A Deep Belief Network for Classifying Remotely-Sensed Hyperspectral Data

Justin H. Le, Ali Pour Yazdanpanah(✉), Emma E. Regentova, and Venkatesan Muthukumar

Department of Electrical and Computer Engineering, University of Nevada, Las Vegas, Las Vegas, USA
pouryazd@unlv.nevada.edu

Abstract. Improving the classification accuracy of remotely sensed data is of paramount interest for science and defense applications. In this paper, we investigate deep learning architectures (DLAs), whose popularity has grown recently due to the discovery of efficient algorithms to train them, one of which, unsupervised pre-training, seeks to initialize the learned model in a way that greatly encourages efficient supervised learning. We propose a structure for a DLA, the deep belief network (DBN), suitable for the classification of remotely-sensed hyperspectral data. To arrive at this structure, we first study the role of the DBN's width and the duration of pre-training in the learning of features used for the multiclass discrimination of spectral data. We then study the effect of exploiting joint spectral-spatial information. The support vector machine (SVM) is used as a baseline to determine that the proposed method is feasible, offering consistently high classification accuracies in comparison.

1 Introduction

Research in hyperspectral data classification has focused primarily on either feature selection (FS) or feature extraction (FE) as a way to prime the data for use by a classifier [1–3,22,23]. FS involved the limiting of data to an appropriate subset of spectral bands, and FE involved the transformation of the data into a space of lesser dimensionality [2]. Being especially well-suited for the classification of high-dimensional data with few training samples, support vector machines (SVMs) were a prime candidate for the classification of hyperspectral data and could have potentially removed the need for further research in dimensionality reduction [3–5]. Interest in feature extraction remains strong, however, as methods that exploit joint spectral-spatial information have been shown to outperform SVMs that use untransformed data [3,6–9].

We are primarily interested in deep learning architectures (DLA) that perform FE effectively for hyperspectral data. DLAs are a recent development in neural networks that have achieved historically high classification accuracies and, in many cases, have yet to be outperformed by other methods [10–12,24]. It has been suggested that their ability to generalize comes closer to hard artificial

G. Bebis et al. (Eds.): ISVC 2015, Part I, LNCS 9474, pp. 682–692, 2015.
DOI: 10.1007/978-3-319-27857-5_61

intelligence than do other state-of-the-art learning machines [13–15]. Two similar types of DLA, the deep belief network (DBN) and the stacked autoencoder, have been shown to outperform SVMs in the classification of hyperspectral data [16–18]. These experiments sought to determine how the depth of the networks and the number of principal components of the data would affect classification accuracy and running time. In this paper, we seek to further study how the structure of a DBN affects its performance by observing how the accuracy is affected by the width of the network and the duration of training. Based on the results of our study, we propose a structure for a DBN that allows it to achieve high classification accuracies on remotely-sensed hyperspectral data, using a support vector machine (SVM) as a reference for comparison.

1.1 Why Deep Architectures?

We define a local estimator to be a learning machine that classifies an input x under the assumption that the target function $f(x)$ is smooth. That is, after learning from a data point (x_i, y_i), the estimator outputs a y similar to y_i if x is similar to x_i. Examples of local estimators are kernel machines such as the support vector machine and nonparametric methods such as the k-nearest neighbor algorithm. Local estimators have two major disadvantages [13–15]. Firstly, under the smoothness assumption, each new input x can only be predicted if the estimator has previously seen training data x_i in the neighborhood of x, and the estimator would thus require many training examples in order to generalize well. For target functions that have many complex variations, the required amount of training data becomes unmanageably large. Secondly, good local estimators employ either well-designed features or an appropriate kernel for a particular task, which requires prior knowledge of the task that is often unavailable.

These issues can be addressed by learning representations. Particularly, the learned representations must be compact and distributed [13–15, 19]. Here, "distributed" refers to the effect of representing an input x by features that are not mutually exclusive, i.e., that encode information about x through their collective pattern of activation rather than through the local, independent activation of each feature. Distributed representations can be compact in the sense that they may require exponentially fewer values to encode information about x than would be required by one-hot representations such as those learned by local estimators. For example, the local (one-hot) representation of $x = 7$ in binary digits is 1000000, whereas its distributed (base-2) representation is 111. This shows that a distributed representation can encode information in fewer digits, or features, than a local one.

