Classification of Small-Scale Hyperspectral Images with Multi-Source Deep Transfer Learning

Xin Zhao[a], Yi Liang[a], Alan J.X. Guo[a] , and Fei Zhu[a]

[a] Center for Applied Mathematics, Tianjin University, Tianjin, China.

ARTICLE HISTORY
Compiled January 14, 2020

ABSTRACT
Recently, many methods based on deep learning (DL) have been used for hyperspectral image (HSI) classification and achieved good performance. But such approaches often need numerous labeled training samples. This issue is aggravated when applying DL on small-scale HSIs. To alleviate this problem, transfer learning (TL) is introduced to HSI analysis by some existing works. Most of these works transfer knowledge from a single source domain to the target domain. However, the single source TL tends to learn specific knowledge instead of general knowledge. Moreover, since the samples are limited in one source domain, it only partially alleviates the shortage of labeled samples. To learn more general knowledge and further alleviate the issue of limited samples, we introduce the multi-source transfer learning strategy to classify HSIs. Specifically, a framework named multi-source deep transfer learning (MS-DTL) is proposed. This framework consists of a multi-source compatible model and a customized loss function. We perform experiments by comparing the proposed method with the baseline methods on the well-known hyperspectral datasets. The results show that the proposed MS-DTL performs better than the benchmarks on the classification tasks of the small-scale HSIs. Thanks to the strategy of TL, the proposed network is also time-saving.

KEYWORDS
Deep transfer learning; hyperspectral image classification; resnet

1. Introduction

Different from the single-channeled grayscale images and the three-channeled RGB images, hyperspectral images (HSIs) have complex structures with hundreds of spectral channels spanning from the visible spectrum to the infrared spectrum. Due to the abundant spectral information, techniques utilizing HSIs have been applied to many fields, for instance, food safety, biomedicine, industry, biometric, etc. (Bioucas-Dias et al. 2013). In the field of HSI analysis, the classification task, which aims at assigning pixels to specific classes based on their spectral or spatial-spectral characters, is one of the most popular topics (Chen et al. 2014). However, manually labeling HSI is laborious and expensive, hence the annotated HSI data is not enough in most cases of classification tasks. Nowadays, hyperspectral image classification (HSIC) is still a challenging task (He et al. 2018).

CONTACT Alan J.X. Guo., Email: jiaxiang.guo@tju.edu.cn, Address: Center for Applied Mathematics, Tianjin University, Tianjin 300072, China.

Over the past two decades, various methods have been proposed to deal with these issues on HSIC. Nowadays, some conventional ways to classify HSI data are still of reference value, for example, support vector machines (SVMs) (Melgani and Bruzzone 2004), extended morphological profiles (EMPs) (Benediktsson, Palmason, and Sveinsson 2005), kernel sparse representation (KSR) (Chen, Nasrabadi, and Tran 2013), etc. In recent years, deep learning (DL) has made great achievements in the fields of computer vision and artificial intelligence (Han et al. 2018). As the core part of the DL, the neural networks are capable of extracting abstract, complex, and hierarchical features automatically. Furthermore, convolutional neural network (CNN) could be easily applied to process the spectral-spatial information integrally. Nowadays, several DL-based trials have been applied to HSIC. Typical deep neural network models include stacked autoencoder (SAE) (Chen et al. 2014), deep belief network (DBN) (Chen, Zhao, and Jia 2015), and CNN (Romero, Gatta, and Camps-Valls 2016). Recently, in the field of computer vision, CNN has made great breakthroughs by increasing its model depth (Zhong et al. 2018). Residential network (ResNet) (He et al. 2016), fully convolutional networks (FCN) (Long, Shelhamer, and Darrell 2015), and densely connected convolutional network (DenseNet) (Wang et al. 2018) are representatives of the pretty deep models.

