An Approach of Transferring Pre-trained Deep Convolutional Neural Networks for Aerial Scene Classification

Nilakshi Devi(✉) and Bhogeswar Borah

Department of CSE, Tezpur University, Tezpur 784028, India
{nilakshid,bgb}@tezu.ernet.in

Abstract. Feature selection or feature extraction plays a vital role in image classification task. Since the advent of deep learning methods, significant efforts have been given by researchers to obtain an optimal feature set of images for improving classification performance. Though several deep architectures of Convolutional Neural Networks (CNNs) have been successfully designed but training such deep architectures with small datasets like aerial scenes often leads to overfitting hence affects the classification accuracy. To tackle this issue in past few works, pre-trained CNNs are adopted as feature extractor where features are directly transferred to train only the classification layer for classifying images on the target dataset. In this work, an approach of feature extraction is proposed where both "multi-layer" and "multi-model" features are extracted from pre-trained CNNs. "Multi-layer" features are concatenation of features from multiple layers within a same CNN and "Multi-model" are concatenation of features from different CNN models. The concatenated features are further reduced with some method to obtain an optimal feature set.

Keywords: Convolutional neural network · Feature extraction · Transfer learning

1 Introduction

Aerial scene classification has been receiving remarkable attention due to it's important role in a wide range of applications such as land mapping, natural hazards detection, vegetation mapping, environment monitoring and urban planning. Though CNNs have greatly improved the classification performance of aerial scenes compared to earlier methods with handcrafted features and conventional classifiers but there still exist some issues for CNN-based aerial scene classification. A good classification accuracy depends on the depth of the neural network which requires large number of images for proper training to well-optimize it's parameters. This creates an issue for aerial scene datasets which have not enough samples to train such deep architecture hence leads to overfitting problem for the network.

B. Deka et al. (Eds.): PReMI 2019, LNCS 11941, pp. 551–558, 2019.
https://doi.org/10.1007/978-3-030-34869-4_60

To alleviate the overfitting problem in aerial scene classification, lots of work [3,5,8,12] have been published in transfer learning where the features extracted from either convolutional layers or last fully connected layers of pre-trained CNN models are transferred for scene classification. In transfer learning, the pre-trained models that are trained with some large dataset such as Imagenet, are considered as feature extractor. The extracted features are then fed to either a traditional classifier or a neural network classification layer for classification. Some works have proved that the features extracted from convolutional layers are more generic than features from fully connected layers. In convolutional neural networks, lower layers learn features similar to gabor filter like edges, dots and corners which are not specific to a particular dataset [10] but these are global features applicable to many datasets. As proceeding towards the last layers (fully connected layers) of CNN, high level features which are the combination of low levels features like objects seems to be more specific. Hence each layer has different information that can be combined to obtain a high discriminative feature representation for an image for better performance in scene classification. Such multilayer features of pre-trained CNN are integrated by the proposed fusion strategy in paper [4], where the features extracted from convolutional layers are fused with the features of fully connected layers by a principal component analysis or spectral regression kernel discriminant analysis method. In this work, we have proposed an approach of concatenating both "multi-layer" features and "multi-model" features of pre-trained convolutional neural networks for transferring them to aerial scene classification task.

2 Brief Introduction of the Architecture of Convolutional Neural Networks

The architecture of a convolutional neural network is comprising of three components: convolutional layer, pooling layer and fully-connected layer as shown in Fig. 1. The basic advantage of CNN is that unlike other neural network, each neuron in convolutional layers receives input from only some of the neurons of it's previous layer which reduces the total trainable parameters of the network. The three types of layers in CNN are briefly described below:

Convolutional layer: This is the most significant layer for feature extraction which comprises of a set of filters, each of them extract a particular feature from that image. Each filter is independently convolved throughout the image which end up with the output of a set of feature maps equals to number of filters applied. The size of filter maps after a convolution operation is calculated using the formula: $(W - F + 2P)/S + 1$; W is the input dimension, F is filter size, P is padding and S is stride. The stride is the number of pixels taken at a time to move the filter to fit into next block during convolution.