Furthermore, the expressive power of distributed representations allows them to be reused in forming more abstract representations [13–15, 19, 20]. It is expected that, at higher levels of abstraction, the learned representation can capture any essential factors that explain the data and shed any inessential details that would otherwise complicate it. A machine that learns such high-level representations can generalize well, even when the target function varies greatly in the input space, as the prediction does not rely solely on the target function's

smoothness but also on the assumption that descriptive patterns in the data can be discovered and used in explaining unseen data. The reuse of learned representations can be achieved by learning a representation at each layer of a deep neural network, passing the representation learned at each layer as the input to the next layer. With the advent of efficient training algorithms and highly-parallel computing hardware, this deep approach to learning has become feasible [10,20].

2 Method

To explain the structure and theory of the DBN, we first describe its main component, the Restricted Boltzmann Machine (RBM) and the algorithm used to train it. We then describe how training is performed for the DBN as a whole and the parameters involved in the construction and training of the DBN. We refer to these parameters henceforth as hyper-parameters to distinguish them from the parameters of the learned model (the weights and biases that connect the layers of the DBN).

2.1 Restricted Boltzmann Machines

A distributed representation can be formed by a restricted Boltzmann machine (RBM), an undirected graphical model composed of two layers of units, one visible $\mathbf{v}$ and one hidden $\mathbf{h}$, with full connections between the two layers. The RBM is an energy-based model; the joint distribution of the layers is expressed as the Boltzmann distribution, a function of the energy of the joint configuration of units:

$$P(\mathbf{h}|\mathbf{v}) = \frac{\exp(-\text{Energy}(\mathbf{v}, \mathbf{h}; \theta))}{\sum_{\mathbf{h}} \exp(-\text{Energy}(\mathbf{v}, \mathbf{h}; \theta))}. \tag{1}$$

θ is the model to be updated during learning and includes weights $\mathbf{w}$ between pairs of connected units, visible biases $\mathbf{b}$, and hidden biases $\mathbf{c}$. As in the definition borrowed from statistical physics, states of high energy have a low probability of occurrence. Units within the same layer are not connected (hence the name "Restricted"), which allows the energy function to be linear in both $\mathbf{v}$ and $\mathbf{h}$ and as a result, to factorize well, leading to an expression for the conditional probability that can be efficiently computed [14]. For the case where $v_i = \{0, 1\}^I$ and $h_j = \{0, 1\}^J$, where I and J are the number of visible and hidden units, respectively:

$$P(h_j|\mathbf{v}) = \text{sigmoid}(c_j + \sum_{i \in I} W_{ij} v_i), \tag{2}$$

where W_{ij} is the matrix of weights between visible and hidden units. Symmetry across the two layers allows the derivation of Eq. 2 to be repeated for the reverse conditional case [14]:

$$P(v_i|\mathbf{h}) = \text{sigmoid}(c_i + \sum_{j \in J} W_{ij} h_j), \tag{3}$$

which enables the RBM to reconstruct its visible units. Reconstruction is an efficient way to estimate the log-probability gradient of the visible units with respect to the model and thus leads to learning rules that can be efficiently computed [20]:

$$\Delta w = \epsilon(\mathbf{h_0}\mathbf{v_0'} - P(\mathbf{h_1}. = \mathbf{1}|\mathbf{v_1})\mathbf{v_1'}) \tag{4}$$

$$\Delta b = \epsilon(\mathbf{h_0} - P(\mathbf{h_1}. = \mathbf{1}|\mathbf{v_1})) \tag{5}$$

$$\Delta c = \epsilon(\mathbf{v_0} - \mathbf{v_1}), \tag{6}$$

where Δ is the change in a parameter per update, ϵ is the learning rate (the scaling factor of the change per update), and $\mathbf{v_k}$ and $\mathbf{h_k}$ are the k-th sample of the visible and hidden vectors' probability distributions as computed by Eqs. 3 and 2, respectively. The symbol $'$ denotes the transpose of a vector. This method of approximation is known as contrastive divergence [10,19].