Several solutions are aiming at solving the problem of the limited annotated samples, such as active learning (Deng et al. 2018; Deng et al. 2019) and semi-supervised learning (Wu and Prasad 2018). Another useful idea is transfer learning (TL) strategy, which can effectively alleviate the difficulty of applying deep learning to small-scale samples. Since proposed in (Pratt 1993), TL has become a hotspot in DL. The core idea of TL is to learn knowledge from a source domain with abundant data and apply the learned knowledge to a target domain with insufficient data (Pan and Yang 2010). In (Yosinski et al. 2014), it was shown that fine tuning after transferring a deep network can overcome the difference between source domain and target domain, hence have better generalization ability. Following this idea, researchers have applied TL to HSI analysis in several works. In (Jiao et al. 2017), the authors adopted the ImageNet (Deng et al. 2009) of RGB images to pre-train a VGG-16 network (Simonyan and Zisserman 2015) and engaged this pre-trained network as the spatial feature extraction module of their HSIC algorithm. Several works used HSI dataset as the source domain (Windrim et al. 2018; Yang, Zhao, and Chan 2017; Lin, Ward, and Wang 2018). In (Windrim et al. 2018), spectra from public HSI datasets were cut into components of the visible and near infrared (VNIR) and the short-wave infrared (SWIR) for the source domain. Later, the pre-trained networks were transferred for the field-based applications. The work in (Yang, Zhao, and Chan 2017) transferred the networks between datasets obtained from the same sensor. While in (Lin, Ward, and Wang 2018), the authors used canonical correlation analysis (CCA) to transfer knowledge between two SAEs that were trained by source data and target data independently.

In this letter, we introduce the multiple sources transfer learning (Yao and Doretto 2010) to boost the deep neural network for classifying the HSIs, especially the small-scaled ones. Breaking the restrictions on source and target datasets, the proposed multi-source deep transfer learning (MS-DTL) model is designed to be able to learn from multi-source HSI datasets and transfer knowledge to the target domain. To be more precise, the main contributions of the present letter can be summarized as follows.

- To transfer knowledge from multiple sources, we design a deep model with a compatible structure. Cooperating with this deep model, a customized loss function that packages the losses from every single source is applied for backpropagation.

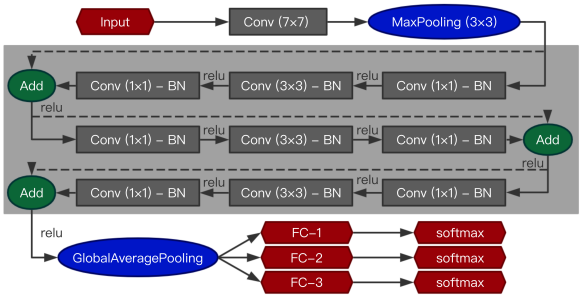

Figure 1. Structure of the model for source domain

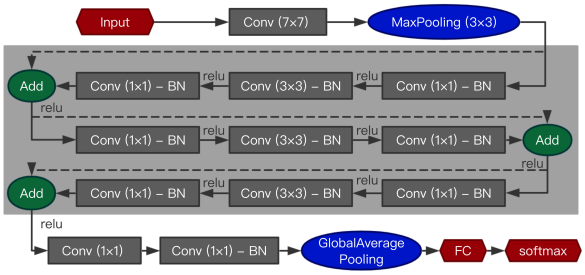

Figure 2. Structure of the model for target domain

The model we proposed is capable of learning from multiple tasks in the source domain and easy to extend to an arbitrary target domain.

- The transferred model is designed to be a fully convolutional network which breaks the limit that the source data should have the same spatial size with the target data.

By these efforts, the proposed MS-DTL shows competitive performance in both classification accuracy and time cost on small-scale HSI datasets.

2. Proposed framework

In this section, we present a detailed description of the proposed framework.