Pooling layer: This is the layer used for dimension reduction of high dimensional feature maps obtained after applying convolution operation.

Fully connected layer: This layer is mainly used for classification apart from feature extraction in some existing CNN architectures.

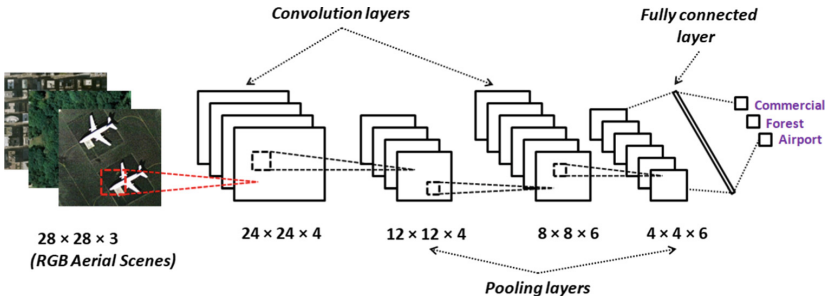

Fig. 1. Architecture of convolutional neural network

3 Methodology

Before taking step towards classification, one have to make sure whether the set of features extracted can well-represent that particular scene or object to be classified. Deep neural networks do feature extraction from the first layer to the last hidden layer at different levels of abstraction. The neurons of the last hidden layer are considered to extract the final features of an image and finally fed to the classification layer or output layer at the end of the neural network. Different types of classifiers can also be adopted instead of neural network classifier as done in many works. Hence the main concentration has to be given on feature extraction part of the deep neural network. In this paper, an approach of extracting features using pre-trained CNN models is proposed where two methods of feature extraction are defined namely, "multi-layer" feature extraction and "multi-model" feature extraction. In other words, we can say those extracted features are either "multi-layer" or "multi-model" features which are further reduced using one-dimensional (1D) convolution operation before fed to a classifier.

Mathematically, $F = \{f_i | i = 1, 2, ..., n\}$ be the feature set of either "multi-layer" or "multi-model" features of size n. After applying 1D convolution on the feature set F, x number of features are extracted from n number of features to reduce the size of final feature set $F' = \{f'_i | i = 1, 2, ..., x\}$ where $x<n$. The proposed framework consisting two architectures is shown in Fig. 2.

In case of "multi-layer" feature extraction, multiple layers of a CNN model are taken as feature extractors where features extracted from the last fully connected layer are concatenated with the convolutional/pooling features from middle or last convolutional/pooling layers. In case the extracted features are convolutional features then global average pooling is applied before proceeding to feature reduction step. Similarly, for "multi-model" feature extraction, features from the last convolutional or pooling layer of multiple CNN models are concatenated to construct a set of "multi-model" features. Finally, the feature set ("multi-layer" or "multi-model") is reduced using 1D convolution operation such that it does not produce a large number of trainable parameters in the single layer neural network for classification. The single layer neural network consists