2.2 Deep Belief Networks

RBMs can be stacked by feeding the hidden vector of one RBM as the input (visible vector) of another RBM. We use the term "layer" to refer to a layer of model parameters between two vectors of visible or hidden units. In the layers beyond the first one, the conditional probabilities given by Eqs. 2 and 3 can be rewritten in the general form

$$P(h_j^{(\ell+1)}|\mathbf{h}^{(\ell)}) = \text{sigmoid}(\mathbf{c_i^{(\ell+1)}} + \sum_{\mathbf{j}\in\mathbf{J}} \mathbf{W_{ij}^{(\ell+1)}}\ \mathbf{h_j^{(\ell)}}) \tag{7}$$

where $\mathbf{h}^{(\ell)}$ denotes the input vector of the RBM at the ℓ-th layer of the network, and $\mathbf{h^{(0)}}$ represents the input vector of the DBN [10,20]. When the network is constructed in this way and trained layerwise in an unsupervised fashion using Eq. 7, it is known as a Deep Belief Network (DBN). Although a DBN can be used as a generative model due to the RBM's ability to reconstruct data, we are only interested here in the discriminative case, as our intention is to classify data rather than to reconstruct it. To use a DBN discriminatively, the output of the top-layer RBM is fed as input to a classifier such as a logistic regression, at which point, the model is then trained in a supervised fashion [10,20], as shown in Fig. 1.

The unsupervised training stage is referred to as pre-training. Its purpose is to initialize the model in such a way as to improve the efficiency of supervised training. The supervised training stage is referred to as fine-tuning, as it adjusts the classifier's prediction to match the ground truth of the data. Each iteration of pre-training or fine-tuning is referred to as an epoch [10,21].

In order for accurate predictions to be made, the DBN must learn to produce "good" representations. A good representation is one that captures only the most essential underlying patterns of the data and discards the rest. The best representation is one which not only captures these patterns, but disentangles them from each other, just as the human mind learns to recognize an object

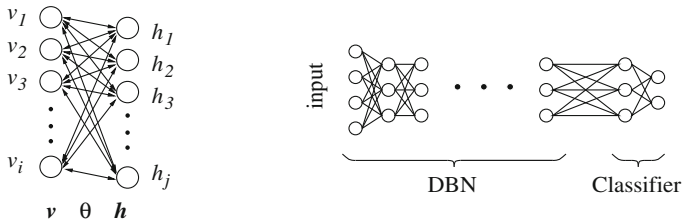

Fig. 1. Left: An RBM with i visible units, j hidden units, and model parameters θ. Right: A DBN with arbitrary width and a classifier, e.g., softmax, at the output

by disentangling it from the shadows cast upon it [13]. These definitions give rise to the term "representation learning". The term "deep learning" refers to the implementation of this theory through a learning machine that consist of many layers, such as a DBN. In this paper, the DBN is chosen over other deep architectures as we are interested in studying one particular technique used by the DBN, pre-training, whose nature has yet to be fully understood [13].

3 Experiments

To study the performance of the DBN across multiple settings and ensure that it yields consistently high accuracies, we conduct experiments in classification with spectral signatures, in which we vary the hyper-parameters whose effect on performance would be unpredictable without empirical guidance. We then repeat the experiments using joint spectral-spatial information in an attempt to support or dispute the evidence of any trends observed in spectral classification. In this section, we describe the dataset, our choice of hyper-parameters and their justification, and the process of learning using spectral and spectral-spatial data.

3.1 Data

Pavia Centre is a hyperspectral dataset with 102 bands and 1096-by-715 pixels, depicting a scene from the town of Pavia in Italy. Figure 4 depicts a false-color map of the data. The data was collected by the ROSIS sensor at a geometric resolution of 1.3 m. The sensor has a bandwidth of 4 nm and a wavelength range of 400 to 900 nm.