2.1. *Training a model with multiple HSIs*

The existing works that apply TL to HSIC tasks usually have restrictions on the source and target domains. In these works, data from the source and target domains should have the same size or should be obtained by the same sensor. These restrictions are set to make better compatibility between the source and target domains, but they also lead to some weaknesses. Under these restrictions, the source domain is usually limited to one specific HSI dataset. Even the source data are chosen with relatively large cardinality, they are still not comparable with abundant RGB images. The limited training samples may lead to overfitting in the source domain. Superior to these works, we propose a multi-source domain strategy, which means the source data are from different datasets.

In this letter, we use a ResNet-based model, which is called the base model in the following, to learn from multiple hyperspectral datasets in the source domain. The

model's structure is shown in Figure 1. The ResNet has advantages in both extracting hierarchical features and alleviating the following issues. When the depth of neural network increases, the main problem of gradient vanishing/exploding rises (Bengio, Simard, and Frasconi 1994). To solve this problem, the authors of ResNet (He et al. 2016) proposed a deep residual learning framework by making neural network layers to learn the residual mapping $\mathcal{R}(\cdot)$ instead of the desired underlying mapping $\mathcal{H}(\cdot)$. As illustrated in the light gray shaded area of Figure 1, the solid arrows indicate the network that behaves the residual mapping $\mathcal{R}(\cdot)$, while the dashed arrows represent the identity mapping $\mathcal{I}(\cdot)$. Combining these two mappings by addition, the network fits the desired underlying mapping by

$$\mathcal{H}(\boldsymbol{x}) = \mathcal{R}(\boldsymbol{x}) + \mathcal{I}(\boldsymbol{x}), \tag{1}$$

where $\boldsymbol{x}$ represents the input vector of the layers considered. Under the setting that the network treats the residual mapping and the identity mapping separately, the behavior of the residual block is prone to fit mappings that range from the identity mapping $\mathcal{I}(\cdot)$ to the desired underlying mapping $\mathcal{H}(\boldsymbol{x})$. Three continuous convolutional layers and an additional layer form a residual block in Figure 1.

In this letter, we choose three aforementioned residual blocks to form the base model, as illustrated in Figure 1. After the final residual block, a global average pooling (GAP) layer is concatenated, which outputs features in a fixed length. Further, each branch is constituted by a fully connected (FC) layer and a softmax layer. These branches are connected to the GAP layer to give predictions on different sources and to back propagate the gradients. The number of branches should be the same as the number of sources, which is set to be three in this letter.

To describe the multi-source training stage clearly, we will first introduce some mathematical notations. In our experiments, $\boldsymbol{x}$ represents the input HSI patch, which refers to a square pixel patch labeled according to its centre pixel. Assuming that we have M HSI sources to train the base model. The training samples from the K-th source are denoted by $(\boldsymbol{x}_{K,i}, y_{K,i}), i = 1, 2, \cdots, m_K$, where m_K represents the total number of the training samples, $\boldsymbol{x}_{K,i}$ is the i-th HSI patch and $y_{K,i}$ is the corresponding ground truth label, in the K-th source. Focus on a branch K of our model, i.e. the K-th source. Let $p_K(\hat{y} = i \,|\, \boldsymbol{x})$ be the chance that predicted label equals to i on the HSI patch $\boldsymbol{x}$ from the branch K, which is the i-th component in the output vector of the softmax layer for the branch K. Following these notations, the commonly used estimation of cross-entropy loss $\mathcal{L}_K$ is formulated as:

$$\mathcal{L}_K(\boldsymbol{x}, y) = -\ln(p_K(\hat{y} = y \,|\, \boldsymbol{x})). \tag{2}$$

To back propagate properly, each branch should only respond to one source, we have the multi-source loss $\mathcal{L}$ on a training sample $(\boldsymbol{x}_{K,i}, y_{K,i})$, which is given by

$$\mathcal{L}(\boldsymbol{x}_{K,i}, y_{K,i}) = \sum_{I=1}^{M} \delta_{I,K} \mathcal{L}_K(\boldsymbol{x}_{K,i}, y_{K,i}), \tag{3}$$

where the $\delta_{I,K}$ is the Kronecker delta, defined by

$$\delta_{I,K} = \begin{cases} 1, & I = K; \\ 0, & I \neq K. \end{cases} \tag{4}$$

Note that in Equation (3), the loss $\mathcal{L}$ sums over all the sources, while the Kronecker delta controls each branch only responds to one source. In the end, we calculate gradients of the loss $\mathcal{L}$ on the parameters and apply the optimization methods to optimize the base model.