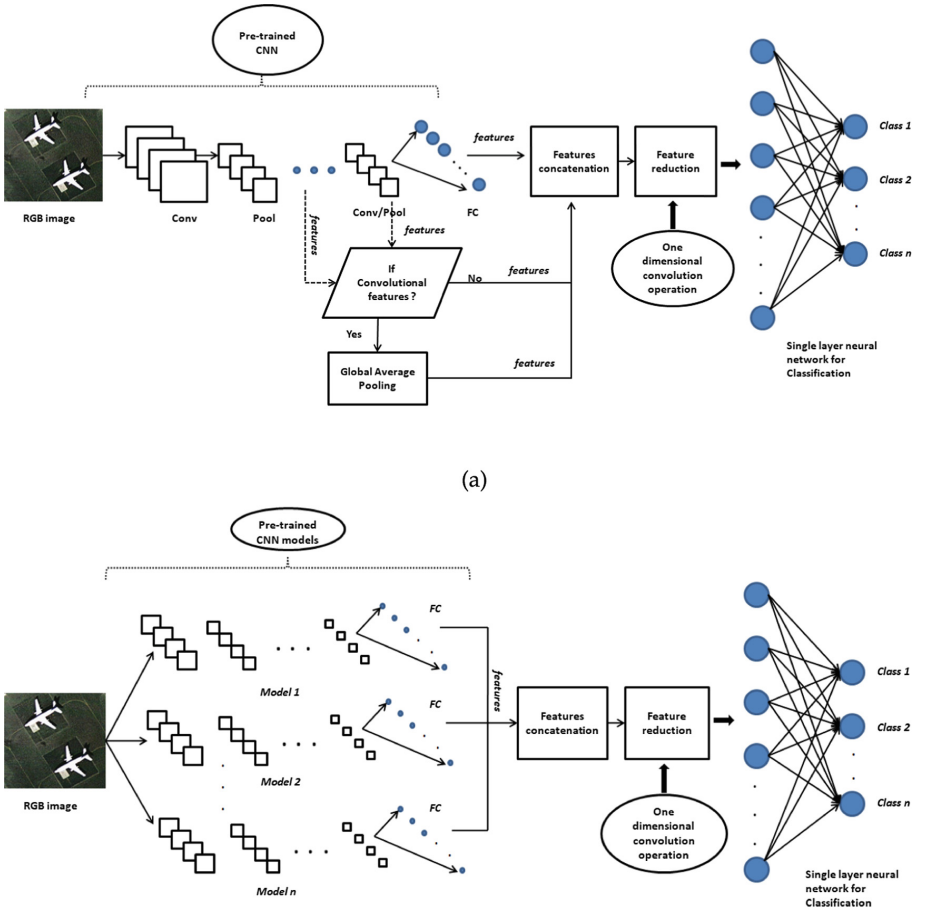

Fig. 2. Proposed framework (a) Architecture I (Multi-layer feature extraction) (b) Architecture II (Multi-model feature extraction)

of a fully connected layer and the total network parameters is calculated as number of features fed to it multiplied by number of scene classes (neurons in output layer) of respective datasets. In our proposed approach, instead of fine-tuning the entire CNN model, only the single layer neural network is trained using three target datasets. Hence before training as well as classification, the final set of "multi-layer" or "multi-model" features is build up by collecting the useful information from multiple layers or multiple models respectively which are considered as feature extractors, but without increasing it's size. Each of the

two feature set is used to train the single layer neural network which acts as a classifier and compared with each other in terms of classification accuracy.

4 Experimental Results

4.1 Datasets Used

For evaluation of the proposed approach, three publicly available aerial scene datasets (containing RGB images) namely, RSSCN7 [13], UC Merced land-use [9] and WHU-RS [6] are used which consist of total 2800 images over 7 categories, 2100 images over 21 categories and 950 images with 19 aerial scene classes respectively. The aerial scene classes are like grassland, forest, farmland, river/lake, mountain, desert, industry, residential and so on.

4.2 Evaluation of Our Proposed Framework

The proposed approach is evaluated using five fold cross validation method where 80% of total images of each dataset are taken for training and the rest 20% for testing. Three existing CNN architectures are adopted to see the performance of our approach, namely: VGG19, Inception v3 and Inceptionresnet v2 which are pre-trained with a very large natural image dataset, Imagenet consisting 1000 image categories. The details of how features are extracted using the proposed framework are given below.

Multi-layer Feature Extraction

VGG19. The architecture of VGG19 can be viewed as five blocks: each of first two blocks have two convolution and one pooling operations and the last three blocks have four convolution followed by one pooling operation each. The features extracted from the last two blocks each having 512 features are concatenated to form a set of 1024 features. Then 1D convolution of 2×2 kernel size with stride 2 is applied to reduce the feature set to size of 512 features which is equal to the last feature extraction layer of the network.