The nine materials present in Pavia Centre are labelled according to the color scheme shown in Fig. 4. Pixels that belong to the zero class are those that do not belong to any of these nine classes. Zero-class pixels have their own spectral signatures, but they must be excluded from the training process, as they are so numerous (relative to other classes) that they would skew the learning process in a way that causes the classifier to predict only zero-class pixels. Training was thus performed using only the pixels that belong to one of the nine labelled classes. Particularly, training was performed using a randomly selected half of

Table 1. The labels, names, and number of training and validation samples of each land cover class in the spectral and spectral-spatial datasets

Label	Class	Number of samples					
		Spectral data			Spectral-spatial data		
		Training	Validation	Total	Training	Validation	Total
1	Water	32986	32985	65971	32662	32661	65323
2	Trees	3799	3799	7598	3767	3767	7534
3	Asphalt	1545	1545	3090	1470	1470	2940
4	Bricks	1343	1342	2685	1343	1342	2685
5	Bitumen	3292	3292	6584	3285	3285	6570
6	Tiles	4624	4624	9248	4590	4589	9179
7	Shadows	3644	3643	7287	3644	3643	7287
8	Meadows	21413	21413	42826	21190	21189	42379
9	Bare soil	1432	1431	2863	1429	1428	2857

this set, with the other half of the pixels reserved for validation. Table 1 shows how many pixels from each class are present in the dataset.

Each pixel of the Pavia Centre dataset is treated as a single data point. The values at the input (visible) layer are normalized with respect to the maximum value across all pixels and all bands (a value of 8000).

3.2 Variation of Hyper-Parameters

We define width to be the number of units in a hidden vector $\mathbf{h}^{(\ell)}$ for $\ell > 0$, as described by Eq. 7, and depth to be the number of layers of the model. It is expected that accuracy will either improve or remain roughly the same as the number of fine-tuning epochs increases, and so we fix the fine-tuning duration at 20,000 epochs. In this paper, we are only interested in studying the width of the DBN and the pre-training duration, and so, we wish to fix the depth. We choose a depth of 4 for spectral classification and 3 for spectral-spatial, as suggested by the experimental results in [18].

Given that the fine-tuning duration and depth are fixed, the hyper-parameters that chiefly interest our study are the width of the layers and the pre-training duration. We perform a grid-search of several configurations of these two hyper-parameters as given in Tables 2 and 3. The widths vary in increments of 5 from 20 to 35. These values are chosen because they correspond to a "valley" in the classification error; we found that widths beyond this range tend toward higher errors regardless of other hyper-parameter settings. For the same reason, we vary the pre-training duration between 1000 and 3000 epochs.

For all experiments, we use binary RBM units and apply the equations given in Sects. 2.1 and 2.2.

3.3 Learning from Spectral Features

The input vector is defined as $\mathbf{h}^{(\ell)}$ in Eq. 7 for $\ell = 0$ and represents the visible vector of the first-layer RBM. To learn from spectral signatures, we construct the input vector as a pixel of the hyperspectral image, as shown in Fig. 2, and its length is thus equal to the number of spectral bands (102). The first RBM undergoes pre-training by first updating the binary values and probabilities of activation of its hidden units using Eq. 2, then reconstructing the visible units (input vector) using Eq. 3. To obtain the conditional probability used in Eqs. 4 and 5 for updating the weights and visible biases, respectively, the probabilities of the hidden units are then computed again using the reconstructed visible vector as the prior. This process is repeated for each data point (pixel). An epoch is defined as the time taken for this process to take place over all data points in a layer. Once the specified number of epochs have taken place in the first layer, the process is repeated for all subsequent layers using the hidden vector from the previous layer as the visible (input) vector of the current layer, as described by Eq. 7.

Table 2 compares the DBN's performance to that of a radial basis function kernel support vector machine (RBF-SVM). "Best Epoch" indicates the epoch that produced the highest overall accuracy (OA) for a certain configuration. OA is measured as the total number of correctly classified samples to the total number of samples, whereas average accuracy (AA) is measured as the average across all classes of the number of correctly classified samples in a class to the number of samples in that class. The Kappa statistic (κ) measures the agreement between these two accuracies. The DBN outperforms the SVM when constructed with a width of 25 and pre-trained for 1000 epochs, yielding an OA of 97.928 %, an improvement of 2.528 %. Although the SVM yielded a higher AA than the DBN, the DBN yields a higher κ in the case when it achieves the highest OA.