2.2. *Transfer knowledge to target HSI*

Generally speaking, transfer learning is to reuse the pre-trained network to a new model for new tasks. In the field of neural network, the weight of each node in each layer is transferred from the pre-trained network to another brand-new network (Yosinski et al. 2014). The migrated network does not need to be trained from scratch and can be directly used to extract features.

In this letter, we transfer the base model trained by multi-source in Section 2.1 to the target domain. Under the assumption that the pre-trained base model is capable of extracting universal features from HSIs for classification, we build the model for target source. Firstly, we remove the structure of branches and the GAP layer from the model in Figure 1. The remaining structure of the base model is called as the remained base model in this letter. Layer GAP has an inherent attribute that calculating the mean of all the pixels on each feature map and then outputting a value, which destroys the spatial structure of the extracted features. In this way, the transferred model is designed as a fully convolutional network, hence the spatial size of the input target data can be any size. Secondly, an additional structure is concatenated to the remained base model, and the resulting model is used in the target domain, as shown in Figure 2. This new adding structure is constituted by two convolutional layers, one GAP layer, one fully connected layer and one softmax layer, which is called as the additional model in this letter. During the training stage of the target domain, the weights in the pre-trained base model are transferred and kept invariant. We feed the target data to the remained base model and directly utilize the resulting output as the input of the additional model, that is, using the resulting output to train the additional model from scratch. This approach makes the new branch could adapt to the new classification task of the target domain. While training, a cross-entropy loss is engaged in back propagating in the additional layers of the model. From another point of view, we could regard the base model as a feature extractor to extract features from HSIs, and regard the additional model as a classifier, based on a shallow neural network, to classify the extracted features.

3. Experimental results and analysis

3.1. *Datasets and quantitative metrics*

Our experiments are carried out on five public available hyperspectral datasets, namely the Pavia Centre (PC) scene, the Pavia University (PU) scene, the Salinas (SA) scene, the Indian Pines (IP) scene, and the Botswana (BO) Scene. From these five datasets, the PC scene, PU scene, and SA scene are chosen for the source domains, while the other two scenes are chosen for the target domains. The detailed information about these five datasets is shown in Table 1.

We adopt the universal quantitative metrics, that is, overall accuracy (OA), average accuracy (AA) and the kappa coefficient (κ), to evaluate the classification performance of the model. The metric OA is the ratio between the number of correctly predicted

Table 1. Basic attributes of the datasets

Dataset	Spatial size (pixels)	No. of spectral bands	Spectral range(μm)	Spatial resolution (m)	No. of labeled classes	Sensor
Pavia University	610×340	103	$0.43 - 0.86$	1.3	9	ROSIS
Pavia Centre	1096×715	102	$0.43 - 0.86$	1.3	9	ROSIS
Salinas	512×217	204	$0.2 - 2.4$	3.7	16	AVIRIS
Indian Pines	145×145	200	$0.2 - 2.4$	20	16	AVIRIS
Botswana	1476×256	145	$0.4 - 2.5$	30	14	NASA EO-1

samples and the total samples from a dataset, while the metric AA is the average number of the accuracies from each class. κ is a measure of agreement between the predicted labels and the ground truth. It is defined as $\kappa = (p_0 - p_e)/(1 - p_e)$, where p_0 represents the probability that a predicted label is the same as the ground truth, which equals to the metric OA, and p_e represents the hypothetical probability of chance agreement. It is calculated by

$$p_e = (m_1 \times n_1 + m_2 \times n_2 + ... + m_C \times n_C)/(N \times N), \tag{5}$$

where C is the total number of categories, N is the total number of samples, m_i represents the number of samples in class i, and n_i represents the number of predicted labels which equal to class i , where i ranges from 1 to C.