Inception v3. The architecture of Inception v3 consists of five convolution and two pooling operations followed by 11 inception modules. Features extracted from eighth, ninth and eleventh inception module are concatenated to construct "multi-layer" feature set of total size 4096 features and here the feature set is also reduced to the size of last layer before the classification layer which is 2048 by applying 1D convolution with stride 2.

Inceptionresnet v2. Here, the features from layer named "mixed_7a" obtaining 2080 features are concatenated with the features from the last feature extraction layer named "conv_7b" obtaining 1536 features after applying global average pooling to each of the two layer to form the feature set of 3616 features which is reduced to size 1808.

Multi-model Feature Extraction. The features from the last fully connected layers of VGG19, Inception v3 and Inceptionresnet v2 are combined to form a feature set of 4096 features. Like, multi-layer features, this feature set is also reduced by applying 1D convolution operation with stride to 2048 features.

Table 1. Classification accuracies of recent state-of-the-art methods and the proposed methods on UC merced dataset

Method	Year	Classification accuracy (%)
GoogleNet + finetune [2]	2015	97.10
Multi-scale ADPM [5]	2016	94.86
LQPCANet [8]	2017	96.75
vgg19_EMR [7]	2017	97.90
Resnet512 (intermediate) + vgg19 [7]	2017	99.48
TEX-Nets [1]	2018	97.72
Inception-v3-CapsNet [12]	2019	99.05 ± 0.24
Architecture I (vgg19) (ours)	–	**98.52**
Architecture I (Inception v3) (ours)	–	**99.22**
Architecture I (Inceptionresnet v2) (ours)	–	**99.28**
Architecture II (ours)	–	**97.44**

The classification accuracies of our proposed framework with respect to each of three aerial scene datasets and their comparison with some recent state-of-the-art methods are shown in Tables 1, 2 and 3 respectively. The two proposed architectures beat most of the state-of-the-art works in terms of classification accuracy. One possible reason for which architecture I performs much better than architecture II is that in architecture II, there is a high chance that the

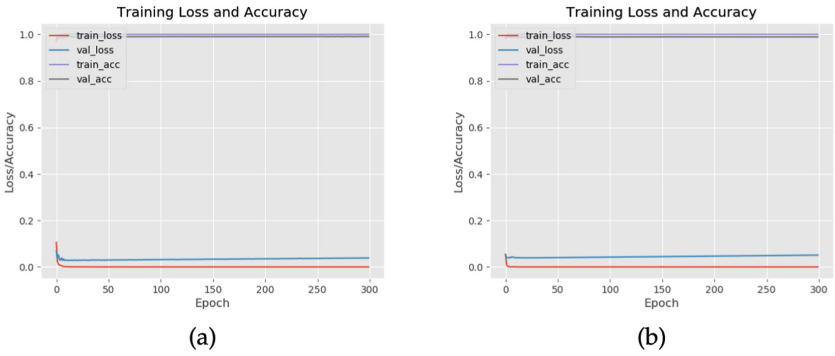

Fig. 3. Loss and accuracy curve of UC merced land use dataset (a) Architecture I (b) Architecture II

Table 2. Classification accuracies of recent state-of-the-art methods and the proposed methods on WHU_RS dataset

Method	Year	Classification accuracy (%)
Multi-scale ADPM [5]	2016	84.67
LQPCANet [8]	2017	96.22
TEX-Nets [1]	2018	98.20
Architecture I (vgg19) (ours)	–	**98.69**
Architecture I (Inception v3) (ours)	–	**98.97**
Architecture I (Inceptionresnet v2) (ours)	–	**98.99**
Architecture II (ours)	-	**98.82**

Table 3. Classification accuracies of recent state-of-the-art methods and the proposed methods on RSSCN7 dataset