3.4 Learning from Spectral-Spatial Features

Spatial information conveys the probability that the given pixel belongs to a particular, e.g., a pixel is likely to depict soil if it is surrounded by other soil pixels, as soil often appears in patches that are spanned by multiple pixels. To extract spatial information, a neighborhood of 7-by-7 pixels is chosen as a

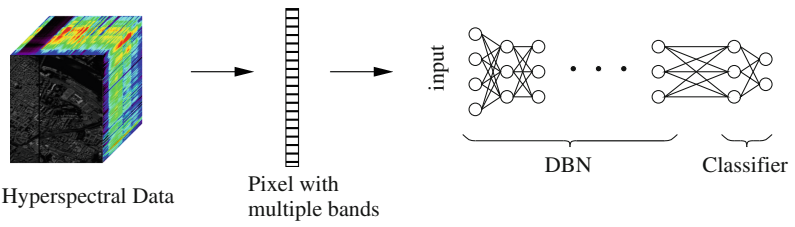

Fig. 2. Training the DBN pixelwise with spectral signatures

data point. The center pixel of this neighborhood represents this data point's coordinate. Hence, the label of the center pixel is used as the label for the data vector extracted in this manner. The values are extracted in row major fashion. Due to the immensity of these data points, the number of bands for each pixel in a neighborhood must be reduced to 5 using principal component analysis, as suggested by [18]. Each reduced data point obtained this way is then stacked onto the corresponding data point used in the spectral classification method to form a combined spectral-spatial feature vector, which is then fed as input to the DBN, as shown in Fig. 3.

Unlike the previous experiment, a width of 35 yields the lowest accuracy when the duration of pre-training is fixed, and the best case is achieved with more pre-training epochs rather than less. The other three cases show that fewer epochs yield higher accuracy with fixed width.

Table 2. Overall accuracy, average accuracy, Kappa statistic, and the epoch at which the highest accuracy was reached for spectral classification

Pre-training epochs	Width	OA (%)	AA (%)	κ	Best epoch
1000	20	92.2828	89.6493	0.2544	19888
	25	**97.9285**	89.7680	**0.7975**	19737
	30	94.3058	89.7029	0.4470	4492
	35	97.9224	89.7644	0.7970	17954
3000	20	92.5076	89.6689	0.2748	15736
	25	93.3237	89.7156	0.3508	18099
	30	95.2784	89.7092	0.5412	19881
	35	94.0290	89.7113	0.4197	7776
	SVM	95.4054	**89.8087**	0.5492	

The results of the two experiments show that a greater number of pre-training epochs does not necessarily yield higher accuracy, although in some cases it does. It is possible that further experiments would show no correlation between pre-training duration and accuracy. In contrast, the results suggest that a width of 25 is ideal for learning the features of the Pavia Centre dataset, as both spectral and spectral-spatial classification reach a peak in accuracy when using this width.

3.5 Logistic Regression

The final hidden vector of the DBN is fed as input to a logistic regression stage in order to learn a model for predicting the class to which a data point belongs, as depicted by the Classifier in Fig. 1. Learning in this layer is referred to as fine-tuning and is performed by stochastic gradient descent. As in pre-training, an epoch is defined as the time taken for fine-tuning to take place over all data points.

Table 3. Overall accuracy, average accuracy, Kappa statistic, and the epoch at which the highest accuracy was reached for spectral-spatial classification

Pre-training epochs	Width	OA (%)	AA (%)	κ	Best epoch
1000	20	97.3268	83.5989	0.8370	18897
	25	97.3431	83.8029	0.8360	18247
	30	97.8474	83.7758	0.8673	18543
	35	96.3612	83.1496	0.7841	19832
3000	20	95.0835	83.6071	0.7001	12906
	25	**98.1233**	83.7873	**0.8842**	19921
	30	95.6395	83.6134	0.7339	17591
	35	93.7642	83.3949	0.6245	6415
	SVM	89.3068	**83.8555**	0.3377	