3.2. *Experimental setup and parameter analysis*

In the experiments, all five datasets are preprocessed by the following steps. Firstly, each raw HSI dataset is normalized to have zero mean and unit variance. Secondly, we use principal component analysis (PCA) (Farrell and Mersereau 2005) to process each pixel of the HSIs and retain the first 30 principal components (PCs) of the spectral dimensions to make sure that all the datasets have the same spectral dimensionality. Finally, we cut the resulting HSIs into patches for training the model.

In the experiments, the PU scene, the PC scene, and the SA scene are used to train the base model. To form the multi-source data, we randomly choose 2000 patches from each class, for each of the three datasets. If some classes have fewer samples than 2000, we simply duplicate its samples to 2000 randomly.

We use the small-scale datasets, the IP scene and the BO scene, as target domains. In respect of the IP scene, we randomly choose 10% of annotated patches as training samples, 10% as validation samples, and the remained 80% as testing samples. For the BO scene, 5%, 15%, and 80% annotated patches are randomly selected for training, validation, and testing, respectively.

As shown in Figure 1, the base model is mainly composed of a convolutional layer with 256 filters and three residual blocks. The numbers of the convolutional filters in the first and second residual block are both $(64, 64, 256)$, while in the third residual block, these numbers are set to be $(128, 128, 256)$. The sizes of convolutional kernels are set to be 1×1 pixels and 3×3 pixels in the residual blocks, and 7×7 pixels in the leading convolutional layer, which are the same with the ResNet in (He et al. 2016) and is indicated by the numbers in Figure 1. After each convolutional layer in the residual blocks, there is a Batch Normalization (BN) layer (Ioffe and Szegedy 2015). When training in the source domains, the fully connected layer in each branch is set to be compatible with the number of classes in its corresponding source dataset. As shown in Figure 2, the additional structure in the target domain model uses the 1×1 pixels kernelled convolutional layers. The numbers of filters in the two additional

Table 2. Classification overall accuracy (averaged over 10 runs) of different sizes of patches from Indian Pines scene and Botswana scene

	Patch size (pixels)					
Dataset	9×9	11×11	13×13	15×15	17×17	19×19
Indian Pines	96.35 ± 0.36	97.42 ± 0.51	97.53 ± 0.22	97.41 ± 0.54	97.25 ± 0.41	97.62 ± 0.44
Botswana	98.85 ± 0.38	98.58 ± 0.91	98.69 ± 0.41	98.55 ± 0.46	98.34 ± 0.67	98.21 ± 0.77

Table 3. Classification results (averaged over 10 runs) of MS-DTL, TFS, SS-DTL, 3D-CNN, and FDSSC on Indian Pines scene and Botswana scene

		Method				
Dataset		MS-DTL	TFS	SS-DTL	3D-CNN	FDSSC
Indian Pines	AA(%)	97.79 ± 0.39	96.96 ± 1.00	94.42 ± 1.50	97.29 ± 0.69	93.16 ± 5.54
	OA(%)	97.53 ± 0.22	96.77 ± 0.75	95.74 ± 1.21	96.12 ± 0.36	92.03 ± 7.40
	κ	0.9718 ± 0.0026	0.9630 ± 0.0086	0.9388 ± 0.0160	0.9557 ± 0.0041	0.9078 ± 0.0878
Botswana	AA(%)	98.83 ± 0.53	97.86 ± 0.81	-	95.48 ± 0.92	98.87 ± 0.49
	OA(%)	98.85 ± 0.38	97.82 ± 0.82	-	95.50 ± 0.67	98.67 ± 0.71
	κ	0.9875 ± 0.0041	0.9763 ± 0.0089	-	0.9512 ± 0.0073	0.9856 ± 0.0077

convolutional layers are set to be 128 and 256. In all the aforementioned models, the parameters in the BN layer and the GAP layer are all set as default. In all experiments, RMSprop (Ruder 2016) with default parameters is adopted to optimize the networks.

