to improve the classification effect of the model on two remote sensing data sets.

In view of model improvement, the GoogleNet model with the sparse method is much better than the plain model. Compared with a sparse model SqueezeNet, it can be seen that the SqueezeNet model has the lowest recognition accuracy among the four models. This is because although the SqueezeNet model is sparsified, it just reduces the output of convolution feature map of each layer. That is to say, it discards the over-complete scheme to achieve a sparse model, but which would result in the model can not cover all the potential feature maps. While, our model dose not discard the over-complete, instead compress the feature maps of each layer of convolution layer. The recognition accuracy of GoogleNet-RSparse model is much better than SqueezeNet because it can cover all the potential feature maps and reduce the noise and the number of similar feature maps.

4. Discussion

In order to obtain enough features, CNNs adopt an over-complete scheme, so there may be one or more similar feature maps in each convolution layer. It would greatly increase the burden of the next layer which leads to generate too many parameters. Therefore, in order to solve the problem of excessive parameters in convolutional neural networks, a sparse model based on SVD idea is proposed. SVD is employed to reduce the input dimension of each layer by compression, so as to reduce the number of parameters. A series of experiments with our method are enforced in different convolutional neural networks such as GoogleNet. The output feature maps of the bottom convolution layer of the GoogleNet model is relative less than the top convolution layer. And commonly the bottom convolution layer of convolution neural network model extracts the data such as the texture and the edge of an image. This information is the basis for the abstraction of the convolutional neural network model. Therefore, for bottom convolution layer, it is not necessary to enforce sparse method. While, for the top convolution layer, the output feature maps are great and some of them are similar or even identical. In order to alleviate the burden of next convolution layer, a RSparse method is proposed to compress the top convolution layer output feature map to reduce the dimension and parameters, which is called "GoogleNet-RSparse". A RSparse layer is injected after each Inception layer, thus the input of RSparse layer is $\frac{m'}{c}$ of plain GoogleNet. m' and c are hyper-parameter, and $m'<c$, so the parameters of GoogleNet-RSparse are less than Plain GoogleNet. Besides, since RSparse reduces the dimension by compression and eliminate a lot of noise, it can help to improve the learning accuracy.

Experimental results show that the proposed method achieve the desired objectives. In the experiments, the relation of classification accuracy and various hyper-parameter configuration is explored. Besides, the output feature maps of plain model and sparse model are visualized and a phenomenon that many output features of plain model are similar or even identical is found. With appropriate configuration of hyper-parameter, sparse model by SVD method can compress the output feature maps and reduce the noise input, thus it not only improve the classification accuracy of the model with small sample data, but also speed up convergence. Finally, experiments with fine-tuning are carried out. The results show that the model has a good performance by using SVD with appropriate hyper-parameters, which can improve the target recognition effect on small sample data sets.

5. Conclusion

We propose a sparse method for deep learning based privacy attacks on remote sensing images. In order to apply deep learning model on small-sample remote sensing images, we employ SVD to compress output feature maps and dimension reduction, thus parameters of the deep model are greatly decreased. Experiment results show that the proposed method can restrain redundant feature map, simplify the deep model and reduce the heavy parameter problems. And it can achieve better classification accuracy than the plain model on small sample remote sensing images. In conclusion, our method helps apply deep learning model into privacy detection on remote sensing images.

Acknowledgments

This research was supported in part by National Natural Science Foundation of China (No.61572157), grant No.2016A030313660 and 2017A030313365 from Guangdong Province Natural Science Foundation, JCYJ20160608161351559, KQJSCX70726103044992, JCYJ20170811155158682 and JCYJ20160428092427867 from Shenzhen Municipal Science and Technology Innovation Project. The authors thank the reviewers for their comments.

Conflict of interest

We declare that there is no conflict of interests regarding the publication of this article.

References

1. K. H. Wang, C. M. Chen, W. Fang et al., On the security of a new ultra-lightweight authentication protocol in iot environment for rfid tags, *J. Supercomput.*, **74** (2017), 65–70.
2. C. M. Chen, B. Xiang, K. H. Wang, et al., A Robust mutual authentication with a key agreement scheme for session initiation protocol, *Appl. Sci.*, **8** (2018).
3. E. Hazan, A. Klivans and Y. Yuan, Hyperparameter optimization: a spectral approach. ICLR 2018.
4. Y. Bengio, Deep learning of representations for unsupervised and transfer learning, *ICML Unsupervised and Transfer Learning*, **27** (2012), 17–36.
5. G. Mesnil, Y. Dauphin, X. Glorot, et al., Unsupervised and Transfer Learning Challenge: a Deep Learning Approach. *ICML Unsupervised and Transfer Learning*, **27** (2012), 97–110.
6. Y. Lecun, J. S. Denker and S. A. Solla, Optimal brain damage. international conference on neural information processing systems. MIT Press, San Mateo, CA: Morgan Kaufmann, (1989), 598–605.
7. S. Han, H. Mao and W. J. Dally, Deep compression: compressing deep neural networks with pruning, trained quantization and huffman coding, *Fiber*, **56** (2015), 3–7.
8. F. N. Iandola, S. Han, M. W. Moskewicz, et al., SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and ¡ 0.5 MB model size. arXiv preprint arXiv:1602.07360, 2016.

9. A. G. Howard, M. Zhu, B. Chen, et al., Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.
10. Y. Jia, E. Shelhamer, J. Donahue, et al., Caffe: Convolutional architecture for fast feature embedding. In Proceedings of the 22nd ACM international conference on Multimedia, *ACM* (2014), 675–678.
11. A. Vedaldi and K. Lenc, Matconvnet: Convolutional neural networks for matlab. In: Proceedings of the 23rd ACM international conference on Multimedia, 2015.
12. I. V. Oseledets. Tensor-Train decomposition. *SIAM J. Sci. Comput.*, **33** (2011), 2295–2317.
13. E. Denton, W. Zaremba, J. Bruna, et al., Exploiting linear structure within convolutional networks for efficient evaluation. In: Advances in Neural Information Processing Systems, 2014.
14. M. Jaderberg, A. Vedaldi, and A. Zisserman, Speeding up convolutional neural networks with low rank expansions. Computer Science. In: Proc. BMVC, 2014.
15. B. Liu, W. Min, H. Foroosh, et al., Sparse convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, (2015), 806–814.
16. C. Kuo and C. Jay, Understanding convolutional neural networks with a mathematical model, *J. Vis. Commun. Image R.*, **41** (2016).
17. Z. Wang, X. Wang, G. Wang, Learning Fine-grained Features via a CNN Tree for Large-scale Classification. Computer Science, 2015.
18. O. Alter, P. O. Brown and D. Botstein, Singular value decomposition for genome-wide expression data processing and modeling. *P. Natl Acad. Sci. USA*, **97** (2000), 10101–10106.

AIMS Press