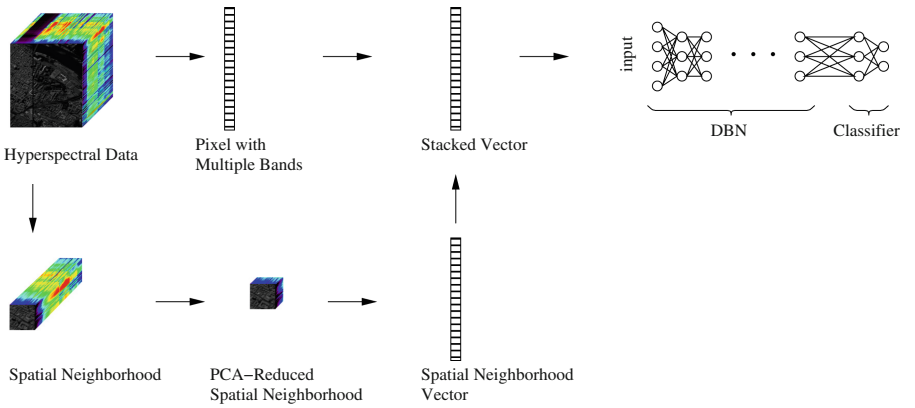

Fig. 3. Stacking spectral data onto spatial data that is constructed by flattening PCA-reduced neighborhoods

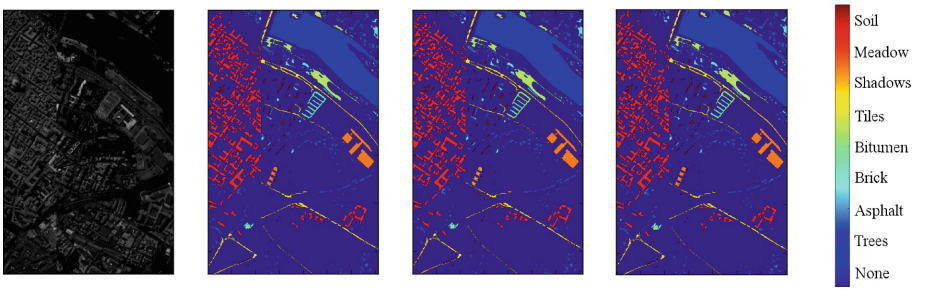

Fig. 4. From left to right: the Pavia Centre data in grayscale, the color maps for the ground truth and labels predicted by spectral and spectral-spatial classification, and the color scale (Color figure online)

In the experiments that produced the highest accuracies, the best epoch occurred near the final epoch of fine-tuning, which shows that in these cases, the accuracy improved even toward the end of training, suggesting that it would continue to improve had training been extended beyond the maximum number of epochs used.

4 Conclusion

We have presented a new DBN structure for the classification of remotely-sensed hyperspectral images. To achieve the best classification performance, we investigated the effect of the DBN's width and pre-training time on multi-class discrimination by spectral signatures. Then, we incorporated spatial information to form a spectral-spatial input, which improved the accuracy for five out of eight cases and produced the best accuracy. The performance of the achieved architecture was compared with that of the radial basis function kernel support vector machine.

Acknowledgements. This research was supported by NASA EPSCoR under cooperative agreement No. NNX10AR89A.