The sizes of patches from the source datasets and the target datasets control the spatial information that are taken into account in our algorithm, hence they are the two most important hyperparameters. Since the remained base model is designed to be a fully convolutional network, these two parameters are not required to be the same. According to the different spectral characteristics of the datasets, we retain the appropriate spatial size of each dataset, which takes full account of the spatial information about different datasets. To simplify the experiments, when training the base model, the size of patches is fixed as 11×11 pixels. When training the additional structure, the sizes of patches are set to be 13×13 pixels and 9×9 pixels for the IP scene and the BO scene respectively. The effect of different patch sizes for the IP scene and the BO scene is reported in Table 2. As we can see, different spatial sizes could influence the final classification accuracy, and it is effective to make the source data and the target data have different spatial sizes.

3.3. *Comparison with different methods and results analysis*

We have some experiments for comparison. Firstly, the experiments, which engage the same model structure with the MS-DTL but are trained from scratch (TFS) by the training data from the IP scene or the BO scene, are carried out. Secondly, instead of transferring knowledge from multi-source, we transfer knowledge from one dataset to another collected by the same sensor, which is the single source deep transfer learning

Table 4. Comparison of training time (in seconds)

	Method			
Dataset	MS-DTL	TFS	3D-CNN	FDSSC
Indian Pines	17.52	27.94	78.17	1683.04
Botswana	5.19	13.70	19.49	589.34

(SS-DTL). The SS-DTL shares the same model structure with the MS-DTL. In this experiment, the base model and additional structure are trained with the SA scene and the IP scene, respectively. Finally, we choose a classic model and a DenseNet-based state-of-art model, which are the 3D-CNN (Chen et al. 2016) and the FDSSC (Wang et al. 2018), as our baselines. In the contrast experiments, all the experimental settings, such as the ratio of training samples, the size of patches, are the same as the MS-DTL. We list the experimental results of the proposed MS-DTL and contrast experiments in Table 3. The results are formed of the mean values and the standard deviations over 10 runs.

In (Donahue et al. 2014), the authors showed that the shallow layers learn low-level universal features, whereas the deep layers learn high-level specific features. Moreover, in (Huh, Agrawal, and Efros 2016), the authors illustrated that by increasing the number of classes in the source data set, more universal features can be learned and more knowledge can be transferred to target learning tasks. In this letter, we transfer the shallow layers and increase the number of classes by using multiple sources to extract universal features and improve performance of the proposed method. As one could see, the comparison of the first two columns of Table 3 shows that the strategy of transferring knowledge from multi-source performs better than directly training a deep model from scratch, when the dataset is small. This indicates that the MS-DTL strategy helps extract better features. When comparing the first and third columns of Table 3, although the source HSI and the target HSI are chosen to be collected by the same sensor, the results from SS-DTL indicate that transferring from only one HSI performs worse than the MS-DTL and the TFS in this letter. This illustrates that when we transfer from a single source, the model may learn dataset-specific knowledge, which may even be negative knowledge instead of universal knowledge.

Compared to the baseline algorithm 3D-CNN, the proposed MS-DTL shows overwhelming advantage in classification accuracies, especially on the BO scene. The MS-DTL also outperforms the FDSSC on the IP scene. It is worth mentioning that because only the small amount of parameters from the additional structure need to be trained, the MS-DTL greatly reduces the training time. The training time is listed in Table 4, as one can see, the FDSSC takes about a hundred times than the MS-DTL in time cost. Also, the MS-DTL occupies 50% less in time than the TFS algorithm. Generally speaking, benefiting from the deep transfer model structure, the MS-DTL model performs its advantages in both classification accuracy and training time.