Method	Year	Classification accuracy (%)
TEX-Nets [1]	2018	94.00
Global + Local [11]	2018	95.59 ± 0.49
Architecture I (vgg19) (ours)	–	**95.30**
Architecture I (Inception v3) (ours)	–	**95.77**
Architecture I (Inceptionresnet v2) (ours)	–	**95.74**
Architecture II (ours)	–	**93.74**

final feature set consists of redundant features. Features are collected from the last layer of different CNN models where each model may produce some common features. Due to this feature redundancy problem, the gap between the loss curves of training and validation (also called overfitting gap) in architecture II is slightly more than that of architecture I as shown in Fig. 3 that can also affect the classification results.

5 Conclusion

Each layer of a neural network extracts different information about an image so it is always a good idea of considering multiple layers of the network as feature extractors in transfer learning. Moreover, each state-of-the-art CNN model may not extract exactly same set of features of the image. So in the proposed framework, each of the architecture defines a way of feature extraction namely, "multi-layer" and "multi-model" feature extraction respectively using pre-trained CNNs. Each feature set are then reduced using convolution operation to lower the computation cost during classification. The implementation of the proposed approach is done in keras with tensorflow 1.7.0 as backend. The python version is 3.6.4 with cache memory 30720 KB and CPU speed 2.50 GHz.

References

1. Anwer, R.M., Khan, F.S., van de Weijer, J., Molinier, M., Laaksonen, J.: Binary patterns encoded convolutional neural networks for texture recognition and remote sensing scene classification. ISPRS J. Photogramm. Remote Sens. **138**, 74–85 (2018)
2. Castelluccio, M., Poggi, G., Sansone, C., Verdoliva, L.: Land use classification in remote sensing images by convolutional neural networks. arXiv preprint arXiv:1508.00092 (2015)
3. Hu, F., Xia, G.S., Hu, J., Zhang, L.: Transferring deep convolutional neural networks for the scene classification of high-resolution remote sensing imagery. Remote Sens. **7**(11), 14680–14707 (2015)
4. Li, E., Xia, J., Du, P., Lin, C., Samat, A.: Integrating multilayer features of convolutional neural networks for remote sensing scene classification. IEEE Trans. Geosci. Remote Sens. **55**(10), 5653–5665 (2017)
5. Liu, Q., Hang, R., Song, H., Zhu, F., Plaza, J., Plaza, A.: Adaptive deep pyramid matching for remote sensing scene classification. arXiv preprint arXiv:1611.03589 (2016)
6. Sheng, G., Yang, W., Xu, T., Sun, H.: High-resolution satellite scene classification using a sparse coding based multiple feature combination. Int. J. Remote Sens. **33**(8), 2395–2412 (2012)
7. Wang, G., Fan, B., Xiang, S., Pan, C.: Aggregating rich hierarchical features for scene classification in remote sensing imagery. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **10**(9), 4104–4115 (2017)
8. Wang, J., Luo, C., Huang, H., Zhao, H., Wang, S.: Transferring pre-trained deep cnns for remote scene classification with general features learned from linear pca network. Remote Sens. **9**(3), 225 (2017)
9. Yang, Y., Newsam, S.: Bag-of-visual-words and spatial extensions for land-use classification. In: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, pp. 270–279. ACM (2010)
10. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Advances in Neural Information Processing Systems, pp. 3320–3328 (2014)
11. Zeng, D., Chen, S., Chen, B., Li, S.: Improving remote sensing scene classification by integrating global-context and local-object features. Remote Sens. **10**(5), 734 (2018)
12. Zhang, W., Tang, P., Zhao, L.: Remote sensing image scene classification using CNN-CapsNet. Remote Sens. **11**(5), 494 (2019)
13. Zou, Q., Ni, L., Zhang, T., Wang, Q.: Deep learning based feature selection for remote sensing scene classification. IEEE Geosci. Remote Sens. Lett. **12**(11), 2321–2325 (2015)