References

1. Bruce, L., Koger, C., Li, J.: Dimensionality reduction of hyperspectral data using discrete wavelet transform feature extraction. IEEE Trans. Geosci. Remote Sens. **40**, 2331–2338 (2002)
2. Harsanyi, J., Chang, C.I.: Hyperspectral image classification and dimensionality reduction: an orthogonal subspace projection approach. IEEE Trans. Geosci. Remote Sens. **32**, 779–785 (1994)
3. Kang, X., Li, S., Benediktsson, J.: Spatial hyperspectral image classification with edge-preserving filtering. IEEE Trans. Geosci. Remote Sens. **52**, 2666–2677 (2014)
4. Melgani, F., Bruzzone, L.: Classification of hyperspectral remote sensing images with support vector machines. IEEE Trans. Geosci. Remote Sens. **42**, 1778–1790 (2004)
5. Pal, M., Foody, G.: Feature selection for classification of hyperspectral data by SVM. IEEE Trans. Geosci. Remote Sens. **48**, 2297–2307 (2010)
6. Li, J., Bioucas-Dias, J., Plaza, A.: Hyperspectral image segmentation using a new bayesian approach with active learning. IEEE Trans. Geosci. Remote Sens. **49**, 3947–3960 (2011)
7. Li, J., Bioucas-Dias, J.M., Plaza, A.: Semisupervised hyperspectral image classification using soft sparse multinomial logistic regression. IEEE Geosci. Remote Sens. Lett. **10**, 318–322 (2013)
8. Li, J., Bioucas-Dias, J., Plaza, A.: Spatial classification of hyperspectral data using loopy belief propagation and active learning. IEEE Trans. Geosci. Remote Sens. **51**, 844–856 (2013)
9. Bernard, K., Tarabalka, Y., Angulo, J., Chanussot, J., Benediktsson, J.: Spatial classification of hyperspectral data based on a stochastic minimum spanning forest approach. IEEE Trans. Image Process. **21**, 2008–2021 (2012)

10. Hinton, G.E., Osindero, S., Teh, Y.W.: A fast learning algorithm for deep belief nets. Neural Comput. **18**, 1527–1554 (2006)
11. Larochelle, H., Erhan, D., Courville, A., Bergstra, J., Bengio, Y.: An empirical evaluation of deep architectures on problems with many factors of variation. In: Proceedings of the 24th International Conference on Machine Learning, ICML 2007, pp. 473–480. ACM, New York (2007)
12. Hinton, G.E., Salakhutdinov, R.R.: Reducing the dimensionality of data with neural networks. Science **313**, 504–507 (2006)
13. Bengio, Y., Courville, A.C., Vincent, P.: Unsupervised feature learning and deep learning: a review and new perspectives. CoRR abs/1206.5538 (2012)
14. Bengio, Y.: Learning deep architectures for AI. Found. Trends Mach. Learn. **2**, 1–127 (2009)
15. Bengio, Y., Lecun, Y., Operationnelle, D.D.E.R., Montreal, U.D.: Scaling learning algorithms towards AI. In: Bottou, L., Chapelle, O., DeCoste, D., Weston, J. (eds.) Large-Scale Kernel Machines. MIT Press, Cambridge (2007)
16. Lin, Z., Chen, Y., Zhao, X., Wang, G.: Spectral-spatial classification of hyperspectral image using autoencoders. In: 2013 9th International Conference on Information, Communications and Signal Processing (ICICS), pp. 1–5 (2013)
17. Chen, Y., Lin, Z., Zhao, X., Wang, G., Gu, Y.: Deep learning-based classification of hyperspectral data. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **7**, 2094–2107 (2014)
18. Chen, Y., Zhao, X., Jia, X.: Spectral-spatial classification of hyperspectral data based on deep belief network. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **8**, 2381–2392 (2015)
19. Le Roux, N., Bengio, Y.: Representational power of restricted boltzmann machines and deep belief networks. Neural Comput. **20**, 1631–1649 (2008)
20. Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., Montral, U.D., Qubec, M.: Greedy layer-wise training of deep networks. In: Schölkopf, B., Platt, J., Hoffman, T. (eds.) NIPS. MIT Press, Cambridge (2007)
21. Bengio, Y.: Practical recommendations for gradient-based training of deep architectures. CoRR abs/1206.5533 (2012)
22. Serpico, S., Bruzzone, L.: A new search algorithm for feature selection in hyperspectral remote sensing images. IEEE Trans. Geosci. Remote Sens. **39**, 1360–1367 (2001)
23. Chang, C.I., Du, Q., Sun, T.L., Althouse, M.: A joint band prioritization and band-decorrelation approach to band selection for hyperspectral image classification. IEEE Trans. Geosci. Remote Sens. **37**, 2631–2641 (1999)
24. Makantasis, K., Karantzalos, K., Doulamis, A., Doulamis, N.: Deep supervised learning for hyperspectral data classification through convolutional neural networks. In: IGARSS, pp. 1771–1800 (2015)