4. Conclusion

In this letter, we proposed the MS-DTL framework to classify the small-scale HSIs. The ResNet-like base model was designed to meet the requirements of the multiple sources and a single target. To train the base model, we proposed a loss that combined the cross-entropy losses from each source HSI. After having learned universal knowledge from multiple HSIs, the proposed framework only needed limited training samples. The experiments showed that the proposed framework performed better than both the single-source algorithm and the algorithm that directly trained from scratch, on small-scale HSIs. Comparing to the state-of-art algorithms, the MS-DTL also provided competitive results in both the classification accuracies and time cost.

5. Funding

The work was supported in part by the National Natural Science Foundation of China under Grant 11801409, the National Natural Science Foundation of China under Grant 61701337, and the Natural Science Foundation of Tianjin under Grand 18JCQNJC01600.

References

Benediktsson, J. A., J. A. Palmason, and J. R. Sveinsson. 2005. "Classification of hyperspectral data from urban areas based on extended morphological profiles." *IEEE Transactions on Geoscience and Remote Sensing* 43 (3): 480–491.

Bengio, Y., P. Simard, and P. Frasconi. 1994. "Learning long-term dependencies with gradient descent is difficult." *IEEE Transactions on Neural Networks* 5 (2): 157–166.

Bioucas-Dias, J. M., A. Plaza, G. Camps-Valls, P. Scheunders, N. Nasrabadi, and J. Chanussot. 2013. "Hyperspectral Remote Sensing Data Analysis and Future Challenges." *IEEE Geoscience and Remote Sensing Magazine* 1 (2): 6–36.

Chen, Y., H. Jiang, C. Li, X. Jia, and P. Ghamisi. 2016. "Deep Feature Extraction and Classification of Hyperspectral Images Based on Convolutional Neural Networks." *IEEE Transactions on Geoscience and Remote Sensing* 54 (10): 6232–6251.

Chen, Y., Z. Lin, X. Zhao, G. Wang, and Y. Gu. 2014. "Deep Learning-Based Classification of Hyperspectral Data." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 7 (6): 2094–2107.

Chen, Y., N. M. Nasrabadi, and T. D. Tran. 2013. "Hyperspectral Image Classification via Kernel Sparse Representation." *IEEE Transactions on Geoscience and Remote Sensing* 51 (1): 217–231.

Chen, Y., X. Zhao, and X. Jia. 2015. "Spectral-Spatial Classification of Hyperspectral Data Based on Deep Belief Network." *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 8 (6): 2381–2392.

Deng, C., Y. Xue, X. Liu, C. Li, and D. Tao. 2019. "Active Transfer Learning Network: A Unified Deep Joint Spectral-Spatial Feature Learning Model for Hyperspectral Image Classification." *IEEE Transactions on Geoscience and Remote Sensing* 57 (3): 1741–1754.

Deng, Cheng, Xianglong Liu, Chao Li, and Dacheng Tao. 2018. "Active multi-kernel domain adaptation for hyperspectral image classification." *Pattern Recognition* 77: 306 – 315.

Deng, J., W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. 2009. "ImageNet: A large-scale hierarchical image database." In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June, 248–255.

Donahue, Jeff, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell. 2014. "DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition." In *Proceedings of the 31st International Conference on Machine Learning*, Vol. 32 of *Proceedings of Machine Learning Research*, 22–24 Jun, 647–655. PMLR.

Farrell, M. D., and R. M. Mersereau. 2005. "On the impact of PCA dimension reduction for hyperspectral detection of difficult targets." *IEEE Geoscience and Remote Sensing Letters* 2 (2): 192–195.

Han, J., D. Zhang, G. Cheng, N. Liu, and D. Xu. 2018. "Advanced Deep-Learning Techniques for Salient and Category-Specific Object Detection: A Survey." *IEEE Signal Processing Magazine* 35 (1): 84–100.

He, K., X. Zhang, S. Ren, and J. Sun. 2016. "Deep Residual Learning for Image Recognition." In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June, 770–778.

He, L., J. Li, C. Liu, and S. Li. 2018. "Recent Advances on Spectral-Spatial Hyperspectral Image Classification: An Overview and New Guidelines." *IEEE Transactions on Geoscience*

and Remote Sensing 56 (3): 1579–1597.

Huh, Minyoung, Pulkit Agrawal, and Alexei A Efros. 2016. "What makes ImageNet good for transfer learning?" *arXiv preprint arXiv:1608.08614* .

Ioffe, Sergey, and Christian Szegedy. 2015. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, 448–456. JMLR.org.

Jiao, L., M. Liang, H. Chen, S. Yang, H. Liu, and X. Cao. 2017. "Deep Fully Convolutional Network-Based Spatial Distribution Prediction for Hyperspectral Image Classification." *IEEE Transactions on Geoscience and Remote Sensing* 55 (10): 5585–5599.

Lin, J., R. Ward, and Z. J. Wang. 2018. "Deep Transfer Learning for Hyperspectral Image Classification." In *2018 IEEE 20th International Workshop on Multimedia Signal Processing (MMSP)*, Aug, 1–5.

Long, J., E. Shelhamer, and T. Darrell. 2015. "Fully convolutional networks for semantic segmentation." In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June, 3431–3440.

Melgani, F., and L. Bruzzone. 2004. "Classification of hyperspectral remote sensing images with support vector machines." *IEEE Transactions on Geoscience and Remote Sensing* 42 (8): 1778–1790.

Pan, S. J., and Q. Yang. 2010. "A Survey on Transfer Learning." *IEEE Transactions on Knowledge and Data Engineering* 22 (10): 1345–1359.

Pratt, Lorien Y. 1993. "Discriminability-based transfer between neural networks." In *Advances in neural information processing systems*, 204–211.

Romero, A., C. Gatta, and G. Camps-Valls. 2016. "Unsupervised Deep Feature Extraction for Remote Sensing Image Classification." *IEEE Transactions on Geoscience and Remote Sensing* 54 (3): 1349–1362.

Ruder, Sebastian. 2016. "An overview of gradient descent optimization algorithms." *arXiv preprint arXiv:1609.04747* .

Simonyan, Karen, and Andrew Zisserman. 2015. "Very Deep Convolutional Networks for Large-Scale Image Recognition." In *International Conference on Learning Representations*, .

Wang, Wenju, Shuguang Dou, Zhongmin Jiang, and Liujie Sun. 2018. "A Fast Dense Spectral-Spatial Convolution Network Framework for Hyperspectral Images Classification." *Remote Sensing* 10 (7).

Windrim, L., A. Melkumyan, R. J. Murphy, A. Chlingaryan, and R. Ramakrishnan. 2018. "Pretraining for Hyperspectral Convolutional Neural Network Classification." *IEEE Transactions on Geoscience and Remote Sensing* 56 (5): 2798–2810.

Wu, H., and S. Prasad. 2018. "Semi-Supervised Deep Learning Using Pseudo Labels for Hyperspectral Image Classification." *IEEE Transactions on Image Processing* 27 (3): 1259–1270.

Yang, J., Y. Zhao, and J. C. Chan. 2017. "Learning and Transferring Deep Joint Spectral-Spatial Features for Hyperspectral Classification." *IEEE Transactions on Geoscience and Remote Sensing* 55 (8): 4729–4742.

Yao, Y., and G. Doretto. 2010. "Boosting for transfer learning with multiple sources." In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June, 1855–1862.

Yosinski, Jason, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. "How transferable are features in deep neural networks?" In *Advances in Neural Information Processing Systems 27*, 3320–3328. Curran Associates, Inc.

Zhong, Z., J. Li, Z. Luo, and M. Chapman. 2018. "Spectral-Spatial Residual Network for Hyperspectral Image Classification: A 3-D Deep Learning Framework." *IEEE Transactions on Geoscience and Remote Sensing* 56 (2): 847–858.